Democracy Live Inc.
Technical Response to:

Request for Proposal
West Virginia Secretary of State
CRFP SOS2400000001

For: Election Division E-Ballot Delivery Technology

Event	Date	Time
Bid Submission Date	**November 08, 2023**	**1:30 PM EST**

RFP Issued By:
State of West Virginia
Department of Administration
Purchasing Division
2019 Washington Street East
Charleston, WV 25305-0130

Bidder: Democracy Live, Inc.
Bryan Finney, President/CEO
(206) 465-5636
bryan@democracylive.com

Submitted on November 07, 2023

Department of Administration
Purchasing Division
2019 Washington Street East
Post Office Box 50130
Charleston, WV 25305-0130

State of West Virginia
Centralized Request for Proposals
Info Technology

Proc Folder: 1288892 **Doc Description:** Addendum No. 2 WVSOS Election Division E-Ballot Delivery Tech **Proc Type:** Central Contract - Fixed Amt	**Reason for Modification:** Addendum No. 2 is issued to publish vendor questions and responses, and to modify the specifications.

Date Issued	Solicitation Closes	Solicitation No	Version
2023-10-27	2023-11-08 13:30	CRFP 1600 SOS2400000001	3

BID RECEIVING LOCATION

BID CLERK
DEPARTMENT OF ADMINISTRATION
PURCHASING DIVISION
2019 WASHINGTON ST E
CHARLESTON WV 25305
US

VENDOR

Vendor Customer Code: VS0000026409

Vendor Name Democracy Live, Inc.

Address: 2900 NE Blakeley Street

Street:

City: Seattle

State: WA **Country:** USA **Zip:** 98105

Principal Contact: Bryan Finney, President

Vendor Contact Phone: 206-465-5636 **Extension:**

FOR INFORMATION CONTACT THE BUYER
Toby L Welch
(304) 558-8802
toby.l.welch@wv.gov

Vendor Signature X **FEIN# 45-4826119** **DATE 11-06-2023**

All offers subject to all terms and conditions contained in this solicitation

ADDITIONAL INFORMATION

Addendum No. 2 is issued for the following reasons:

1) To publish a copy of vendor questions with their responses.
2) To modify specifications. (4.1.3.3)

--no other changes--

**** Online responses has been prohibited for this solicitation, if you have questions contact the Buyer - Toby Welch @ toby.l.welch@wv.gov

See attached instructions for requirements for responding.

INVOICE TO	SHIP TO
SECRETARY OF STATE BLDG 1 STE 157K 1900 KANAWHA BLVD E CHARLESTON WV 25305-0770 Us	SECRETARY OF STATE BLDG 1 STE 157K 1900 KANAWHA BLVD E CHARLESTON WV 25305-0770 us

Line	Comm Ln Desc	Qty	Unit of Measure	Unit Price	Total Price
	Election E-Ballot Delivery Technology system	1.00000	EA		

Comm Code	Manufacturer	Specification	Model#
81112200			

Extended Description:
Vendors must fill out Cost Sheet included as an attachment.
ONLINE SUBMISSIONS OF REQUESTS FOR PROPOSAL ARE PROHIBITED

SCHEDULE OF EVENTS

Line	Event	Event Date
1	Questions are due by 3:00 p.m.	2023-10-06

	Document Phase	Document Description	Page 3
SOS2400000001	Final	Addendum No 2 WVSOS Election Division E-Ballot Delivery Tech	

ADDITIONAL TERMS AND CONDITIONS

See attached document(s) for additional Terms and Conditions

REQUEST FOR PROPOSAL

West Virginia Secretary of State
CRFP SOS2400000001
Election E-Ballot Delivery Technology

Table of Contents

DEMOCRACYLIVE
VOTER INFORMATION TECHNOLOGIES

In partnership with aws

November 6, 2023

Toby L. Welch
toby.l.welch@wv.gov
Bid Clerk
Department of Administration
Purchasing Division
2019 Washington St E
Charleston, WV 25305

Reference: **Cover Letter – Request for Proposal # CRFP SOS2400000001**
Election Division E-Ballot Delivery Technology
Bid Response

Dear Bid Clerk:

Thank you for the opportunity to respond to your RFP for a secure, accessible remote ballot delivery and return system. It has been an honor to support the State of West Virginia since 2020 on this important initiative.

Since 2008, Democracy Live has delivered secure, accessible remote balloting solutions in over 5,000 elections to more than 2,500 jurisdictions across 34 states in the U.S. Democracy Live, in partnership with Amazon (AWS), has the proven experience and background to ensure West Virginia's requirements and expectations for a secure ballot delivery and return system are fully met.

Democracy Live developed and deployed the nation's first remote accessible absentee balloting technology over 15 years ago. Evolving and improving year over year, the Democracy Live OmniBallot system has been reviewed, selected and deployed in more elections than all other remote balloting active providers combined in the U.S.

Since 2008, over 20 million U.S. voters in the U.S. and over 120 countries have had access to OmniBallot for either accessible absentee, UOCAVA and/or accessible sample ballots.

As a founding member of the Department of Homeland Security sponsored Elections Sector Executive Committee (SCC) and founding Chair of the SCC Emergency Response Task Force, I understand security is our key priority. Together with our partners at AWS, Democracy Live is confident the State of West Virginia will receive a best-of-breed team that offers the most experience, stability, scalability and security for West Virginia's UOCAVA and accessible remote electronic balloting needs.

Sincerely,

Bryan D. Finney
President/CEO
Democracy Live

DESIGNATED CONTACT: Vendor appoints the individual identified in this Section as the Contract Administrator and the initial point of contact for matters relating to this Contract.

Bryan Finney, President

2900 NE Blakeley Street, Seattle, WA 98105
Main (855) 655-VOTE (8683)
Mobile (206) 465-5636
bryan@democracylive.com

CERTIFICATION AND SIGNATURE: By signing below, I certify that: I have reviewed this Solicitation in its entirety; that I understand the requirements, terms and conditions, and other information contained herein; that this bid, offer or proposal constitutes an offer to the State that cannot be unilaterally withdrawn; that the product or service proposed meets the mandatory requirements contained in the Solicitation for that product or service, unless otherwise stated herein; that the Vendor accepts the terms and conditions contained in the Solicitation, unless otherwise stated herein; that I am submitting this bid, offer or proposal for review and consideration; that I am authorized by the vendor to execute and submit this bid, offer, or proposal, or any documents related thereto on vendor's behalf; that I am authorized to bind the vendor in a contractual relationship; and that to the best of my knowledge, the vendor has properly registered with any State agency that may require registration.

By signing below, I further certify that I understand this Contract is subject to the provisions of West Virginia Code § 5A-3-62, Which automatically voids certain contractclauses that violate State law.

Democracy Live, Inc.
(Company)

(Authorized Signature) (Representative Name, Title)

Bryan Finney, President
(Printed Name and Title of Authorized Representative)

11-06-2023
(Date)

206-465-5636
(Phone Number) (Fax Number)

REQUEST FOR PROPOSAL

West Virginia Secretary of StateCRFP SOS2400000001 Election Division E-Ballot Delivery Technology

SECTION 4: PROJECT SPECIFICATIONS

Background and Current Operating Environment:

Following the passage of SB 94 (2020), the Agency is statutorily required to prescribe an electronic ballot transmission and marking tool for the 2024 West Virginia primary and general elections for use by registered West Virginia voter who are eligible for the option of participating in the election(s) via electronic absentee voting under the W. Va. Code§ 3-3-1 *et seq.* (eligible absentee voters living with a physical disability which prevents them from voting independently) as well as the Uniformed and Overseas Citizens Absentee Voting Act ("UOCAVA") set forth in 52 U.S.C.A. §20301 *et seq.* and W. Va. Code§ 3-3-2.

Elections systems and associated technology have been classified by the Department of Homeland Security as Critical Infrastructure since 2017. Electronic voting systems are currently certified and uniform standards created by the United States Election Assistance Commission. However, currently, electronic ballot delivery technologies have no such uniform security standards.

Due to the supreme importance of protecting Critical Infrastructure, rapidly changing elections technology landscape, and regularly evolving federal standards for systems used in Critical Infrastructure sectors, the Agency collaborated with a federal security and compliance company with specialized services and expertise with implementing federally approved cybersecurity controlsfor Critical Infrastructure technology, evaluating acquisition documents and contracts for such technologies, and auditing compliance with such standards for government agencies in Critical Infrastructure sectors.

4.1. Project Goals and Mandatory Requirements:

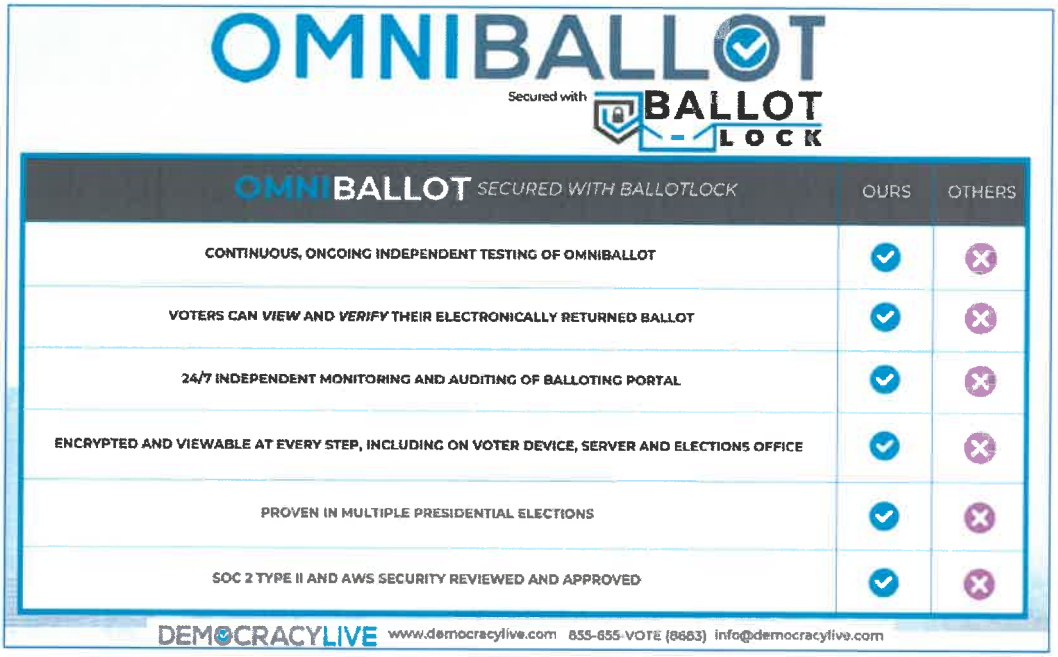
The electronic absentee ballot transmission and marking tool shall be prescribed for use by all 55 West Virginia counties. The Agency will serve in an administrative capacity by ensuring uniformity, providing support, and assisting with issue resolution when necessary. The tool shall comport with all goals and objectives set forth herein and as required by applicable West Virginia and federal laws.

Vendor should describe its approach and methodology to providing the service or solving the problem described by meet the goals/objectives identified below. Vendor's response should includeany information about how the proposed approach is superior or inferior to other possible approaches.

4.1.1. Goals and Objective

Democracy Live Response (Subject to 20 Page RFP Limit): The Democracy Live approach to providing the services requested in this RFP is founded on Democracy Live's 15 years of experience developing, deploying and supporting State and local elections across 34 states in over 5,000 elections. Partnering with Amazon (AWS), the technologies deployed in the field must not only meet the highest standards for security and accessibility, the tools and technology must also be understood and explainable to elections officials, voters, media and the public.

As shown in the comparison chart below, the Democracy Live approach to this solution ensures West Virginia has unmatched levels of security, accessibility, ongoing independent testing, ongoing independent monitoring and end-end auditable ballot confirmation, using verification tools that voters and clerks can describe and easily understand.



4.1.1.1 The Vendor provides an electronic ballot delivery and marking tool to all 55 West Virginia Counties in the State. The tool shall be ready for go-live use by no later than the statutory absentee ballot mailing deadline on March 29, 2024. All development, proofing, training, and other necessary actions shall be complete prior to that date.

Democracy Live Response: OmniBallot has been honored to support the State of West Virginia since 2020 on this important ballot access initiative. Democracy Live offers a proven, turn-key experience specific to West Virgina elections. Partnering with Amazon (AWS) and offering the largest support team for accessible remote balloting, Democracy Live ensures a seamless, consistent, and proven solution for the State and all 55 counties. With our West Virginia specific experience, the State can be assured all development, proofing, training, and other necessary actions will be completed by the State's March 29, 2024 absentee ballot mailing deadline.

As the only electronic ballot return provider proven in multiple Presidential elections, Democracy Live has the demonstrated experience and state-specific knowledge to ensure West Virginia has a proven and established partner to ensure a successful implementation.

Since 2008, Democracy Live has been deployed in over 5,000 elections across 34 states in 2,500 jurisdictions. Of active providers, 90% of all electronic ballot return has been through Democracy Live.

4.1.1.2 The tool satisfies all West Virginia and federal requirements for electronic absentee voting, including but not limited to W. Va. Code§ 3-3-1 *et seq.*, the Uniformed and Overseas Absentee Voting Act, the Military and Overseas Voter Empowerment Act, and the Americans with Disabilities Act.

Democracy Live Response: OmniBallot satisfies all West Virginia and federal requirements for electronic absentee voting. In support of UOCAVA voters and in compliance with the MOVE Act, Democracy Live was the first firm selected directly by the U.S. Department of Defense in the 2012 UOCAVA MOVE Act funding program. Democracy Live technologies continue to be selected for more states, jurisdictions, and funding than all other DoD (FVAP) funded technologies combined.

Working with the University of Washington Center for Technology and Disabilities Democracy Live pioneered the nation's first fully accessible, ADA-compliant absentee balloting technology in 2008. For our leadership in accessible voting, Democracy Live won the 2019 Accessibility in Voting Award, presented at the United Nations. Multiple Democracy Live customers have won national accessibility awards for deploying our accessible balloting technologies.

4.1.1.3. The tool's functionality allows convenient confirmation of voter eligibility, voter identity, and accessibility.

Democracy Live Response: Democracy Live has a proven, easy-to-use voter look-up tool to ensure all eligible voters, whether domestic, or abroad can conveniently and securely confirm their eligibility without delay. Since 2008, over 20 million voters have had access to the Democracy Live voter look-up and credentials tools. Only eligible voters have access to the OmniBallot portal, as only eligible voters are imported into the system. OmniBallot has a proven, robust VR import tool to efficiently update the voter list in the portal.

4.1.2. Mandatory Project Requirements -The following mandatory requirements relate to the goals and objectives and must be met by the Vendor as a part of its submitted proposal. Vendor should describe how it will comply with the mandatory requirements and include any areas whereits proposed solution exceeds the mandatory requirement. Failure to comply with mandatory requirements will lead to disqualification, but the approach/methodology that the vendor uses to comply, and areas where the mandatory requirements are exceeded, will be included in technical scores where appropriate. The mandatory project requirements are listed below.

4.1.2.1 The tool is capable of recognizing and reading each ballot style based on the "Ballot Design" files in the format provided by the Agency, a county, or a county's ballot programming vender.

Democracy Live Response: In collaboration with ES&S, OmniBallot has a proven and well-established ES&S data importing tool, specifically developed for the State of West Virginia. Since our start in 2008, Democracy Live has been importing ballot data from a wide range of tabulation and ballot data systems, including ES&S. The OmniBallot ES&S Importer ensures ballot data can be easily, securely, and accurately

imported to ensure 100% on-time implementations for every election.

4.1.2.2 The tool includes a cloud server or equivalent backend which securely processes each electronic absentee ballot submission into a cast vote record (CVR) format, stores the records in a tamper-resistant manner, and enables all participating counties to access the CVRs as requiredby the election schedule and process for in-county tallying.

Democracy Live Response: OmniBallot is hosted in Amazon Web Services (AWS) which is the largest provider to Federal, State and local governments in the U.S. OmniBallot processes each electronic absentee ballot submission into a cast vote record (CVR) format, stores the records in a federally approved, FedRamp-compliant cloud infrastructure, and enables all participating counties to access the CVRs. Ballots securely stored in AWS, are then printed directly onto paper ballots and processed.

AWS was recently selected by the National Security Agency (NSA) to securely host and secure some of the nation's most classified and critical documents (such as ballots). Like the NSA, OmniBallot leverages AWS to securely store digital versions of documents (ballots), before printing and processing.

4.1.2.3 The tool includes a web-based or equivalent administration console for reporting and tracking voter participation.

Democracy Live Response: The OmniBallot Admin Dashboard is a Web-based console that allows state and local account users to test, view, QA and approve ballots in OmniBallot. From the Admin console, using the OmniBallot Account Manager Module, administrators can run an array of reports on voter participation, site usage, tracking, log reports and other activities. All OmniBallot logs, activity and reporting is available to the State to conduct a fully transparent audit of each election.

4.1.2.4 The tool permits a voter to mark a ballot independently and without assistance.

Democracy Live Response: Voter's using OmniBallot are able to login to the secure balloting portal to access, review, mark, return and confirm their ballot submission.

The OmniBallot ballot marking portal has been designed to meet the highest levels of accessibility, tested with over 90 combinations of screen readers, browsers, operating systems and devices. Since 2008, OmniBallot has been used by voters representing a full array of disabilities. For over fifteen years, voters with Parkinson's, palsy, paralysis, vision loss and other disabilities have used OmniBallot to access their absentee ballot privately, independently, and securely.

Tested by the University of Washington Center on Technology and Disabilities in 2023, OmniBallot fully meets Section 508 WCAG 2.1 accessibility requirements, ensuring OmniBallot permits a voter to mark a ballot independently and without assistance.

As stated by a West Virginia voter, *"There is hardly anything more important than to have our voice heard in the political arena via the voting booth. Unfortunately, transportation for some is difficult, often making it impossible to have easy access to voting. Therefore, the accessible absentee ballots has been an incredibly important part of our lives. The creation of the accessible absentee ballot created by Democracy Live, has been revolutionary and life-changing to individuals who are blind, or have low vision. The software is extremely easy to use with speech and low vision, access, software, consistent, accurate, and secure. Thanks to this amazing software our voices are now heard.*

Sheri Koch, President
National Federation of the Blind of West Virginia

4.1.2.5 The tool provides a voter the option to transmit a marked ballot, along with a return packet that includes the requisite forms and disclosures, to the county clerk electronically, or alternatively to print a voted ballot with the aforementioned return packet for return via other approved means to the county clerk.

Democracy Live Response: All voter's using OmniBallot have the option of either printing their ballot and all required return materials, or electronically returning the ballot and return materials. If electing to electronically return the ballot packet, the Clerk's office will be electronically notified a packet has been submitted and is ready to be reviewed, downloaded, printed and processed. In the OmniBallot Admin tools, using the EBR Module, election administrators can download and print approved ballots and other required return materials directly from the portal.

Importantly, all actions within the portal are independently logged, tracked and monitored for auditing and review purposes. If an elections office has a Ballot on Demand printer, they can print a fully tabulatable ES&S ballot directly from the OmniBallot portal.

Includes All Required Return Materials

The ballot and all required absentee return materials are printed by the voter, or at the Elections office

OMNIBALLOT

4.1.2.5.1 The option for a voter to return a ballot shall be an optional functionality available to the Agency at no cost, and the Vendor shall not be compensated in any manner in the event the Agency opts to allow voters to return a ballot electronically.

Democracy Live Response: OmniBallot offers a suite of optional modules that the State may turn on or off. One of the modules is the OmniBallot EBR module. OmniBallot EBR has been deployed in 20% of the States in the U.S. The OmniBallot EBR module is the only EBR tool deployed in multiple statewide elections during Presidential and federal elections. The OmniBallot EBR module is included for use with the West Virginia OmniBallot Balloting Portal.

4.1.2.6 The tool includes a verification portal that permits a voter to review their marked, submitted ballot, in a secure and anonymous manner, and in a read-only format, affording the voter with the ability to confirm the ballot cast is the ballot received by the county.

Democracy Live Response: Democracy Live offers a proven and tested Verification portal, called the *Verifier*. The Verifier verification portal permits a voter to review their marked, submitted ballot, in a secure and anonymous manner, and in a read-only format.

The Verifier is a proven tool that ensures the voter can easily understand and confirm the actual ballot (PDF)

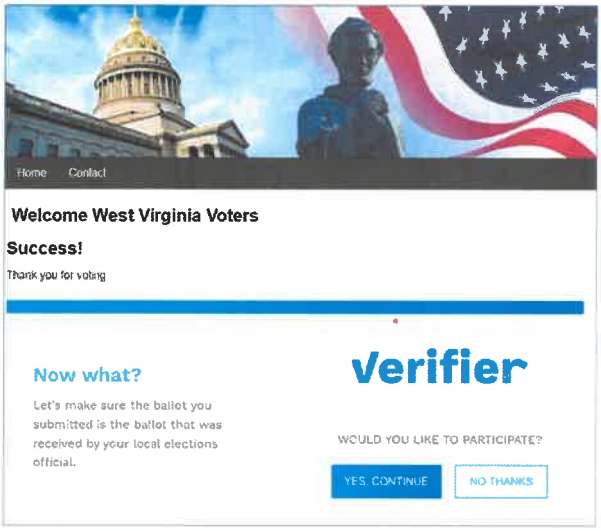
submitted by the voter was the ballot received by the county elections office. The Verifier allows a voter to review their marked, submitted ballot (PDF) after it has been received by the election office.

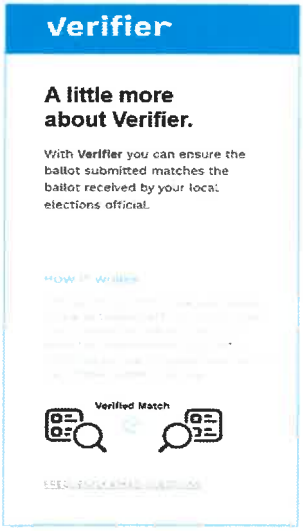
Democracy Live deployed the Verifier in West Virginia in 2022. The Verifier as deployed in West Virginia, offered eligible voters the ability to access any Web connected device to enter a verification code to view and confirm the ballot they submitted was the ballot the election office received.

The Verifier has ongoing, independent, cybersecurity testing from a pool of over 1,500 independent cybersecurity experts. (Via Synack)

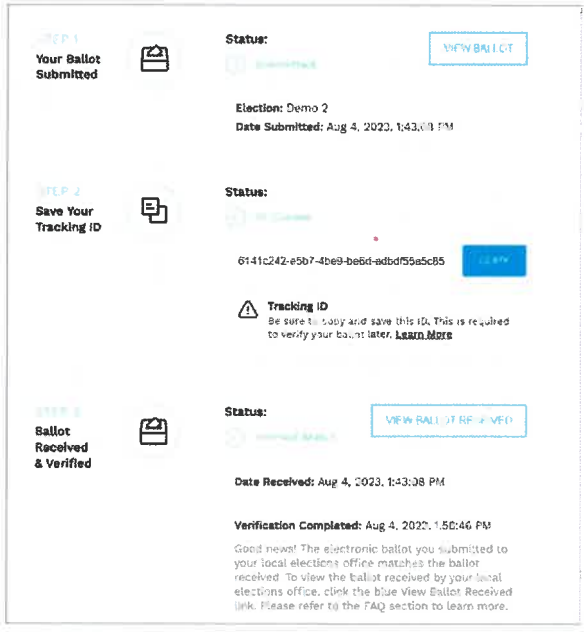
Voter Verification using Verifier.

4.1.2.6.1 Voter submits their ballot. Voter then has the *option* to use the Verifier.



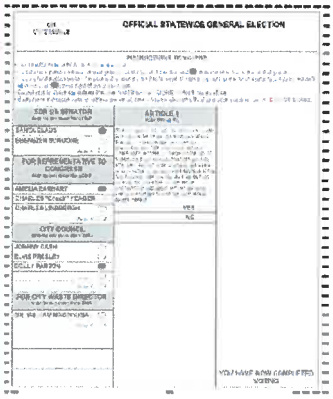


2. Voter enters their personal verification code into Verifier to view and confirm the Ballot Returned was the Ballot Received (BR^2 Verification).



Differences in Voter Verification:
Some verification options in the market attest to securely encrypting the ballot using freely available tools such as ElectionGuard. Democracy Live reviewed ElectionGuard, and while it may be a good option for polling places, it is unproven at scale in public elections. One key factor in our decision was the company that built ElectionGuard (Microsoft) has stated publicly that it does *not* support the use of ElectionGuard for the purposes of electronic ballot return[1].

In our review, while ElectionGuard can validate the voter's encrypted ballot has been received, it does not provide assurance their selections were not modified prior to encryption. We believe that the ability to view the actual ballot received by the election official provides transparency and easy-to-understand assurance to voters.

OmniBallot EBR (w/Verifier)	ElectionGuard for EBR
The Verifier uses Object Lock, which is fully supported by Amazon (AWS), using the same federally approved security protections used by the National Security Agency, CIA, DoD and other AWS customers.	ElectionGuard is not endorsed nor recommended for Electronic Ballot Return by Microsoft (the developer of the technology)
Voters are able to easily view and confirm their ballot was accurately received by the elections office: 	Voters view a long number and must trust that number means the ballot was securely transmitted: Example of ballot verification number: 04 CE D7 61 49 49 FD 4B 35 8B 1B 86 BC A3 C5 BC D8 20 6E 31 17 2D 92 8ª B7 34 F4 DB 11 70 4E 49 16 61 FC AE FA 7F BA 6F 0C 05 53 74 C6 79 7F 81 12 8ª F7 E2 5E 6C F5 FA 10 69 6B 67 D9 D5 96 51 B0
Voters can easily understand and transparently **confirm their ballot was accurately received.**	Voter cannot tell if their ballot has been compromised.
Deployed in West Virginia	Untested in West Virginia
Verifier with OmniBallot is easy to train, set-up and explain. *"Simply enter the verification code to view and confirm your ballot."*	Encrypted verification keys and methodologies are historically difficult to understand and explain to staff, media and voters.

4.1.2.6.2 "Microsoft has never endorsed the use of ElectionGuard for Internet Voting (including electronic ballot return)…." – Microsoft Corp

Quote cited at a presentation at Washington State legislature, October 27, 2023

4.1.2.7 The Vendor provides training and support to the Agency and counties during the duration of the contract.

Democracy Live Response: Democracy Live has a well-established, tested, and proven training, support and project planning process that ensures delivery of a turn-key fully deployed ADA and MOVE Act compliant Electronic Ballot Delivery and Return System to the State of West Virginia, meeting all stated deadlines.

Democracy Live offers individual one-on-one Online Training to all 55 West Virginia counties (outlined below). Democracy Live provides complete support of the OmniBallot system every step of the way. In addition to initial setup, our team provides proven training and QA tools so election administrators will be able to easily manage a county account, setup an election and create reports. Democracy Live through decades of practice and user groups has determined online zoom trainings are the most efficient for this implementation. Each zoom lasts approximately 15-25 minutes.

Democracy Live will schedule online training at the convenience of West Virginia's state and county official's schedules. Democracy Live suggest an initial training for the State prior to scheduling the first set of online training sessions with counties and election administrators. Online trainings will be conducted by the Democracy Live Project Manager and a Democracy Live Technical Accounts Manager.

Online trainings are done one-on-one with each County and their required election officials.
Each State and/or County designated representative will receive training materials that are customized to their county needs and requirements.

The initial Online Session includes:

- A review of the data files agreed upon to setup the OmniBallot system.
- Overview of timelines and election deployment dates.
- A one-on-one walkthrough training of the OmniBallot system to ensure confident election management.
- A review of the administrative tools in the OmniBallot System for Quality Assurance Testing.
- A review of the OmniBallot Electronic Ballot Return Portal for downloading and processing ballots.
- A review of the Voter Registration Manager System on how counties are able to import the voter registration file directly into the OmniBallot system (if applicable).
- Explanation of all available report modules.
- Review of key personal contact information for questions and assistance during setup.

After each Online Training the County will be provided with initial user guides for the OmniBallot Balloting Solution.

4.1.2.8 Section 508 Compliance

Section 508 of the Rehabilitation Act, as amended by the Workforce Investment Act of 1998 (P.L. 105-220) requires that when Federal agencies develop, procure, maintain, or use Information and Communication Technology (ICT), it shall be accessible to people living with disabilities. Federal employees and members of the public who have disabilities must have access to, and use of, information and data that is comparable to people without disabilities.

Products, platforms and services delivered as part of this work statement that are ICT, or contain ICT, must conform to the Revised 508 Standards, which are located at 36 C.F.R. § 1194.1 & Apps. A, C & D, and available at https://www.access-board.gov/guidelines-

andstandards/communications-and-it/about-the-ict-refresh/final-rule/text-of-the- standards and-guidelines

Democracy Live Response: Democracy Live deployed the nation's first WCAG, Section 508 compliant remote accessible absentee ballot in 2008. Since that launch and through our engagement with leading disability groups and organizations, Democracy Live continues to ensure that OmniBallot meets the highest levels of accessibility. The OmniBallot Portal conforms to the revised Section 508 and WCAG 2.1AA (and where applicable AAA).

Harvard Law School Project on Disabilities

"The work of Democracy Live demonstrates that full compliance with international standards on accessibility is readily achievable in the elections and voting space. Voters around the world have the right to equal access to participating in the democratic process and Democracy Live has gone a long way to producing technologies that helps achieve that goal."

Janet Lord
Harvard Law School Project on Disability

4.1.2.8.1 Provide list of item(s) that contains ICT. For each item, the following requirements apply:

Democracy Live Response: The OmniBallot Balloting Portal contains ICT.

4.1.2.8.2 All functional performance criteria apply when using an alternative design or technology that achieves substantially equivalent or greater accessibility and usability by individuals with disabilities, than would be provided by conformance to one or more of the requirements in Chapters 4-6 of the Revised 508 Standards, or when Chapters 4-6 do not address one or more functions of ICT.

Democracy Live Response: OmniBallot has been developed in collaboration with the University of Washington Center for Technology and Disabilities to meet WCAG 2.1 AA accessibility guidelines. As a supporter and sponsor of the National Federation of the Blind and the American Council of the Blind, Democracy Live engages with dozens of members of the disability community, representing many of the major disability advocacy organizations.

4.1.2.8.3 Software features and components: All WCAG Level AA Success Criteria, 502 Interoperability with Assistive Technology, 503 Application.

Democracy Live Response: First deployed in 2008, OmniBallot was the first remote balloting technology in U.S. elections to conform to Section 508 and WCAG accessibility criteria. OmniBallot has been developed, tested and proven to meet Section 508 compliance, most recently by the University of Washington Center for Technologies and Disabilities (October, 2023)

4.1.2.8.4 Hardware features and components: All requirements apply.

Democracy Live Response: N/A

4.1.2.8.5 Applicable support services and documentation: All requirements apply.

Democracy Live Response: Democracy Live engages voters living with a wide array of disabilities to test and ensure the entire system, including instructions are fully accessible in the portal. Democracy Live also offers educational videos, education and outreach materials and disability consultants to demonstrate

OmniBallot using assistive technologies.

4.1.2.8.2 Provide an Accessibility Conformance Report (ACR) for each commercially available ICT item offered through this contract. Create the ACR using the Voluntary Product Accessibility Template Version 2.1 or later, located at https://www.itic.org/policy/accessibility/vpat. Complete each ACR in accordance with the instructions provided in the VPAT template. Each ACR must address the applicable Section 508 requirements referenced in the Work Statement. Each ACR shall state exactly how the ICT meets the applicable standards in the remarks/explanations column, or through additional narrative. All "Not Applicable" (N/ A) responses must be explained in the remarks/explanations column or through additional narrative. Address each standard individually and with specificity, and clarify whether conformance is achieved throughout the entire ICT Item (for example – user functionality, administrator functionality, and reporting), or only in limited areas of the ICT Item.

Democracy Live Response: Appendix 1 addresses the Section 508 requirements and Appendix 2, the University of Washington Center on Technology and Disabilities report.

The Center on Technology and Disabilities tested OmniBallot for WCAG and Section 508 compliance summary:

University of Washington Center on Technology and Disabilities Executive Summary on OmniBallot
On October 30 and 31, 2023, ADII's human evaluation team reviewed the OmniBallot by Democracy Live and concluded that it met user acceptance and Section 508 conformity for voters living with disabilities. The review was conducted by individuals who use assistive technology for everyday computing tasks.

See Appendix 1 – OmniBallot Accessibility Conformance Report (based on VPAT 2.4 Rev)
See Appendix 2 – University of Washington Accessibility Test Report

4.1.2.8.3 Provide a description of the evaluation methods used to support Section 508 conformance claims. The Agency reserves the right, prior to making an award decision, to perform testing on some or all of the Vendor's proposed ICT items to validate Section 508 conformance claims made in the ACR.

Democracy Live Response: Democracy Live has continuous evaluations of OmniBallot by voters with disabilities to ensure the system meets Section 508 and WCAG 2.1aa. The system has been tested by the University of Washington Center for Technology and Disabilities, the State of California, the State of Michigan, and the State of North Carolina for accessibility. The Conformance report and VPAT are found in Appendix 1.

See Appendix 1 – OmniBallot Accessibility Conformance Report (based on VPAT2.4 Rev)
See Appendix 2 – University of Washington Accessibility Test Report

4.1.2.8.4 Describe your approach to incorporating universal design principles toensure ICT products or services are designed to support disabled users.

Democracy Live Response: The Democracy Live approach to accessible development and design is founded in our experience working in direct collaboration with local and national disability organizations and individual members of the disability community over the last 15 years. The first accessible absentee technology ever developed and deployed in the U.S. began with our lead

developer (and current Democracy Live CTO) developing side-by-side in the home of a representative of the American Council of the Blind.

Ever since that in-home development session in 2008, each version of our accessible remote balloting technology has been developed in cooperation and collaboration with actual voters with disabilities.

Winner of the Accessibility in Voting Award, presented at the United Nations and sponsor to the Federation of the Blind and the American Council of the Blind, Democracy Live has over 15 years of ongoing testing from voters representing a wide range of disabilities.

Incorporating universal design principals, Democracy Live pioneered the nation's first accessible absentee system. Due to our nearly two decades of work in the field of accessible balloting, Democracy Live is the only provider to have been awarded accessibility funding by both the EAC and the federal Department of Health and Human Services (HHS). Democracy Live was awarded the Accessibility in Voting Award, presented at the United Nations in 2019 for our work in the field of accessible voting.

Of current providers, over 80% of all accessible remote balloting implementations have been through Democracy Live since 2008. Through our continuous testing, across 5,000 elections, OmniBallot is the most deployed accessible remote balloting technology in the nation.

The system is proven to be fully accessible and compliant across all major accessible balloting technologies, including but not limited to:

✓ Screen reader
✓ Sip-and-puff
✓ Audio-tactile input devices
✓ Screen contrast
✓ Screen magnifiers
✓ Jelly button and foot pedals
✓ Accessible keyboard
✓ Other switch technologies

4.1.2.8.5 Describe plans for features that do not fully conform to the Section 508 Standards.

Democracy Live Response: All features of OmniBallot fully conform and are compliant with Section 508.

4.1.2.8.6 Describe "typical" user scenarios and tasks, including individuals with disabilities, to ensure fair and accurate accessibility testing of the ICT product or service being offered.

Democracy Live Response: In 2023, Democracy Live presented to hundreds of voters living with disabilities across multiple statewide disability conferences to listen, learn, and demonstrate our accessible remote balloting technologies. These ongoing relationships help Democracy Live developers ensure OmniBallot is at the leading edge of accessible balloting technologies.

Use case scenario #1:
A voter who has mobility challenges due to severe Parkinson's is able to use their home computer with their own assistive sip and puff technology to access the OmniBallot portal via a link on their browser connected device. Using their personal assistive input technology, the voter is able to access, mark, review and submit their ballot independently and privately.

Use case scenario #2:
A voter who is blind is able to use their home computer with their screen reader technology to access the OmniBallot portal via a link sent to them by their local election office. Using their screen reader, the voter is able to access, mark, review and submit their ballot independently and privately.

Democracy Live is confident that any voter with disabilities who can access and navigate Facebook, Amazon (or any website) can access OmniBallot. Voters using any combination of screen readers, browsers, operating systems and devices, will be able to access, mark and return their ballot independently and privately using OmniBallot. There are over 90 combinations of browsers, operating systems and screen readers that voters typically use. OmniBallot works with every combination.

"I wanted to share that I was fortunate and very pleased to be able to take advantage of the new accessible absentee ballot process. It was through OmniBallot with Democracy Live and it was so totally seamless. It was easy to do, all races appeared on the same page and had check boxes to check or uncheck for my preferences. I was provided a link to the voting platform, I answered a couple questions including which city I vote in, and then I was prompted to put in my PIN number. After that, it took me straight to the ballot for selecting my choices. At the end, it allowed me to review my choices, and then there was a submission button. It also indicated that my submission was successful. My electronic ballot went to my local city election official.

It couldn't have been easier, and probably took no more than ten minutes. It was terrific!"

Kim Charlson
Council of the Blind

4.1.2.9 The tool meets all mandatory security or control requirements as indicated in Attachments D and E.

Democracy Live Response: Democracy Live meets the mandatory security and controls required as described in Attachments D and E.

4.1.2.10 Prior to acceptance, the Agency reserves the right to perform testing on required ICT items to validate the Vendor's Section 508 conformance claims. If the Agency determines that Section 508 conformance claims provided by the Vendor represent a higher level of conformance than what is actually provided to the agency, the government shall, at its option, require the Vendor to remediate the item to align with the Vendor's original Section 508 conformance claims prior to acceptance.

Democracy Live Response: Democracy Live agrees to this requirement.

4.1.3 Establish Cyber Security Systems and Controls

4.1.3.1 Cybersecurity systems and controls are essential to distinguish, counteract, or decrease security risks. These measures are required to manage threats targeting computer systems and networks. These measures must be adaptive and robust. To determine whether your cyber security systems and controls meet requirements:

4.1.3.1.1 Please complete Attachment B – OWASP Application Level Security Verification Levels I -3. For all listed requirements please provide 2 pieces of supporting documentation. For any requirements that are lacking

supporting documentation, please add to Attachment F – POA&M Tracker.

Democracy Live Response: See Attachment B – OWASP Application Level Security Verification Levels I -3

4.1.3.1.2 Please complete Attachment C – OWASP Mobile Application Level Security Verification if applicable. For all listed requirements please provide 2 pieces ofsupporting documentation. For any requirements that are lacking supporting documentation, please add to Attachment F – POA&M Tracker. If not applicable, please put *N/A* by all requirements.

Democracy Live Response: N/A

4.1.3.1.3 Please complete Attachment D – Security Requirements for Databases. For all listed requirements please provide 2 pieces of supporting documentation. For any requirements that are lacking supporting documentation, please add to Attachment F – POA&M Tracker. All requirements flagged as "Mandatory" must be met for award eligibility.

Democracy Live Response: See Attachment D – Security Requirements for Databases.

4.1.3.1.4 Please complete Attachment E – Select Controls from NIST SP 800-171 in the provided format. For all listed requirements please provide 2 pieces of supporting documentation. For any requirements that are lacking supporting documentation, please add toAttachment F – POA&M Tracker. All requirements flagged as "Mandatory" must be met for award eligibility.

Democracy Live Response: See Attachment E – Select Controls from the StateRAMP Moderate Baseline

In addition, please see **Appendix 3 – Disaster Recovery Plan (aka Emergency Response Plan)**

4.1.3.1.5 Please complete Attachment F – POA&M Tracker in the provided format as indicated above for any requirements that are lacking supporting documentation.

Democracy Live Response: See Attachment F – POA&M Tracker

4.1.3.2 "Supporting documentation" includes but is not limited to the following types of documents:

Democracy Live Response: OmniBallot is the most independently tested elections technology in the U.S. elections market. Drawing from a pool of over 1,500 independent cybersecurity researchers, OmniBallot is continuously tested by a team of fully vetted, experienced independent cybersecurity researchers. This level of continuous independent, ongoing testing by hundreds of qualified researchers is unique and possibly

unmatched in U.S. elections.

Democracy Live has engaged Synack Cybersecurity to undertake OmniBallot's ongoing, continuous testing. **See Appendix 4 – Synack 2023 Summary Report**

Additionally, Democracy Live is the only provider in U.S. elections that meets all three of the following:

1) Full Soc 2 Type 2 certification.
2) Tested by an EAC-approved lab (SLI) for full compliance of the CIS Standards for Non-Tabulation Elections Systems.
3) Certified by every State requiring certification (for electronic ballot delivery).

Democracy Live partners with Amazon AWS, the largest secure cloud provider in the U.S. AWS has been selected by the National Security Agency, approved by the Department of Defense, Central Intelligence Agency, Federal Bureau of Investigation and Department of Homeland Security to secure critical documents (such as ballots).

Amazon and Democracy Live team together to offer West Virginia the most tested and deployed, secure platform in the U.S. Remote electronic balloting requires a highly secure, stable and scalable cloud environment. In over 5,000 deployments over 15 years, the Democracy Live OmniBallot remote balloting system has never been compromised.

Additionally, OmniBallot is independently monitored 24/7/365 by an independent cybersecurity firm, Alert Logic. Alert Logic is the winner of the 2023 Excellence in Cybersecurity Award. All activity and events within the portal are tracked and logged. The independent monitoring reports and activity logs from Alert Logic are made available to every Democracy Live customer.

Through our 2023 SOC 2 Type II audit and certification, Democracy Live has implemented controls for Controlled Unclassified Information (CUI) exceeding NIST 800-171. Through our SOC 2 certification, Democracy Live meets the NIST 800-53 CSF Medium Impact controls to comply with the more stringent FISMA standards.

Democracy Live implements and maintains system security controls to protect any and all sensitive data from unauthorized access and use. Least privileged access is practiced with all system services and data along with independent monitoring and execution on our established Incident Response plan.

Security Supporting Documentation

See Appendix 5 – StateRAMP Requirements
See Appendix 4 – Synack 2023 Summary Report
See Appendix 6 – Diagram: Auditable Proof of Secure Ballot Transmission

Through AWS, OmniBallot ballots are secured to meets the highest security, privacy and accessibility certification standards, including but not limited to:

- ISO/IEC 27001 Security Management Controls Standard
- FedRAMP Government Data Standards
- ISO/IEC 27018 Personal Data Protection Standard
- NIST 800-53 Security Control Standards

Democracy Live's comprehensive approach to security is based on a foundation of NIST federal and state cybersecurity best practices, proven compliance with the CIS Guidelines for Non-Tabulation Elections

Technologies, SOC 2 Type II requirements, independent ongoing testing and monitoring and compliance with AWS Security Well Architected standards.

Democracy Live leadership has decades of experience in elections security and state and federal cybersecurity. The Chief Security Officer for Democracy Live is the former CISO for the State of Colorado and Denver International Airport. Democracy Live top leadership have been awarded Security Level clearance. Respectively, our CEO and VP of Government Relations are the former Vice-Chair and current Vice-Chair of the CISA-supported Sector Coordinating Council. A key consultant to Democracy Live is former 4-Star General Robert (Abe) Abrams who bring unmatched experience in national security to the Democracy Live team.

Democracy Live works closely with the AWS security and Artificial Intelligence teams to ensure OmniBallot meets all federal and state cybersecurity best practices. This continuous approach includes identifying assets and risks, mitigating vulnerabilities, measuring effectiveness, and implementing improvements to the process.

Beyond SOC 2 Type II certification, the OmniBallot web application is compliant with the NIST FIPS 200: Security Requirements for Federal Information and Information Systems standard. Additionally, OmniBallot has been reviewed by an EAC-approved lab (SLI) to comply with the Center for Internet Security Guidelines for Non-Tabulation Elections Technologies.

4.1.3.3 For all responses provided for Attachments B – F that are sensitive, confidential, proprietary, or indicative of a material or critical elements of the technology that are not intended or appropriate for public review or information for security or other valid reasons, indicate "CONFIDENTIAL" in an associated logical location for the specific requirement in the Attachment. For all such material or crucial elements not disclosed in any Attachments, Vendor shall provide their response for each requirement marked "CONFIDENTIAL" in writing directly to the agency Chief Information Officer, David Tackett, after the close of bidding period through either Email (DTackett@wvsos.gov) or U.S. Mail to Secretary of State's Office, State Capitol, 1900 Kanawha Blvd, E., Bldg. 1, Ste. 157-K, Charleston, WV 25305, but by no later than 7 days thereafter.

4.2. Qualifications and Experience: Vendor should provide information and documentation regardingits qualifications and experience in providing services or solving problems similar in size, scope and complexity to those requested in this RFP. Information and documentation should include, but are not limited to, proposed staffing plans, descriptions of past projects completed (descriptions shouldinclude the location of the project, project manager name and contact information, type of project, and what the project goals and objectives where and how they were met.), references for prior projects including the value and period of performance of past projects, and any other information that Vendor deems relevant to the items identified as desirable or mandatory below.

Democracy Live Response: Democracy Live has deployed our accessible remote balloting portal in 34 States, covering over 2,500 jurisdictions. The Democracy Live electronic ballot *return* option has been deployed in 20% of all States in the U.S., including West Virginia. Only Democracy Live has deployed electronic ballot return in multiple statewide implementations, including Presidential elections.

Specific to remote, electronic balloting Democracy Live offers the State of West Virginia the most experienced customer support and implementation team in the U.S. The key to a successful implementation is ensuring the State has the support and assistance they need during key election periods and thereafter. Democracy Live assigns the State a designated Project Manager through the entirety of the Project.

The State and counties will have access to the Project Manager's email address and mobile phone, available 24 hours a day, 7 days a week. The State will also have assigned two designated election configuration specialists to assist with configuring the election and providing internal quality assurance testing.

A designated developer and the Democracy Live CTO are available to assist the State with feature requests, customization and technical support throughout the election cycle. At the request of the State, Democracy Live offers weekly check-in calls with the State to check-in on any questions that may have been discovered and any changes that may be required during implementation as well as post-go-live requirements.

A few statewide customer examples are:
State of North Carolina – OmniBallot Balloting Portal (2-way ballot delivery/return)
Value: $245,000 over 1 year, $655,000 over 3 years. Includes accessible ballot request portal. The goal of the NC OmniBallot Portal Implementation was to create a customized all-in-one Balloting Request and Ballot Delivery/return portal for military and overseas voters and voters with disabilities in the State of North Carolina. Democracy Live worked with the State to implement an integrated system with the North Carolina Seams system to allow voters to request an absentee ballot or complete there FPCA Application using the OmniBallot Ballot Request Portal. Qualified voters then had the ability to access, mark and return their issued ballot using the OmniBallot Balloting Portal. Democracy Live delivered an on-time custom system for the State of North Carolina. The North Carolina OmniBallot Portal has been in use since 2020. The system was required to be easily setup and implemented across all 100 counties.

State of North Carolina Project Manager: Lisa Berot Lisa.Berot@ncsbe.gov, (919) 441-0108
Karen Bell, Director of Elections.
(919) 814-0700
karen.bell@ncsbe.gov

Commonwealth of Pennsylvania - OmniBallot Balloting Portal (1-way Ballot delivery)
$530,000 over 1 year. The goal of the Commonwealth of PA Balloting Portal was to implement a one-way client-side ballot marking system for voters with disabilities to access, mark and print their ballot privately and independently. The system required to be easily implemented by all 67 counties. Democracy Live worked with the State to delivery and implement an on-time deployment of the Accessible Balloting Portal. The system has been in use since 2020.

Commonwealth of PA Project Manager, Sindhu Ramachandran,
717-216-9877
sramachand@pa.gov

State of Alabama – OmniBallot Balloting Portal (2-way ballot delivery/return)
Value: $220,000 over one year. The goal of this implementation was to create a custom statewide electronic ballot delivery and return system for military and overseas voters to access, mark and if qualified, return their ballot electronically using the OmniBallot Portal. Democracy Live worked with the State to create a custom workflow and Balloting Portal to meet the requirements of the State of Alabama's special rank choice voting rules and regulations. The system was deployed on-time with no issues reported.

Clay Helms, Chief of Staff
Office of Secretary of State Wes Allen
334-242-7207
Clay.Helms@sos.alabama.gov

State of Michigan – OmniBallot Balloting Portal (1-way ballot delivery) and ballot request portal. Value: $1.5m over 5 years. The goal of this implementation was to create an accessible absentee request portal system integrated with the State of Michigan's voter portal to allow voters with disabilities to request an

accessible ballot privately and independently and to allow approved voters to have the ability through the State of Michigan's voter portal to access, mark and print their ballot privately and independently. The system has been in use since 2020.

State Project Manager: Shelly Belton
Elections Operations Manager, Michigan Department of State – Bureau of Elections
Cell: 517-281-5085
BeltonS@Michigan.gov

State of Colorado: OmniBallot Balloting Portal (1-way Ballot Delivery)
$194,000 over 1 year (Self-Administered). The goal of the State of Colorado implementation was to create a customer system that can be self-administered by the State and each of the 64 counties to have the ability to configure and import their election data to create a remote balloting portal available to military and overseas voters, voters living with disabilities, and emergency voters. The system has been in use since 2019.

State of Colorado Project Manager: Caleb Thornton
Legal, Policy, and Rulemaking Manager - Elections
303.894.2200 x 6386
caleb.thornton@coloradosos.gov

Personnel Qualifications:

The Democracy Live Operations and Support Team

Felicia Erlich, Esq. – Implementation/Project Manager
Chief Operation Officer and Corporate Counsel
felicia@democracylive.com
Felicia has led the Operations and support team to successfully deploy OmniBallot in over 1,200 elections across 30 states. Felicia was the Project Lead for the first implementation of Electronic Ballot Return in a Presidential election. Felicia is a proven leader in meeting the support needs of customers and company stakeholders. She is known to ensure deadlines are met, communication is clear, and issues are resolved in a timely manner. Felicia is a member of the California, New York, Washington and California Bars and is an active member of the Department of Homeland Security sponsored Elections Sector Emergency Response Task Force.

Island Pinnick – Developer Lead
Democracy Live Chief Technology Officer and Lead Architect of the OmniBallot Balloting Portal
island@democracylive.com
Democracy Live's Chief Technology Officer and lead developer of OmniBallot, Island has over 15 years of experience in the elections industry. Island developed the first and most widely deployed remote electronic balloting technology in the U.S. Island was awarded the Bill & Mary Gates Scholar of the Year at the University of Washington and a graduate in nano-technology engineering,

Seth Kulakow – Chief Security Officer
seth@democracylive.com
Seth is the former CISO at the State of Colorado, Denver International Airport where he assessed risk, crisis, business continuity, and IT management solutions. Seth is a nationally recognized cybersecurity expert, championing global transformative information technology, security, and risk programs. With Security Clearance, Seth was recognized by the US Secret Service for Cyber Security Program Excellence. Seth is a former guest member of the Aspen Institute's Cyber Security Group, former DHS advisor, and speaker at numerous prominent cybersecurity, risk, and IT industry conferences.

Lori Augino-VP Government Relations
lori@democracylive.com
As the former Director of Elections for the State of Washington, President of the National Association of State Election Directors, a local election official in Pierce County, WA, and Executive Director for the National Vote at Home Institute, Lori has over 27 years in elections administration. Lori is the Vice-Chair of the DHS-sponsored Elections Infrastructure Sector Coordinating Council (SCC) and Chair of the SCC Emergency Response Elections Task Force.

Michael Hamilton, MS – Security Analyst
Michael has served as a Cybersecurity Policy Advisor for Washington State, Vice-Chair of the State, Local, Tribal, and Territorial Government Coordinating Council (SLTTGCC), Chief Information Security Officer for the City of Seattle, and Managing Consultant for VeriSign Global Security Consulting.

Nick Toomey – Technical Support and Data Coordinator
nick@democracylive.com
Democracy Live's Technical Accounts Manager has experience supporting the implementations of over 1,500 electronic balloting implementations, across 20 states in the United States.

James Johnston – Network and Development Coordinator
james@democracylive.com
Former U.S. Military, with a degree in computer programming and security from the University of Washington, James has deep level experience in secure development for voting and elections technologies.

General Abrams (Military Voting Advisor):
General Abrams earned his commission from the United States Military Academy in 1982. General Abrams commanded at every level from Company thru Four Star Major Command, and led units in combat operations in Saudi Arabia, Kuwait, Iraq and Afghanistan.

As a four-star General for over 6 years, General Abrams served as the 22d Commander of United States Army Forces Command from Aug 2015-Oct 2018, the Army's largest formation comprised of over 229,000 active-duty Soldiers and provided training and readiness oversight of the US Army National Guard and US Army Reserve. In total the Forces Command team included 776,000 Soldiers and 96,000 Civilians. General Abrams final assignment was serving as the Commander United Nations Command, US-ROK Combined Forces Command, and United States Forces Korea from Nov 2018-July 2021.

4.2.1. Qualification and Experience Information: Vendor should describe in its proposal how it meets the desirable qualification and experience requirements listed below.

4.2.1.1. Vendor's tool has been reviewed by at least one (1) independent, nationally recognized organization supporting the Disability Community for its user acceptance and Section 508 conformity for voters living with disabilities. Copiesof any reports or public statements by the organization(s) should be provided tothe Agency for Confidential review.

Democracy Live Response: OmniBallot is the most deployed accessible remote balloting technology in the United States. Pioneering the first accessible absentee deployment in 2008, Democracy Live has implemented secure, accessible remote balloting in more elections and jurisdictions than all other providers combined. As the industry leader in accessible absentee balloting, OmniBallot has had more testing in live elections than any other technology in this category.

OmniBallot has been tested by the University of Washington Center for Technology and Disabilities and testing by dozens of members of leading disability groups, such as the American Council of the

Blind, National Federation of the Blind, Center for Independent Living and other organizations. Below are a few quotes regarding our accessible technology. **See Appendix 2 – University of Washington Accessibility Test Report**

"Democracy Live's system is a marvelous example of a universal design. It provides a valuable resource on voter information for all citizens, including those with disabilities who often lack access to the ballot."

Deborah Cook
University of Washington Center for Technology and Disability Studies

4.2.2. Mandatory Qualification/Experience Requirements - The following mandatory qualification/experience requirements must be met by the Vendor as a part of its submitted proposal. Vendor should describe how it meets the mandatory requirements and include any areas where it exceeds the mandatory requirements. Failure to comply with mandatory requirements will lead to disqualification, but areas where the mandatory requirements are exceeded will be included in technical scores where appropriate. The mandatory qualifications/experience requirements are listed below.

4.2.2.1. Implemented tool in at least two (2) previous federal elections. A list of all previous federal elections, including the jurisdiction, shall be provided to theAgency.

Democracy Live Response: The Democracy Live remote balloting portal has been deployed in eight federal elections – 2008, 2010, 2012, 2014, 2016, 2018, 2020, 2022. This does not include over 2,000 off-year elections where OmniBallot has been deployed. In over 5,000 elections since 2008, Democracy Live has a 99% on-time success rate.

In competitive bids, Democracy Live was selected and funded by the **U.S. Department of Defense** and the **U.S. State Department**. Additionally, the U.S. Department of Health and Human Services approved accessibility funding for jurisdictions to deploy Democracy Live accessible balloting technologies.

Democracy Live has been approved and implemented in jurisdictions in 34 states. Below is a sample of where Democracy Live customers have been deployed in at least one federal election cycle.

(Bold indicates electronic ballot return.)

- Michigan (Statewide)
- Pennsylvania (Statewide)
- **North Carolina** (Statewide)
- **Indiana** (Statewide)
- **New Jersey** (Statewide)
- Florida (Multiple counties)
- **Massachusetts (Statewide)**
- **South Carolina** (Statewide)
- Washington D.C.
- **West Virginia** (Statewide)
- New York City
- Rhode Island (Statewide)
- Vermont (Statewide)
- Ohio (Majority of voters)
- Colorado (Statewide)
- **Delaware** (Statewide)
- Minnesota (Statewide)
- Texas (Multiple Counties)
- California (Majority of counties)
- Washington State (Statewide)
- **Utah (Utah County)**
- **Oregon (Umatilla and Jackson County)**
- State of NH (OmniBallot Tablet Software)
- Nevada (Washoe County)
- State of Illinois

4.2.2.2. Vendor's applicable network and systems or tool have been assessed for security vulnerabilities by at least two (2) independent, federally recognized, certified, of industry specific equivalent technology or cybersecurity auditors. Copies of all assessments or equivalent reports completed in the last 3 years shall be provided to the Agency.

Democracy Live Response: OmniBallot from Democracy Live has had multiple security reviews and ongoing vulnerability testing conducted at the Federal, State, and independent lab levels.

- U.S. Department of Homeland Security – Application Penetration Testing
- Cybersecurity and Infrastructure & Security Agency (CISA) – Security & Vulnerability Test + Idaho National Labs – Code and application Review
- Synack cybersecurity testing – Ongoing vulnerability testing from a pool of over 1,500 independent cybersecurity researchers 365 days per year
- EAC-approved independent test lab (SLI) – Testing of OmniBallot to meet CIS Standards for Non-Tabulation Elections Technologies
- SOC 2 Type 2 – Full audit and certification
- Soteria Cybersecurity – Detailed cybersecurity review
- Amazon Web Services (AWS) – Security Architecture Review/Well Architected Review (WAR)
- Additionally, OmniBallot is the only remote electronic balloting technology to be officially certified in every state that requires certification (for one-way ballot transmission)

From Synack:
"Findings from a recent Synack security report showed over 400 independent researchers tested OmniBallot for vulnerabilities. The security report, available to customers through Democracy Live, found just one low-severity vulnerability in OmniBallot. That one low level vulnerability was later confirmed to be fixed and no longer present in OmniBallot. At Synack, we are working hard with partners like Democracy Live by making it harder for attackers to compromise any part of our election infrastructure. Synack researchers will continue to independently test the OmniBallot portal through the 2024 Presidential election and beyond."

Jay Kaplan
CEO & Co-Founder, SYNACK · The Premier Security Testing Platform

Optional Services Included with this Proposal

1) Mobile, ADA sample ballot. Specific to each voter
2) Digital, mobile ballot cure
3) Automated Ballot Duplication
4) Mobile-only voting app pilot:
As an optional 1-county pilot opportunity, Democracy Live has partnered with the nonprofit Free Democracy Foundation (led by Jocelyn Bucaro) to offer a mobile app system that also has end-to-end verifiable encryption. This mobile app pilot offers the State an option to test a mobile-app with end-to-end encryption verification via a 1-county pilot, while still implementing OmniBallot statewide. Like OmniBallot, the mobile app pilot still offers voters the ability to verify independently that their ballot is recorded and encrypted correctly prior to submitting and verify it is received and counted correctly once cast. Democracy Live is pleased to offer this mobile app pilot opportunity via the Free Democracy Foundation as an add-on option to the State.

Summary

Since 2020, it has been an honor to support the Secretary of State and the voters of West Virginia on this important ballot access initiative. We hope this proposal offers the Secretary of State's office confidence that Democracy Live has the proven, demonstrated experience supporting West Virginia's unique requirements for both security, accessibility and end to end ballot verification. Having been honored to support the office of Secretary of State on this project, Democracy Live understands the state's high expectations and requirements for this project.

A key goal of this proposal is to ensure the state has the most secure, auditable, and verifiable remote balloting technology in the U.S. To that end, Democracy Live is the only electronic ballot return provider that has been audited for SOC 2 Type 2 certification, approved by an EAC-approved lab for compliance with CIS Non-Tabulation Elections Technologies, and undergoes ongoing vulnerability testing 365 days a year. Democracy Live is the only provider that offers a proven, deployed tool that allows voters to easily and transparently verify the ballot *submitted*, was the ballot *received* (via the Verifier).

As the only as provider with a proven history of delivering electronic ballot return technology in multiple statewide and Presidential elections, including West Virginia, we sincerely appreciate the opportunity to respond to this important RFP.

(End of 20 Page Technical Response Limit)

SECTION 5: VENDOR PROPOSAL

5.1. **Economy of Preparation:** Proposals should be prepared simply and economically providing a concise description of the items requested in Section 4. Emphasis should be placed on completeness and clarity of the content.

5.2. **Incurring Cost:** Neither the State nor any of its employees or officers shall be held liable for any expenses incurred by any Vendor responding to this RFP, including but not limited to preparation, delivery, or travel.

5.3. **Proposal Format:** Vendors should provide responses in the format listed below:

5.3.1. **Two-Part Submission:** Vendors must submit proposals in two distinct parts separate from each other: technical and cost. Technical proposals must not contain any cost information relating to the project. Cost proposal must contain all cost information and must be sealed in a separate envelope from the technical proposal to facilitate a secondary cost proposal opening.

5.3.2. **Title Page:** State the RFP subject, number, Vendor's name, business address, telephone number, fax number, name of contact person, e-mail address, and Vendor signature anddate. A title page shall be used for both the technical and cost proposals and will not be included in the page count.

5.3.3. **Table of Contents:** Clearly identify the material by section and page number for both the technical and cost volume. The table of contents will not be included in the page count.

5.3.4. **Response Reference:** Vendor's response should clearly reference how the informationprovided applies to the RFP request. For example, listing the RFP number and restating the RFP request as a header in the proposal would be considered a clear reference.

Proposal Submission: All proposals (both technical and cost) must be submitted to the Purchasing Division **prior** to the date and time listed in Section 2, Instructions to Vendors Submitting Bids as the bid opening date and time. The technical proposal shall not exceed 20 pages. The cost volume does not have a page limitation.

Late proposal submissions may or may not be considered for evaluation.

SECTION 6: EVALUATION AND AWARD

6.1. Evaluation Process: Proposals will be evaluated in two parts by a committee of three (3) or more individuals. The first evaluation will be of the technical proposal and the second is an evaluation ofthe cost proposal. The Vendor who demonstrates that it meets all of the mandatory specifications required and represents best overall value, shall be awarded the contract.

6.2. Evaluation Criteria: Proposals will be evaluated based on criteria set forth in the solicitation and information contained in the proposals submitted in response to the solicitation. The technical evaluation will be based upon the point allocations designated below for a total of 70 of the I 00 points. Cost represents 30 of the 100 total points.

Evaluation Point Allocation:

Project Goals and Proposed Approach (§ 4.1)

- Approach & Methodology to Goals/Objectives **(§ 4.1.2., § 4.1.2.5)** **(15) Points Possible**

The extent to which the vendor demonstrates a convincing approach to achieving the goals and objectives described in this RFP.

- Approach & Methodology to Compliance with Mandatory Project Requirements **(§ 4.1.3)** **(25) Points Possible**

The extent to which the vendor demonstrates a convincing approach to complying with the project requirements described in this RFP.

Qualifications and Experience (§ 4.2)

- Mandatory Qualifications/Experience Requirements **(§ 4.2.2)** **(30) Points Possible**

The extent of the vendor's qualifications and experience indicates the likelihood of success in carrying out the services described in this RFP.

Total Technical Score:	70 Points Possible
Total Cost Score:	30 Points Possible

Total Proposal Score: 100 Points Possible

6.3. Technical Bid Opening: At the technical bid opening, the Purchasing Division will open and announce the technical proposals received prior to the bid opening deadline. Once opened, the technical proposals will be provided to the Agency evaluation committee for technical evaluation.

6.4. Technical Evaluation: The Agency evaluation committee will review the technical proposals, assign points where appropriate, and make a final written recommendation to the Purchasing Division.

6.5. Proposal Disqualification:

6.5.1. **Minimum Acceptable Score ("MAS"):** Vendors must score a minimum of 70% (49 points) of the total technical points possible in order to move past the technical evaluation and have their cost proposal evaluated. All vendor proposals not attaining the MAS will be disqualified.

6.5.2. **Failure to Meet Mandatory Requirement:** Vendors must meet or exceed all mandatory requirements in order to move past the technical evaluation and have their cost proposals evaluated. Proposals failing to meet one or more mandatory requirements of the RFP will bedisqualified.

6.6. Cost Bid Opening: The Purchasing Division will schedule a date and time to publicly open and announce cost proposals after technical evaluation has been completed and the Purchasing Division has approved the technical recommendation of the evaluation committee. All cost bids received will be opened. Cost bids for disqualified proposals will be opened for record keeping purposes only andwill not be evaluated or considered. Once opened, the cost proposals will be provided to the Agencyevaluation committee for cost evaluation.

The Purchasing Division reserves the right to disqualify a proposal based upon deficiencies in thetechnical proposal even after the cost evaluation.

6.7. Cost Evaluation: The Agency evaluation committee will review the cost proposals, assign points in accordance with the cost evaluation formula contained herein and make a final recommendation to the Purchasing Division.

Cost Evaluation Formula: Each cost proposal will have points assigned using the following formula for all Vendors not disqualified during the technical evaluation. The lowest cost of all proposals is divided by the cost of the proposal being evaluated to generate a cost score percentage.That percentage is then multiplied by the points attributable to the cost proposal to determine the number of points allocated to the cost proposal being evaluated.

Step 1: Lowest Cost of All Proposals / Cost of Proposal Being Evaluated = Cost Score Percentage

Step 2: Cost Score Percentage X Points Allocated to Cost Proposal= **Total Cost Score**

Example:

Proposal 1 Cost is $1,000,000
Proposal 2 Cost is $1,100,000
Points Allocated to Cost Proposal is 30

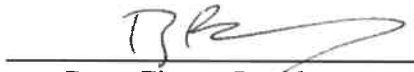
Proposal 1: Step 1 - $1,000,000 / $1,000,000 = Cost Score Percentage of 1 (100%)
Step 2 - 1 X 30 = Total Cost Score of 30

Proposal 2: Step 1-$1,000,000 / $1,100,000 = Cost Score Percentage of 0.909091 (90.9091%)
Step 2- 0.909091 X 30 = Total Cost Score of 27.27273

6.8. Availability of Information: Proposal submissions become public and are available for review immediately after opening pursuant to West Virginia Code §SA-3-1 l(h). All other information associated with the RFP, including but not limited to, technical scores and reasons for disqualification, will not be available until after the contract has been awarded pursuant to West Virginia Code of State Rules §148-1-6.3.d.

By signing below, I certify that I have reviewed this Request for Proposal in its entirety; understand the requirements, terms and conditions, and other information contained herein; that I am submitting this proposal for review and consideration; that I am authorized by the bidder to execute this bid or any documents related thereto on bidder's behalf; that I am authorized to bind the bidder in a contractual relationship; and that, to the best of my knowledge, the bidder has properly registered with any State agency that may require registration.

Democracy Live, Inc.
(Company)



Bryan Finney, President

Contact Phone: (206) 465-5636
Date: 11-06-2023

REQUEST FOR PROPOSAL

West Virginia Secretary of State
CRFP SOS2200000001
Election E-Ballot Delivery Technology

Attachment A: Addendum Number 1 and Acknowledgement

Attachment B -OWASP Application Level Security Verification Levels 1-3

Attachment C - OWASP Mobile Application Level Security Verification if applicable (Democracy Live, N/A)

Attachment D - Security Requirements for Databases

Attachment E - Select Controls from NIST SP 800-171

Attachment F - POA&M Tracker

Attachment A

Addendum Number 1 and 2 Acknowledgement

SOLICITATION NUMBER: CRFP SOS2400000001
Addendum Number: 1

The purpose of this addendum is to modify the solicitation identified as ("Solicitation") to reflect the change(s) identified and described below.

Applicable Addendum Category:

[X] Modify bid opening date and time

[ ] Modify specifications of product or service being sought

[ ] Attachment of vendor questions and responses

[ ] Attachment of pre-bid sign-in sheet

[ ] Correction of error

[ ] Other

Description of Modification to Solicitation:

Addendum No 1 is issued for the following reasons:

1) To modify the bid opening date from 10/17/23 to 11/08/23.

--no other changes--

Additional Documentation: Documentation related to this Addendum (if any) has been included herewith as Attachment A and is specifically incorporated herein by reference.

Terms and Conditions:

1. All provisions of the Solicitation and other addenda not modified herein shall remain in full force and effect.

2. Vendor should acknowledge receipt of all addenda issued for this Solicitation by completing an Addendum Acknowledgment, a copy of which is included herewith. Failure to acknowledge addenda may result in bid disqualification. The addendum acknowledgement should be submitted with the bid to expedite document processing.

Revised 6/8/2012

SOLICITATION NUMBER: CRFP SOS2400000001
Addendum Number: 2

The purpose of this addendum is to modify the solicitation identified as ("Solicitation") to reflect the change(s) identified and described below.

Applicable Addendum Category:

[ ] Modify bid opening date and time

[X] Modify specifications of product or service being sought

[X] Attachment of vendor questions and responses

[ ] Attachment of pre-bid sign-in sheet

[ ] Correction of error

[ ] Other

Description of Modification to Solicitation:

Addendum No 2 is issued for the following reasons:

1) To publish a copy of vendor questions with their responses.

2) To modify specifications. (4.1.3.3)

--no other changes--

Additional Documentation: Documentation related to this Addendum (if any) has been included herewith as Attachment A and is specifically incorporated herein by reference.

Terms and Conditions:

1. All provisions of the Solicitation and other addenda not modified herein shall remain in full force and effect.

2. Vendor should acknowledge receipt of all addenda issued for this Solicitation by completing an Addendum Acknowledgment, a copy of which is included herewith. Failure to acknowledge addenda may result in bid disqualification. The addendum acknowledgement should be submitted with the bid to expedite document processing.

Revised 6/8/2012

ADDENDUM ACKNOWLEDGEMENT FORM
SOLICITATION NO.: CRFP SOS24*001

Instructions: Please acknowledge receipt of all addenda issued with this solicitation by completing this addendum acknowledgment form. Check the box next to each addendum received and sign below. Failure to acknowledge addenda may result in bid disqualification.

Acknowledgment: I hereby acknowledge receipt of the following addenda and have made the necessary revisions to my proposal, plans and/or specification, etc.

Addendum Numbers Received:
(Check the box next to each addendum received)

[X]	Addendum No. 1	[ ]	Addendum No. 6
[X]	Addendum No. 2	[ ]	Addendum No. 7
[ ]	Addendum No. 3	[ ]	Addendum No. 8
[ ]	Addendum No. 4	[ ]	Addendum No. 9
[ ]	Addendum No. 5	[ ]	Addendum No. 10

I understand that failure to confirm the receipt of addenda may be cause for rejection of this bid. I further understand that any verbal representation made or assumed to be made during any oral discussion held between Vendor's representatives and any state personnel is not binding. Only the information issued in writing and added to the specifications by an official addendum is binding.

Democracy Live
Company

Authorized Signature

11/06/2023
Date

NOTE: This addendum acknowledgement should be submitted with the bid to expedite document processing.
Revised 6/8/2012

Attachment B

OWASP Application Level Security Verification Levels 1-3

Open Web Application Security Project - Application Security Verification Standard 4.0.2 Level 1	Points for Meeting Requirements	Status	Documentation
Password Security Requirements			2.1
Verify that user set passwords are at least 8 characters in length (after multiple spaces are combined).	5	Fully Implemented	2.1.1
Verify that passwords 64 characters or longer are permitted but may be no longer than 128 characters.	5	Fully Implemented	2.1.2
Verify that password truncation is not performed. However, consecutive multiple spaces may be replaced by a single space.	5	Fully Implemented	2.1.3
Verify that any printable Unicode character, including language neutral characters such as spaces and Emojis are permitted in passwords.	5	Fully Implemented	2.1.4
Verify users can change their password.	5	Fully Implemented	2.1.5
Verify that password change functionality requires the user's current and new password.	5	Fully Implemented	2.1.6
Verify that passwords submitted during account registration, login, and password change are checked against a set of breached passwords either locally (such as the top 1,000 or 10,000 most common passwords which match the system's password policy) or using an external API. If using an API a zero knowledge proof or other mechanism should be used to ensure that the plain text password is not sent or used in verifying the breach status of the password. If the password is breached, the application must require the user to set a new non-breached password.	5	Fully Implemented	2.1.7
Verify that a password strength meter is provided to help users set a stronger password.	5	Implemented	2.1.8
Verify that there are no password composition rules limiting the type of characters permitted. There should be no requirement for upper or lower case or numbers or special characters.	5	Implemented	2.1.9
Verify that there are no periodic credential rotation or password history requirements.	5	Fully Implemented	2.1.10
Verify that "paste" functionality, browser password helpers, and external password managers are permitted.	5	Fully Implemented	2.1.11
Verify that the user can choose to either temporarily view the entire masked password, or temporarily view the last typed character of the password on platforms that do not have this as built-in functionality.	5	Fully Implemented	2.1.12
General Authenticator Requirements			2.2
Verify that anti-automation controls are effective at mitigating breached credential testing, brute force, and account lockout attacks. Such controls include blocking the most common breached passwords, soft lockouts, rate limiting, CAPTCHA, ever increasing delays between attempts, IP address restrictions, or risk-based restrictions such as location, first login on a device, recent attempts to unlock the account, or similar. Verify that no more than 100 failed attempts per hour is possible on a single account.	5	Fully Implemented	2.2.1
Verify that the use of weak authenticators (such as SMS and email) is limited to secondary verification and transaction approval and not as a replacement for more secure authentication methods. Verify that stronger methods are offered before weak methods, users are aware of the risks, or that proper measures are in place to limit the risks of account compromise.	5	Fully Implemented	2.2.2
Verify that secure notifications are sent to users after updates to authentication details, such as credential resets, email or address changes, logging in from unknown or risky locations. The use of push notifications - rather than SMS or email - is preferred, but in the absence of push notifications, SMS or email is acceptable as long as no sensitive information is disclosed in the notification.	5	Fully Implemented	2.2.3
Authenticator Lifecycle Requirements			2.3

Verify system generated initial passwords or activation codes SHOULD be securely randomly generated, SHOULD be at least 6 characters long, and MAY contain letters and numbers, and expire after a short period of time. These initial secrets must not be permitted to become the long term password.	5	Implemented - Compensating Control	2.3.1
Credential Recovery Requirements			2.4
Verify that a system generated initial activation or recovery secret is not sent in clear text to the user.	5	Fully Implemented	2.4.1
Verify password hints or knowledge-based authentication (so-called "secret questions") are not present.	5	Fully Implemented	2.4.2
Verify password credential recovery does not reveal the current password in any way.	5	Fully Implemented	2.4.3
Verify shared or default accounts are not present (e.g. "root", "admin", or "sa").	5	Fully Implemented	2.4.4
Verify that if an authentication factor is changed or replaced, that the user is notified of this event.	5	Partially Implemented	2.4.5
Verify forgotten password, and other recovery paths use a secure recovery mechanism, such as time-based OTP (TOTP) or other soft token, mobile push, or another offline recovery mechanism. (C6)	5	Fully Implemented	2.4.6
Out of Band Verifier Requirements			2.5
Verify that clear text out of band (NIST "restricted") authenticators, such as SMS or PSTN, are not offered by default, and stronger alternatives such as push notifications are offered first.	5	Fully Implemented	2.5.1
Verify that the out of band verifier expires out of band authentication requests, codes, or tokens after 10 minutes.	5	Fully Implemented	2.5.2
Verify that the out of band verifier authentication requests, codes, or tokens are only usable once, and only for the original authentication request.	5	Fully Implemented	2.5.3
Verify that the out of band authenticator and verifier communicates over a secure independent channel.	5	Fully Implemented	2.5.4
Single or Multi-factor One Time Verifier Requirements			2.6
Verify that time-based OTPs have a defined lifetime before expiring.	5	Fully Implemented	2.6.1
Fundamental Session Management Requirements			2.7
Verify the application never reveals session tokens in URL parameters	5	Fully Implemented	2.7.1
Session Binding Requirements			2.8
Verify the application generates a new session token on user authentication.	5	Fully Implemented	2.8.1
Verify that session tokens possess at least 64 bits of entropy.	5	Fully Implemented	2.8.2
Verify the application only stores session tokens in the browser using secure methods such as appropriately secured cookies (see section 3.4) or HTML 5 session storage.	5	Partially Implemented - Compensating Control	2.8.3
Session Logout and Timeout Requirements			2.9
Verify that logout and expiration invalidate the session token, such that the back button or a downstream relying party does not resume an authenticated session, including across relying parties.	5	Fully Implemented	2.9.1
Cookie-Based Session Management	5	Fully Implemented	2.10
Verify that cookie-based session tokens have the 'Secure' attribute set.	5	Fully Implemented	2.10.1

Verify that cookie-based session tokens have the 'HttpOnly' attribute set.	5	Partially Implemented - Compensating Control	2.10.2
Verify that cookie-based session tokens utilize the 'SameSite' attribute to limit exposure to cross-site request forgery attacks.	5	Partially Implemented - Compensating Control	2.10.3
Verify that cookie-based session tokens use "__Host-" prefix (see references) to provide session cookie confidentiality.	5	Partially Implemented - Compensating Control	2.10.4
Verify that if the application is published under a domain name with other applications that set or use session cookies that might override or disclose the session cookies, set the path attribute in cookie-based session tokens using the most precise path possible.	5	Not Applicable	2.10.5
General Access Control Design			2.11
Verify that the application enforces access control rules on a trusted service layer, especially if client-side access control is present and could be bypassed.	5	Fully Implemented	2.11.1
Verify that all user and data attributes and policy information used by access controls cannot be manipulated by end users unless specifically authorized.	5	Fully Implemented	2.11.2
Verify that the principle of least privilege exists - users should only be able to access functions, data files, URLs, controllers, services, and other resources, for which they possess specific authorization. This implies protection against spoofing and elevation of privilege.	5	Fully Implemented	2.11.3
Verify that the principle of deny by default exists whereby new users/roles start with minimal or no permissions and users/roles do not receive access to new features until access is explicitly assigned.	5	Fully Implemented	2.11.4
Verify that access controls fail securely including when an exception occurs.	5	Fully Implemented	2.11.5
Operation Level Access Control			2.12
Verify that sensitive data and APIs are protected against Insecure Direct Object Reference (IDOR) attacks targeting creation, reading, updating and deletion of records, such as creating or updating someone else's record, viewing everyone's records, or deleting all records.	5	Fully Implemented	2.12.1
Verify that the application or framework enforces a strong anti-CSRF mechanism to protect authenticated functionality, and effective anti-automation or anti-CSRF protects unauthenticated functionality.	5	Fully Implemented	2.12.2
Other Access Control Considerations			2.13
Verify administrative interfaces use appropriate multi-factor authentication to prevent unauthorized use.	5	Fully Implemented	2.13.1
Verify that directory browsing is disabled unless deliberately desired. Additionally, applications should not allow discovery or disclosure of file or directory metadata, such as Thumbs.db, .DS_Store, .git or .svn folders.	5	Fully Implemented	2.13.2
Input Validation Requirements			2.14

Verify that the application has defenses against HTTP parameter pollution attacks, particularly if the application framework makes no distinction about the source of request parameters (GET, POST, cookies, headers, or environment variables).	5	Fully Implemented	2.14.1
Verify that frameworks protect against mass parameter assignment attacks, or that the application has countermeasures to protect against unsafe parameter assignment, such as marking fields private or similar.	5	Fully Implemented	2.14.2
Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc) is validated using positive validation (allow lists).	5	Fully Implemented	2.14.3
Verify that structured data is strongly typed and validated against a defined schema including allowed characters, length and pattern (e.g. credit card numbers or telephone, or validating that two related fields are reasonable, such as checking that suburb and zip/postcode match).	5	Fully Implemented	2.14.4
Verify that URL redirects and forwards only allow destinations which appear on an allow list, or show a warning when redirecting to potentially untrusted content.	5	Fully Implemented	2.14.5
Sanitization and Sandboxing Requirements			2.15
Verify that all untrusted HTML input from WYSIWYG editors or similar is properly sanitized with an HTML sanitizer library or framework feature.	5	Fully Implemented	2.15.1
Verify that unstructured data is sanitized to enforce safety measures such as allowed characters and length.	5	Fully Implemented	2.15.2
Verify that the application sanitizes user input before passing to mail systems to protect against SMTP or IMAP injection.	5	Not Applicable	2.15.3
Verify that the application avoids the use of eval() or other dynamic code execution features. Where there is no alternative, any user input being included must be sanitized or sandboxed before being executed.	5	Fully Implemented	2.15.4
Verify that the application protects against template injection attacks by ensuring that any user input being included is sanitized or sandboxed.	5	Fully Implemented	2.15.5
Verify that the application protects against SSRF attacks, by validating or sanitizing untrusted data or HTTP file metadata, such as filenames and URL input fields, and uses allow lists of protocols, domains, paths and ports.	5	Fully Implemented	2.15.6
Verify that the application sanitizes, disables, or sandboxes user-supplied Scalable Vector Graphics (SVG) scriptable content, especially as they relate to XSS resulting from inline scripts, and foreignObject.	5	Not Applicable	2.15.7
Verify that the application sanitizes, disables, or sandboxes user-supplied scriptable or expression template language content, such as Markdown, CSS or XSL stylesheets, BBCode, or similar.	5	Fully Implemented	2.15.8
Output Encoding and Injection Prevention Requirements			2.16
Verify that output encoding is relevant for the interpreter and context required. For example, use encoders specifically for HTML values, HTML attributes, JavaScript, URL parameters, HTTP headers, SMTP, and others as the context requires, especially from untrusted inputs (e.g. names with Unicode or apostrophes, such as ねこ or O'Hara).	5	Fully Implemented	2.16.1
Verify that output encoding preserves the user's chosen character set and locale, such that any Unicode character point is valid and safely handled.	5	Fully Implemented	2.16.2
Verify that context-aware, preferably automated - or at worst, manual - output escaping protects against reflected, stored, and DOM based XSS.	5	Fully Implemented	2.16.3
Verify that data selection or database queries (e.g. SQL, HQL, ORM, NoSQL) use parameterized queries, ORMs, entity frameworks, or are otherwise protected from database injection attacks.	5	Fully Implemented	2.16.4

Verify that where parameterized or safer mechanisms are not present, context-specific output encoding is used to protect against injection attacks, such as the use of SQL escaping to protect against SQL injection.	5	Not Applicable	2.16.5
Verify that the application protects against JavaScript or JSON injection attacks, including for eval attacks, remote JavaScript includes, Content Security Policy (CSP) bypasses, DOM XSS, and JavaScript expression evaluation.	5	Fully Implemented	2.16.6
Verify that the application protects against LDAP injection vulnerabilities, or that specific security controls to prevent LDAP injection have been implemented.	5	Fully Implemented	2.16.7
Verify that the application protects against OS command injection and that operating system calls use parameterized OS queries or use contextual command line output encoding.	5	Fully Implemented	2.16.8
Verify that the application protects against Local File Inclusion (LFI) or Remote File Inclusion (RFI) attacks.	5	Not Applicable	2.16.9
Verify that the application protects against XPath injection or XML injection attacks.	5	Fully Implemented	2.16.10
Deserialization Prevention Requirements			2.17
Verify that serialized objects use integrity checks or are encrypted to prevent hostile object creation or data tampering.	5	Fully Implemented	2.17.1
Verify that the application correctly restricts XML parsers to only use the most restrictive configuration possible and to ensure that unsafe features such as resolving external entities are disabled to prevent XML eXternal Entity (XXE) attacks.	5	Not Applicable	2.17.2
Verify that deserialization of untrusted data is avoided or is protected in both custom code and third-party libraries (such as JSON, XML and YAML parsers).	5	Fully Implemented	2.17.3
Verify that when parsing JSON in browsers or JavaScript-based backends, JSON.parse is used to parse the JSON document. Do not use eval() to parse JSON.	5	Fully Implemented	2.17.4
Algorithms			2.18
Verify that all cryptographic modules fail securely, and errors are handled in a way that does not enable Padding Oracle attacks.	5	Fully Implemented	2.18.1
Log Content Requirements			2.19
Verify that the application does not log credentials or payment details. Session tokens should only be stored in logs in an irreversible, hashed form.	5	Fully Implemented	2.19.1
Verify that the application does not log other sensitive data as defined under local privacy laws or relevant security policy.	5	Fully Implemented	2.19.2
Error Handling			2.20
Verify that a generic message is shown when an unexpected or security sensitive error occurs, potentially with a unique ID which support personnel can use to investigate	5	Fully Implemented	2.20.1
Client-side Data Protection			2.21
Verify the application sets sufficient anti-caching headers so that sensitive data is not cached in modern browsers.	5	Fully Implemented	2.21.1
Verify that data stored in browser storage (such as HTML5 local storage, session storage, IndexedDB, or cookies) does not contain sensitive data or PII.	5	Fully Implemented	2.21.2
Verify that authenticated data is cleared from client storage, such as the browser DOM, after the client or session is terminated.	5	Fully Implemented	2.21.3
Sensitive Private Data			2.22

Verify that sensitive data is sent to the server in the HTTP message body or headers, and that query string parameters from any HTTP verb do not contain sensitive data.	5	Fully Implemented	2.22.1
Verify that users have a method to remove or export their data on demand.	5	Fully Implemented	2.22.2
Verify that users are provided clear language regarding collection and use of supplied personal information and that users have provided opt-in consent for the use of that data before it is used in any way.	5	Fully Implemented	2.22.3
Verify that all sensitive data created and processed by the application has been identified, and ensure that a policy is in place on how to deal with sensitive data.	5	Fully Implemented	2.22.4
Deployed Application Integrity Controls			2.23
Verify that if the application has a client or server auto-update feature, updates should be obtained over secure channels and digitally signed. The update code must validate the digital signature of the update before installing or executing the update.	5	Fully Implemented	2.23.1
Verify that the application employs integrity protections, such as code signing or subresource integrity. The application must not load or execute code from untrusted sources, such as loading includes, modules, plugins, code, or libraries from untrusted sources or the Internet.	5	Fully Implemented	2.23.2
Verify that the application has protection from subdomain takeovers if the application relies upon DNS entries or DNS subdomains, such as expired domain names, out of date DNS pointers or CNAMEs, expired projects at public source code repos, or transient cloud APIs, serverless functions, or storage buckets (*autogen-bucket-id* .cloud.example.com) or similar. Protections can include ensuring that DNS names used by applications are regularly checked for expiry or change.	5	Fully Implemented	2.23.3
Business Logic Security Requirements			2.24
Verify the application will only process business logic flows for the same user in sequential step order and without skipping steps.	5	Fully Implemented	2.24.1
Verify the application will only process business logic flows with all steps being processed in realistic human time, i.e. transactions are not submitted too quickly.	5	Fully Implemented	2.24.2
Verify the application has appropriate limits for specific business actions or transactions which are correctly enforced on a per user basis.	5	Fully Implemented	2.24.3
Verify the application has sufficient anti-automation controls to detect and protect against data exfiltration, excessive business logic requests, excessive file uploads or denial of service attacks.	5	Fully Implemented	2.24.4
Verify the application has business logic limits or validation to protect against likely business risks or threats, identified using threat modeling or similar methodologies.	5	Fully Implemented	2.24.5
File Upload Requirements			2.25
Verify that the application will not accept large files that could fill up storage or cause a denial of service.	5	Fully Implemented	2.25.1
File Execution Requirements			2.26
Verify that user-submitted filename metadata is not used directly by system or framework filesystems and that a URL API is used to protect against path traversal.	5	Fully Implemented	2.26.1
Verify that user-submitted filename metadata is validated or ignored to prevent the disclosure, creation, updating or removal of local files (LFI).	5	Fully Implemented	2.26.2

Verify that user-submitted filename metadata is validated or ignored to prevent the disclosure or execution of remote files via Remote File Inclusion (RFI) or Server-side Request Forgery (SSRF) attacks.	5	Fully Implemented	2.26.3
Verify that the application protects against Reflective File Download (RFD) by validating or ignoring user-submitted filenames in a JSON, JSONP, or URL parameter, the response Content-Type header should be set to text/plain, and the Content-Disposition header should have a fixed filename.	5	Fully Implemented	2.26.4
Verify that untrusted file metadata is not used directly with system API or libraries, to protect against OS command injection.	5	Fully Implemented	2.26.5
File Storage Requirements			2.27
Verify that files obtained from untrusted sources are stored outside the web root, with limited permissions, preferably with strong validation.	5	Fully Implemented	2.27.1
Verify that files obtained from untrusted sources are scanned by antivirus scanners to prevent upload of known malicious content.	5	Fully Implemented	2.27.2
File Download Requirements			2.28
Verify that the web tier is configured to serve only files with specific file extensions to prevent unintentional information and source code leakage. For example, backup files (e.g. .bak), temporary working files (e.g. .swp), compressed files (.zip, .tar.gz, etc) and other extensions commonly used by editors should be blocked unless required.	5	Fully Implemented	2.28.1
Verify that direct requests to uploaded files will never be executed as HTML/JavaScript content.	5	Fully Implemented	2.28.2
SSRF Protection Requirements			2.29
Verify that the web or application server is configured with an allow list of resources or systems to which the server can send requests or load data/files from.	5	Fully Implemented	2.29.1
Generic Web Service Security Verification Requirements			2.30
Verify that all application components use the same encodings and parsers to avoid parsing attacks that exploit different URI or file parsing behavior that could be used in SSRF and RFI attacks.	5	Fully Implemented	2.30.1
Verify that access to administration and management functions is limited to authorized administrators.	5	Fully Implemented	2.30.2
Verify API URLs do not expose sensitive information, such as the API key, session tokens etc.	5	Fully Implemented	2.30.3
RESTful Web Service Verification Requirements			2.31
Verify that enabled RESTful HTTP methods are a valid choice for the user or action, such as preventing normal users using DELETE or PUT on protected API or resources.	5	Fully Implemented	2.31.1
Verify that JSON schema validation is in place and verified before accepting input.	5	Fully Implemented	2.31.2
Verify that RESTful web services that utilize cookies are protected from Cross-Site Request Forgery via the use of at least one or more of the following: double submit cookie pattern, CSRF nonces, or Origin request header checks.	5	Fully Implemented	2.31.3
SOAP Web Service Verification Requirements			2.32
Verify that XSD schema validation takes place to ensure a properly formed XML document, followed by validation of each input field before any processing of that data takes place.	5	Fully Implemented	2.32.1
Dependency			2.33
Verify that all components are up to date, preferably using a dependency checker during build or compile time.	5	Fully Implemented	2.33.1
Verify that all unneeded features, documentation, samples, configurations are removed, such as sample applications, platform documentation, and default or example users.	5	Fully Implemented	2.33.2

Verify that if application assets, such as JavaScript libraries, CSS or web fonts, are hosted externally on a Content Delivery Network (CDN) or external provider, Subresource Integrity (SRI) is used to validate the integrity of the asset.	5	Not Implemented	2.33.3
Unintended Security Disclosure Requirements			2.34
Verify that web or application server and framework error messages are configured to deliver user actionable, customized responses to eliminate any unintended security disclosures.	5	Fully Implemented	2.34.1
Verify that web or application server and application framework debug modes are disabled in production to eliminate debug features, developer consoles, and unintended security disclosures.	5	Fully Implemented	2.34.2
Verify that the HTTP headers or any part of the HTTP response do not expose detailed version information of system components.	5	Fully Implemented	2.34.3
HTTP Security Headers Requirements			2.35
Verify that every HTTP response contains a Content-Type header. text/*, /+xml and application/xml content types should also specify a safe character set (e.g., UTF-8, ISO-8859-1).	5	Fully Implemented	2.35.1
Verify that all API responses contain a Content-Disposition: attachment; filename="api.json" header (or other appropriate filename for the content type).	5	Fully Implemented	2.35.2
Verify that a Content Security Policy (CSP) response header is in place that helps mitigate impact for XSS attacks like HTML, DOM, JSON, and JavaScript injection vulnerabilities.	5	Fully Implemented	2.35.3
Verify that all responses contain a X-Content-Type-Options: nosniff header.	5	Fully Implemented	2.35.4
Verify that a Strict-Transport-Security header is included on all responses and for all subdomains, such as Strict-Transport-Security: max-age=15724800; includeSubdomains.	5	Fully Implemented	2.35.5
Verify that a suitable "Referrer-Policy" header is included, such as "no-referrer" or "same-origin".	5	Fully Implemented	2.35.6
Verify that the content of a web application cannot be embedded in a third-party site by default and that embedding of the exact resources is only allowed where necessary by using suitable Content-Security-Policy: frame-ancestors and X-Frame-Options response headers.	5	Fully Implemented	2.35.7
Validate HTTP Request Header Requirements			2.36
Verify that the application server only accepts the HTTP methods in use by the application/API, including pre-flight OPTIONS, and logs/alerts on any requests that are not valid for the application context.	5	Fully Implemented	2.36.1
Verify that the supplied Origin header is not used for authentication or access control decisions, as the Origin header can easily be changed by an attacker.	5	Fully Implemented	2.36.2
Verify that the Cross-Origin Resource Sharing (CORS) Access-Control-Allow-Origin header uses a strict allow list of trusted domains and subdomains to match against and does not support the "null" origin.	5	Fully Implemented	2.36.3
Select Controls from Open Web Application Security Project - Application Security Verification Standard 4.0.2 Level 2			3
Secure Software Development Lifecycle			3.1
Verify the use of a secure software development lifecycle that addresses security in all stages of development	3	Fully Implemented	3.1.1
Verify documentation and justification of all the application's trust boundaries, components, and significant data flows.	3	Fully Implemented	3.1.2
Verify implementation of centralized, simple (economy of design), vetted, secure, and reusable security controls to avoid duplicate, missing, ineffective, or insecure controls.	3	Fully Implemented	3.1.3

Verify availability of a secure coding checklist, security requirements, guideline, or policy to all developers and testers.	3	Fully Implemented	3.1.4
Authentication Architectural Requirements			3.2
Verify that communications between application components, including APIs, middleware and data layers, are authenticated. Components should have the least necessary privileges needed	3	Fully Implemented	3.2.1
Verify that the application uses a single vetted authentication mechanism that is known to be secure, can be extended to include strong authentication, and has sufficient logging and monitoring to detect account abuse or breaches.	3	Fully Implemented	3.2.2
Verify that all authentication pathways and identity management APIs implement consistent authentication security control strength, such that there are no weaker alternatives per the risk of the application.	3	Fully Implemented	3.2.3
Access Control Architectural Requirements			3.3
Verify enforcement of the principle of least privilege in functions, data files, URLs, controllers, services, and other resources. This implies protection against spoofing and elevation of privilege.	3	Fully Implemented	3.3.1
Verify the application uses a single and well-vetted access control mechanism for accessing protected data and resources. All requests must pass through this single mechanism to avoid copy and paste or insecure alternative paths.	3	Fully Implemented	3.3.2
Verify that attribute or feature-based access control is used whereby the code checks the user's authorization for a feature/data item rather than just their role. Permissions should still be allocated using roles.	3	Fully Implemented	3.3.3
Input and Output Architectural Requirements			3.4
Verify that input and output requirements clearly define how to handle and process data based on type, content, and applicable laws, regulations, and other policy compliance.	3	Fully Implemented	3.4.1
Verify that serialization is not used when communicating with untrusted clients. If this is not possible, ensure that adequate integrity controls (and possibly encryption if sensitive data is sent) are enforced to prevent deserialization attacks including object injection.	3	Fully Implemented	3.4.2
Cryptographic Architectural Requirements			3.5
Verify that there is an explicit policy for management of cryptographic keys and that a cryptographic key lifecycle follows a key management standard such as NIST SP 800-57.	3	Fully Implemented	3.5.1
Verify that consumers of cryptographic services protect key material and other secrets by using key vaults or API based alternatives.	3	Fully Implemented	3.5.2
Verify that all keys and passwords are replaceable and are part of a well-defined process to re-encrypt sensitive data.	3	Implemented	3.5.3
Verify that the architecture treats client-side secrets--such as symmetric keys, passwords, or API tokens--as insecure and never uses them to protect or access sensitive data.	3	Fully Implemented	3.5.4
Errors, Logging and Auditing Architectural Requirements			3.6
Verify that a common logging format and approach is used across the system.	3	Fully Implemented	3.6.1
Verify that logs are securely transmitted to a preferably remote system for analysis, detection, alerting, and escalation.	3	Fully Implemented	3.6.2
Data Protection and Privacy Architectural Requirements			3.7
Verify that all sensitive data is identified and classified into protection levels.	3	Fully Implemented	3.7.1
Verify that all protection levels have an associated set of protection requirements, such as encryption requirements, integrity requirements, retention, privacy and other confidentiality requirements, and that these are applied in the architecture.	3	Fully Implemented	3.7.2

Communications Architectural Requirements			3.8
Verify the application encrypts communications between components, particularly when these components are in different containers, systems, sites, or cloud providers.	3	Fully Implemented	3.8.1
Verify that application components verify the authenticity of each side in a communication link to prevent person-in-the-middle attacks. For example, application components should validate TLS certificates and chains.	3	Fully Implemented	3.8.2
Malicious Software Architectural Requirements			3.9
Verify that a source code control system is in use, with procedures to ensure that check-ins are accompanied by issues or change tickets. The source code control system should have access control and identifiable users to allow traceability of any changes.	3	Fully Implemented	3.9.1
Secure File Upload Architectural Requirements			3.10
Verify that user-uploaded files are stored outside of the web root.	3	Fully Implemented	3.10.1
Verify that user-uploaded files - if required to be displayed or downloaded from the application - are served by either octet stream downloads, or from an unrelated domain, such as a cloud file storage bucket. Implement a suitable Content Security Policy (CSP) to reduce the risk from XSS vectors or other attacks from the uploaded file.	3	Fully Implemented	3.10.2
Configuration Architectural Requirements			3.11
Verify the segregation of components of differing trust levels through well-defined security controls, firewall rules, API gateways, reverse proxies, cloud-based security groups, or similar mechanisms.	3	Fully Implemented	3.11.1
Verify that the build pipeline warns of out-of-date or insecure components and takes appropriate actions.	3	Fully Implemented	3.11.2
Verify that the build pipeline contains a build step to automatically build and verify the secure deployment of the application, particularly if the application infrastructure is software defined, such as cloud environment build scripts.	3	Fully Implemented	3.11.3
Verify that application deployments adequately sandbox, containerize and/or isolate at the network level to delay and deter attackers from attacking other applications, especially when they are performing sensitive or dangerous actions such as deserialization.	3	Fully Implemented	3.11.4
Verify the application does not use unsupported, insecure, or deprecated client-side technologies such as NSAPI plugins, Flash, Shockwave, ActiveX, Silverlight, NACL, or client-side Java applets.	3	Fully Implemented	3.11.5
Credential Storage Requirements			3.12
Verify that passwords are stored in a form that is resistant to offline attacks. Passwords SHALL be salted and hashed using an approved one-way key derivation or password hashing function. Key derivation and password hashing functions take a password, a salt, and a cost factor as inputs when generating a password hash	3	Fully Implemented	3.12.1
Verify that the salt is at least 32 bits in length and be chosen arbitrarily to minimize salt value collisions among stored hashes. For each credential, a unique salt value and the resulting hash SHALL be stored.	3	Fully Implemented	3.12.2
Verify that if PBKDF2 is used, the iteration count SHOULD be as large as verification server performance will allow, typically at least 100,000 iterations.	3	Not Applicable	3.12.3
Credential Recovery Requirements			3.13
Verify that if OTP or multi-factor authentication factors are lost, that evidence of identity proofing is performed at the same level as during enrollment.	3	Fully Implemented	3.13.1
Look-up Secret Verifier Requirements			3.14
Verify that lookup secrets can be used only once.	3	Not Applicable	3.14.1

Verify that lookup secrets have sufficient randomness (112 bits of entropy), or if less than 112 bits of entropy, salted with a unique and random 32-bit salt and hashed with an approved one-way hash.	3	Fully Implemented	3.14.2
Verify that lookup secrets are resistant to offline attacks, such as predictable values.	3	Not Applicable	3.14.3
Out of Band Verifier Requirements			3.15
Verify that the out of band verifier retains only a hashed version of the authentication code.	3	Not Applicable	3.15.1
Verify that the initial authentication code is generated by a secure random number generator, containing at least 20 bits of entropy (typically a six digital random number is sufficient).	3	Not Applicable	3.15.2
Single or Multi-factor One Time Verifier Requirements			3.16
Verify that symmetric keys used to verify submitted OTPs are highly protected, such as by using a hardware security module or secure operating system based key storage.	3	Not Applicable	3.16.1
Verify that approved cryptographic algorithms are used in the generation, seeding, and verification of OTPs.	3	Not Applicable	3.16.2
Verify that time-based OTP can be used only once within the validity period.	3	Fully Implemented	3.16.3
Verify that if a time-based multi-factor OTP token is re-used during the validity period, it is logged and rejected with secure notifications being sent to the holder of the device.	3	Partially Implemented	3.16.4
Cryptographic Software and Devices Verifier Requirements			3.17
Verify that cryptographic keys used in verification are stored securely and protected against disclosure, such as using a Trusted Platform Module (TPM) or Hardware Security Module (HSM), or an OS service that can use this secure storage.	3	Fully Implemented	3.17.1
Verify that the challenge nonce is at least 64 bits in length, and statistically unique or unique over the lifetime of the cryptographic device.	3	Not Applicable	3.17.2
Verify that approved cryptographic algorithms are used in the generation, seeding, and verification.	3	Not Applicable	3.17.3
Service Authentication Requirements			3.18
Verify that if passwords are required for service authentication, the service account used is not a default credential. (e.g. root/root or admin/admin are default in some services during installation).	3	Fully Implemented	3.18.1
Verify that passwords are stored with sufficient protection to prevent offline recovery attacks, including local system access.	3	Fully Implemented	3.18.2
Session Binding Requirements			3.19
Verify that session token are generated using approved cryptographic algorithms.	3	Fully Implemented	3.19.1
Session Logout and Timeout Requirements			3.20
Verify that the application gives the option to terminate all other active sessions after a successful password change (including change via password reset/recovery), and that this is effective across the application, federated login (if present), and any relying parties.	3	Fully Implemented	3.20.1
Verify that users are able to view and (having re-entered login credentials) log out of any or all currently active sessions and devices.	3	Not Applicable	3.20.2
Token-Based Session Management			3.21
Verify the application uses session tokens rather than static API secrets and keys, except with legacy implementations.	3	Fully Implemented	3.21.1
Verify that stateless session tokens use digital signatures, encryption, and other countermeasures to protect against tampering, enveloping, replay, null cipher, and key substitution attacks.	3	Fully Implemented	3.21.2
Other Access Control Considerations			3.22

Verify the application has additional authorization (such as step up or adaptive authentication) for lower value systems, and / or segregation of duties for high value applications to enforce anti-fraud controls as per the risk of application and past fraud.	3	Fully Implemented	3.22.1
Data Classificiation			3.23
Verify that regulated private data is stored encrypted while at rest, such as Personally Identifiable Information (PII), sensitive personal information, or data assessed likely to be subject to EU's GDPR.	3	Fully Implemented	3.23.1
Algorithms			3.24
Verify that industry proven or government approved cryptographic algorithms, modes, and libraries are used, instead of custom coded cryptography.	3	Fully Implemented	3.24.1
Verify that encryption initialization vector, cipher configuration, and block modes are configured securely using the latest advice.	3	Fully Implemented	3.24.2
Verify that random number, encryption or hashing algorithms, key lengths, rounds, ciphers or modes, can be reconfigured, upgraded, or swapped at any time, to protect against cryptographic breaks.	3	Fully Implemented	3.24.3
Verify that known insecure block modes (i.e. ECB, etc.), padding modes (i.e. PKCS#1 v1.5, etc.), ciphers with small block sizes (i.e. Triple-DES, Blowfish, etc.), and weak hashing algorithms (i.e. MD5, SHA1, etc.) are not used unless required for backwards compatibility.	3	Fully Implemented	3.24.4
Verify that nonces, initialization vectors, and other single use numbers must not be used more than once with a given encryption key. The method of generation must be appropriate for the algorithm being used.	3	Fully Implemented	3.24.5
Random Values			3.25
Verify that all random numbers, random file names, random GUIDs, and random strings are generated using the cryptographic module's approved cryptographically secure random number generator when these random values are intended to be not guessable by an attacker.	3	Fully Implemented	3.25.1
Verify that random GUIDs are created using the GUID v4 algorithm, and a Cryptographically-secure Pseudo-random Number Generator (CSPRNG). GUIDs created using other pseudo-random number generators may be predictable.	3	Fully Implemented	3.25.2
Secret Management			3.26
Verify that a secrets management solution such as a key vault is used to securely create, store, control access to and destroy secrets.	3	Fully Implemented	3.26.1
Verify that key material is not exposed to the application but instead uses an isolated security module like a vault for cryptographic operations	3	Fully Implemented	3.26.2
Log Content Requirements			3.27
Verify that the application logs security relevant events including successful and failed authentication events, access control failures, deserialization failures and input validation failures.	3	Fully Implemented	3.27.1
Verify that each log event includes necessary information that would allow for a detailed investigation of the timeline when an event happens	3	Fully Implemented	3.27.2
Log Processing Requirements			3.28
Verify that all authentication decisions are logged, without storing sensitive session tokens or passwords. This should include requests with relevant metadata needed for security investigations.	3	Fully Implemented	3.28.1

Verify that all access control decisions can be logged and all failed decisions are logged. This should include requests with relevant metadata needed for security investigations.	3	Fully Implemented	3.28.2
Log Protection Requirements			3.29
Verify that the application appropriately encodes user-supplied data to prevent log injection.	3	Fully Implemented	3.29.1
Verify that all events are protected from injection when viewed in log viewing software.	3	Fully Implemented	3.29.2
Verify that security logs are protected from unauthorized access and modification.	3	Fully Implemented	3.29.3
Verify that time sources are synchronized to the correct time and time zone. Strongly consider logging only in UTC if systems are global to assist with post-incident forensic analysis.	3	Fully Implemented	3.29.4
Error Handling			3.30
Verify that a "last resort" error handler is defined which will catch all unhandled exceptions.	3	Fully Implemented	3.30.1
General Data Protection			3.31
Verify the application protects sensitive data from being cached in server components such as load balancers and application caches.	3	Fully Implemented	3.31.1
Verify that all cached or temporary copies of sensitive data stored on the server are protected from unauthorized access or purged/invalidated after the authorized user accesses the sensitive data.	3	Fully Implemented	3.31.2
Verify the application minimizes the number of parameters in a request, such as hidden fields, Ajax variables, cookies and header values.	3	Fully Implemented	3.31.3
Verify the application can detect and alert on abnormal numbers of requests, such as by IP, user, total per hour or day, or whatever makes sense for the application.	3	Fully Implemented	3.31.4
Sensitive Private Data			3.32
Verify accessing sensitive data is audited (without logging the sensitive data itself), if the data is collected under relevant data protection directives or where logging of access is required.	3	Fully Implemented	3.32.1
Verify that sensitive information contained in memory is overwritten as soon as it is no longer required to mitigate memory dumping attacks, using zeroes or random data.	3	Not Applicable	3.32.2
Verify that sensitive or private information that is required to be encrypted, is encrypted using approved algorithms that provide both confidentiality and integrity.	3	Fully Implemented	3.32.3
Verify that sensitive personal information is subject to data retention classification, such that old or out of date data is deleted automatically, on a schedule, or as the situation requires.	3	Fully Implemented	3.32.4
Client Communications Security Requirements			3.33
Verify that secured TLS is used for all client connectivity, and does not fall back to insecure or unencrypted protocols.	3	Fully Implemented	3.33.1
Verify using online or up to date TLS testing tools that only strong algorithms, ciphers, and protocols are enabled, with the strongest algorithms and ciphers set as preferred.	3	Fully Implemented	3.33.2
Verify that old versions of SSL and TLS protocols, algorithms, ciphers, and configuration are disabled, such as SSLv2, SSLv3, or TLS 1.0 and TLS 1.1. The latest version of TLS should be the preferred cipher suite.	3	Fully Implemented	3.33.3
Server Communications Security Requirements			3.34
Verify that connections to and from the server use trusted TLS certificates. Where internally generated or self-signed certificates are used, the server must be configured to only trust specific internal CAs and specific self-signed certificates. All others should be rejected.	3	Fully Implemented	3.34.1

Verify that encrypted communications such as TLS is used for all inbound and outbound connections, including for management ports, monitoring, authentication, API, or web service calls, database, cloud, serverless, mainframe, external, and partner connections. The server must not fall back to insecure or unencrypted protocols.	3	Fully Implemented	3.34.2
Verify that all encrypted connections to external systems that involve sensitive information or functions are authenticated.	3	Not Applicable	3.34.3
Verify that proper certification revocation, such as Online Certificate Status Protocol (OCSP) Stapling, is enabled and configured.	3	Fully Implemented	3.34.4
Malicious Code Search			3.35
Verify that the application source code and third party libraries do not contain unauthorized phone home or data collection capabilities. Where such functionality exists, obtain the user's permission for it to operate before collecting any data.	3	Fully Implemented	3.35.1
Verify that the application does not ask for unnecessary or excessive permissions to privacy related features or sensors, such as contacts, cameras, microphones, or location.	3	Fully Implemented	3.35.2
Business Logic Security Requirements			3.36
Verify the application has configurable alerting when automated attacks or unusual activity is detected.	3	Fully Implemented	3.36.1
File Integrity Requirements			3.37
Verify that files obtained from untrusted sources are validated to be of expected type based on the file's content.	3	Fully Implemented	3.37.1
File Execution Requirements			3.38
Verify that the application does not include and execute functionality from untrusted sources, such as unverified content distribution networks, JavaScript libraries, node npm libraries, or server-side DLLs.	3	Fully Implemented	3.38.1
RESTful Web Service Verification Requirements			3.39
Verify that the message headers and payload are trustworthy and not modified in transit. Requiring strong encryption for transport (TLS only) may be sufficient in many cases as it provides both confidentiality and integrity protection. Per-message digital signatures can provide additional assurance on top of the transport protections for high-security applications but bring with them additional complexity and risks to weigh against the benefits.	3	Fully Implemented	3.39.1
SOAP Web Service Verification Requirements			3.40
Verify that the message payload is signed using WS-Security to ensure reliable transport between client and service.	3	Not Applicable	3.40.1
GraphQL and other Web Service Data Layer Security Requirements			3.41
Verify that a query allow list or a combination of depth limiting and amount limiting is used to prevent GraphQL or data layer expression Denial of Service (DoS) as a result of expensive, nested queries. For more advanced scenarios, query cost analysis should be used.	3	Not Applicable	3.41.1
Build			3.42
Verify that the application build and deployment processes are performed in a secure and repeatable way, such as CI / CD automation, automated configuration management, and automated deployment scripts.	3	Fully Implemented	3.42.1
Verify that compiler flags are configured to enable all available buffer overflow protections and warnings, including stack randomization, data execution prevention, and to break the build if an unsafe pointer, memory, format string, integer, or string operations are found.	3	Fully Implemented	3.42.2

Verify that server configuration is hardened as per the recommendations of the application server and frameworks in use.	3	Fully Implemented	3.42.3
Verify that the application, configuration, and all dependencies can be re-deployed using automated deployment scripts, built from a documented and tested runbook in a reasonable time, or restored from backups in a timely fashion.	3	Fully Implemented	3.42.4
Dependency			3.43
Verify that third party components come from pre-defined, trusted and continually maintained repositories	3	Fully Implemented	3.43.1
Verify that the attack surface is reduced by sandboxing or encapsulating third party libraries to expose only the required behaviour into the application	3	Fully Implemented	3.43.2
Validate HTTP Request Header Requirements			3.44
Verify that HTTP headers added by a trusted proxy or SSO devices, such as a bearer token, are authenticated by the application.	3	Fully Implemented	3.44.1
Select Controls from Open Web Application Security Project - Application Security Verification Standard 4.0.2 Level 3			4
General Authenticator Requirements			4.1
Verify impersonation resistance against phishing, such as the use of multi-factor authentication, cryptographic devices with intent (such as connected keys with a push to authenticate), or at higher AAL levels, client-side certificates.	1	Fully Implemented	4.1.1
Verify that where a Credential Service Provider (CSP) and the application verifying authentication are separated, mutually authenticated TLS is in place between the two endpoints.	1	Fully Implemented	4.1.2
Session Logout and Timeout Requirements			4.2
If authenticators permit users to remain logged in, verify that re-authentication occurs periodically with 2FA both when actively used after 12 hours or after an idle period of 15 minutes	1	Partially Implemented - Compensating Controls	4.2.1
Algorithms			4.3
Verify that encrypted data is authenticated via signatures, authenticated cipher modes, or HMAC to ensure that ciphertext is not altered by an unauthorized party	1	Not Implemented	4.3.1
General Data Protection			4.4
Verify that regular backups of important data are performed and that test restoration of data is performed.	1	Implemented - Compensating Controls	4.4.1
Verify that backups are stored securely to prevent data from being stolen or corrupted.	1	Fully Implemented	4.4.2
Code Integrity Controls			4.5
Verify that a code analysis tool is in use that can detect potentially malicious code, such as time functions, unsafe file operations and network connections.	1	Fully Implemented	4.5.1
Malicious Code Search			4.6

Verify that the application source code and third party libraries do not contain back doors, such as hard-coded or additional undocumented accounts or keys, code obfuscation, undocumented binary blobs, rootkits, or anti-debugging, insecure debugging features, or otherwise out of date, insecure, or hidden functionality that could be used maliciously if discovered.	1	Fully Implemented	4.6.1
Verify that the application source code and third party libraries do not contain time bombs by searching for date and time related functions.	1	Fully Implemented	4.6.2
Verify that the application source code and third party libraries do not contain malicious code, such as salami attacks, logic bypasses, or logic bombs.	1	Fully Implemented	4.6.3
Verify that the application source code and third party libraries do not contain Easter eggs or any other potentially unwanted functionality.	1	Fully Implemented	4.6.4

Total Possible Points **955**

Attachment C

OWASP Mobile Application Level Security Verification

Open Web Application Security Project - Mobile Application Security Verification Standard 1.2 Level 1	Points for Meeting Requirements	Status
Architecture, design and threat modelling		
All app components are identified and known to be needed.	5	N/A
Security controls are never enforced only on the client side, but on the respective remote endpoints.	5	N/A
A high-level architecture for the mobile app and all connected remote services has been defined and security has been addressed in that architecture.	5	N/A
Data considered sensitive in the context of the mobile app is clearly identified.	5	N/A
The app should comply with privacy laws and regulations.	5	N/A
Data Storage and Privacy		
System credential storage facilities need to be used to store sensitive data, such as PII, user credentials or cryptographic keys.	5	N/A
No sensitive data should be stored outside of the app container or system credential storage facilities.	5	N/A
No sensitive data is written to application logs.	5	N/A
No sensitive data is shared with third parties unless it is a necessary part of the architecture.	5	N/A
The keyboard cache is disabled on text inputs that process sensitive data.	5	N/A
No sensitive data is exposed via IPC mechanisms.	5	N/A
No sensitive data, such as passwords or pins, is exposed through the user interface.	5	N/A
Cryptography		
The app does not rely on symmetric cryptography with hardcoded keys as a sole method of encryption.	5	N/A
The app uses proven implementations of cryptographic primitives.	5	N/A
The app uses cryptographic primitives that are appropriate for the particular use-case, configured with parameters that adhere to industry best practices.	5	N/A
The app does not use cryptographic protocols or algorithms that are widely considered deprecated for security purposes.	5	N/A
The app doesn't re-use the same cryptographic key for multiple purposes.	5	N/A
All random values are generated using a sufficiently secure random number generator.	5	N/A
Authentication and Session Management		
If the app provides users access to a remote service, some form of authentication, such as username/password authentication, is performed at the remote endpoint.	5	N/A
If stateful session management is used, the remote endpoint uses randomly generated session identifiers to authenticate client requests without sending the user's credentials.	5	N/A
If stateless token-based authentication is used, the server provides a token that has been signed using a secure algorithm.	5	N/A
The remote endpoint terminates the existing session when the user logs out.	5	N/A
A password policy exists and is enforced at the remote endpoint.	5	N/A
The remote endpoint implements a mechanism to protect against the submission of credentials an excessive number of times.	5	N/A
Sessions are invalidated at the remote endpoint after a predefined period of inactivity and access tokens expire.	5	N/A
Network Communication		
Data is encrypted on the network using TLS. The secure channel is used consistently throughout the app.	5	N/A
The TLS settings are in line with current best practices, or as close as possible if the mobile operating system does not support the recommended standards.	5	N/A
The app verifies the X.509 certificate of the remote endpoint when the secure channel is established. Only certificates signed by a trusted CA are accepted.	5	N/A
Platform Interaction		
The app only requests the minimum set of permissions necessary.	5	N/A
All inputs from external sources and the user are validated and if necessary sanitized. This includes data received via the UI, IPC mechanisms such as intents, custom URLs, and network sources.	5	N/A
The app does not export sensitive functionality via custom URL schemes, unless these mechanisms are properly protected.	5	N/A
The app does not export sensitive functionality through IPC facilities, unless these mechanisms are properly protected.	5	N/A
JavaScript is disabled in WebViews unless explicitly required.	5	N/A
WebViews are configured to allow only the minimum set of protocol handlers required (ideally, only https is supported). Potentially dangerous handlers, such as file, tel and app-id, are disabled.	5	N/A
If native methods of the app are exposed to a WebView, verify that the WebView only renders JavaScript contained within the app package.	5	N/A
Object deserialization, if any, is implemented using safe serialization APIs.	5	N/A
Code Quality and Build Settings		
The app is signed and provisioned with a valid certificate, of which the private key is properly protected.	5	N/A
The app has been built in release mode, with settings appropriate for a release build (e.g. non-debuggable).	5	N/A

Debugging symbols have been removed from native binaries.	5	N/A
Debugging code and developer assistance code (e.g. test code, backdoors, hidden settings) have been removed. The app does not log verbose errors or debugging messages.	5	N/A
All third party components used by the mobile app, such as libraries and frameworks, are identified, and checked for known vulnerabilities.	5	N/A
The app catches and handles possible exceptions.	5	N/A
Error handling logic in security controls denies access by default.	5	N/A
In unmanaged code, memory is allocated, freed and used securely.	5	N/A
Free security features offered by the toolchain, such as byte-code minification, stack protection, PIE support and automatic reference counting, are activated.	5	N/A
Select Controls from Open Web Application Security Project - Mobile Application Security Verification Standard 1.2 Level 2		
Architecture, design and threat modelling		
All app components are defined in terms of the business functions and/or security functions they provide.	3	N/A
A threat model for the mobile app and the associated remote services has been produced that identifies potential threats and countermeasures.	3	N/A
All security controls have a centralized implementation.	3	N/A
There is an explicit policy for how cryptographic keys (if any) are managed, and the lifecycle of cryptographic keys is enforced. Ideally, follow a key management standard such as NIST SP 800-57.	3	N/A
A mechanism for enforcing updates of the mobile app exists.	3	N/A
Security is addressed within all parts of the software development lifecycle.	3	N/A
A responsible disclosure policy is in place and effectively applied.	3	N/A
Data Storage and Privacy		
No sensitive data is included in backups generated by the mobile operating system.	3	N/A
The app removes sensitive data from views when moved to the background.	3	N/A
The app does not hold sensitive data in memory longer than necessary, and memory is cleared explicitly after use.	3	N/A
The app educates the user about the types of personally identifiable information processed, as well as security best practices the user should follow in using the app.	3	N/A
No sensitive data should be stored locally on the mobile device. Instead, data should be retrieved from a remote endpoint when needed and only be kept in memory.	3	N/A
Authentication and Session Management		
A second factor of authentication exists at the remote endpoint and the 2FA requirement is consistently enforced.	3	N/A
Network Communication		
The app either uses its own certificate store, or pins the endpoint certificate or public key, and subsequently does not establish connections with endpoints that offer a different certificate or key, even if signed by a trusted CA.	3	N/A
The app doesn't rely on a single insecure communication channel (email or SMS) for critical operations, such as enrollments and account recovery.	3	N/A
The app only depends on up-to-date connectivity and security libraries.	3	N/A
Platform Interaction		
The app protects itself against screen overlay attacks. (Android only)	3	N/A
A WebView's cache, storage, and loaded resources (JavaScript, etc.) should be cleared before the WebView is destroyed.	3	N/A
Verify that the app prevents usage of custom third-party keyboards whenever sensitive data is entered.	3	N/A
Select Controls from Open Web Application Security Project - Mobile Application Security Verification Standard 1.2 Resiliency against Reverse Engineering		
Impede Dynamic Analysis and Tampering		
The app detects, and responds to, the presence of a rooted or jailbroken device either by alerting the user or terminating the app.	5	N/A
The app prevents debugging and/or detects, and responds to, a debugger being attached. All available debugging protocols must be covered.	5	N/A
Obfuscation is applied to programmatic defenses, which in turn impede de-obfuscation via dynamic analysis.	5	N/A
Impede Comprehension		
All executable files and libraries belonging to the app are either encrypted on the file level and/or important code and data segments inside the executables are encrypted or packed. Trivial static analysis does not reveal important code or data.	5	N/A
Impede Eavesdropping		
As a defense in depth, next to having solid hardening of the communicating parties, application level payload encryption can be applied to further impede eavesdropping.	5	N/A

Total Possible Points **307**

Attachment D

Security Requirements for Databases

Select Controls from Department of Defense - Security Requirements Guide for Databases (Moderate Controls)	Points for Meeting Requirements	Status	Documentation
The DBMS must limit the number of concurrent sessions to an organization-defined number per user for all accounts and/or account types.	3	Fully Implemented	2.1.1
The DBMS must protect against a user falsely repudiating having performed organization-defined actions.	MANDATORY	Fully Implemented	2.1.2
The DBMS must be able to generate audit records when privileges/permissions are retrieved.	3	Fully Implemented	2.1.3
The DBMS must be able to generate audit records when unsuccessful attempts to retrieve privileges/permissions occur.	3	Fully Implemented	2.1.4
The DBMS must initiate session auditing upon startup.	MANDATORY	Fully Implemented	2.1.5
The DBMS must produce audit records containing sufficient information to establish what type of events occurred.	3	Fully Implemented	2.1.6
The DBMS must produce audit records containing time stamps to establish when the events occurred.	3	Fully Implemented	2.1.7
The DBMS must produce audit records containing sufficient information to establish where the events occurred.	MANDATORY	Fully Implemented	2.1.8
The DBMS must produce audit records containing sufficient information to establish the sources (origins) of the events.	MANDATORY	Fully Implemented	2.1.9
The DBMS must produce audit records containing sufficient information to establish the outcome (success or failure) of the events.	3	Fully Implemented	2.1.10
The DBMS must produce audit records containing sufficient information to establish the identity of any user/subject or process associated with the event.	3	Fully Implemented	2.1.11
The DBMS must include additional, more detailed, organization-defined information in the audit records for audit events identified by type, location, or subject.	3	Fully Implemented	2.1.12
The DBMS must by default shut down upon audit failure, to include the unavailability of space for more audit log records; or must be configurable to shut down upon audit failure.	3	Fully Implemented	2.1.13
The DBMS must be configurable to overwrite audit log records, oldest first (First-In-First-Out - FIFO), in the event of unavailability of space for more audit log records.	MANDATORY	Fully Implemented	2.1.14
The DBMS must use system clocks to generate time stamps for use in audit records and application data.	3	Fully Implemented	2.1.15

The audit information produced by the DBMS must be protected from unauthorized read access.	MANDATORY	Fully Implemented	2.1.16
The audit information produced by the DBMS must be protected from unauthorized modification.	MANDATORY	Fully Implemented	2.1.17
The audit information produced by the DBMS must be protected from unauthorized deletion.	MANDATORY	Fully Implemented	2.1.18
The DBMS must protect its audit features from unauthorized access.	MANDATORY	Fully Implemented	2.1.19
The DBMS must protect its audit configuration from unauthorized modification.	MANDATORY	Fully Implemented	2.1.20
The DBMS must protect its audit features from unauthorized removal.	MANDATORY	Fully Implemented	2.1.21
The DBMS must limit privileges to change software modules, to include stored procedures, functions and triggers, and links to software external to the DBMS.	3	Fully Implemented	2.1.22
The DBMS software installation account must be restricted to authorized users.	MANDATORY	Fully Implemented	2.1.23
Database software, including DBMS configuration files, must be stored in dedicated directories, or DASD pools, separate from the host OS and other applications.	3	Fully Implemented	2.1.24
Database objects (including but not limited to tables, indexes, storage, stored procedures, functions, triggers, links to software external to the DBMS, etc.) must be owned by database/DBMS principals authorized for ownership.	3	Fully Implemented	2.1.25
The role(s)/group(s) used to modify database structure (including but not necessarily limited to tables, indexes, storage, etc.) and logic modules (stored procedures, functions, triggers, links to software external to the DBMS, etc.) must be restricted to authorized users.	MANDATORY	Fully Implemented	2.1.26
Default demonstration and sample databases, database objects, and applications must be removed.	3	Fully Implemented	2.1.27
Unused database components, DBMS software, and database objects must be removed.	3	Fully Implemented	2.1.28
Unused database components that are integrated in the DBMS and cannot be uninstalled must be disabled.	3	Fully Implemented	2.1.29
Access to external executables must be disabled or restricted.	3	Fully Implemented	2.1.30
The DBMS must uniquely identify and authenticate organizational users (or processes acting on behalf of organizational users).	3	Fully Implemented	2.1.31
If passwords are used for authentication, the DBMS must store only hashed, salted representations of passwords.	MANDATORY	Fully Implemented	2.1.32

If passwords are used for authentication, the DBMS must transmit only encrypted representations of passwords.	**MANDATORY**	**Not Applicable**	**2.1.33**
The DBMS must obscure feedback of authentication information during the authentication process to protect the information from possible exploitation/use by unauthorized individuals.	**3**	**Fully Implemented**	**2.1.34**
The DBMS must use NIST FIPS 140-2 validated cryptographic modules for cryptographic operations.	**MANDATORY**	**Fully Implemented**	**2.1.35**
The DBMS must separate user functionality (including user interface services) from database management functionality.	**3**	**Fully Implemented**	**2.1.36**
The DBMS must invalidate session identifiers upon user logout or other session termination.	**MANDATORY**	**Fully Implemented**	**2.1.37**
The DBMS must recognize only system-generated session identifiers.	**3**	**Fully Implemented**	**2.1.38**
The DBMS must maintain the authenticity of communications sessions by guarding against man-in-the-middle attacks that guess at Session ID values.	**MANDATORY**	**Fully Implemented**	**2.1.39**
The DBMS must fail to a secure state if system initialization fails, shutdown fails, or aborts fail.	**3**	**Fully Implemented**	**2.1.40**
In the event of a system failure, the DBMS must preserve any information necessary to determine cause of failure and any information necessary to return to operations with least disruption to mission processes.	**MANDATORY**	**Fully Implemented**	**2.1.41**
The DBMS must protect the confidentiality and integrity of all information at rest.	**MANDATORY**	**Fully Implemented**	**2.1.42**
The DBMS must isolate security functions from non-security functions.	**MANDATORY**	**Fully Implemented**	**2.1.43**

Total Points **66** **0**

Attachment E

Select Controls From NIST SP 800-171

Select Controls from the StateRAMP Moderate Baseline	Points for Meeting Requirements	Status	Documentation
Limit system access to authorized users, processes acting on behalf of authorized users, and devices (including other systems).	MANDATORY	Fully Implemented	2.1.1
Limit system access to the types of transactions and functions that authorized users are permitted to execute.	MANDATORY	Fully Implemented	2.1.2
Monitor and control remote access sessions.	MANDATORY	Fully Implemented	2.1.3
Employ cryptographic mechanisms to protect the confidentiality of remote access sessions.	MANDATORY	Fully Implemented	2.1.4
Authorize wireless access prior to allowing such connections.	5	Fully Implemented	2.1.5
Protect wireless access using authentication and encryption.	MANDATORY	Fully Implemented	2.1.6
Control connection of mobile devices.	5	Fully Implemented	2.1.7
Ensure that managers, systems administrators, and users of organizational systems are made aware of the security risks associated with their activities and of the applicable policies, standards, and procedures related to the security of those systems.	MANDATORY	Fully Implemented	2.1.8
Ensure that personnel are trained to carry out their assigned information security-related duties and responsibilities.	MANDATORY	Fully Implemented	2.1.9
Create and retain system audit logs and records to the extent needed to enable the monitoring, analysis, investigation, and reporting of unlawful or unauthorized system activity.	MANDATORY	Fully Implemented	2.1.10
Correlate audit record review, analysis, and reporting processes for investigation and response to indications of unlawful, unauthorized, suspicious, or unusual activity.	MANDATORY	Fully Implemented	2.1.11
Establish and maintain baseline configurations and inventories of organizational systems (including hardware, software, firmware, and documentation) throughout the respective system development life cycles.	5	Fully Implemented	2.1.12
Establish and enforce security configuration settings for information technology products employed in organizational systems.	MANDATORY	Fully Implemented	2.1.13
Define, document, approve, and enforce physical and logical access restrictions associated with changes to organizational systems.	5	Fully Implemented	2.1.14
Employ the principle of least functionality by configuring organizational systems to provide only essential capabilities.	MANDATORY	Fully Implemented	2.1.15
Restrict, disable, or prevent the use of nonessential programs, functions, ports, protocols, and services.	MANDATORY	Fully Implemented	2.1.16
Apply deny-by-exception (blacklisting) policy to prevent the use of unauthorized software or deny-all, permit-by-exception (whitelisting) policy to allow the execution of authorized software.	MANDATORY	Fully Implemented	2.1.17
Identify system users, processes acting on behalf of users, and devices.	MANDATORY	Fully Implemented	2.1.18
Authenticate (or verify) the identities of users, processes, or devices, as a prerequisite to allowing access to organizational systems.	5	Fully Implemented	2.1.19

Use multifactor authentication (MFA) for local and network access to privileged accounts and for network access to non-privileged accounts.	MANDATORY	Fully Implemented	2.1.20
Store and transmit only cryptographically-protected passwords.	MANDATORY	Fully Implemented	2.1.21
Establish an operational incident-handling capability for organizational systems that includes preparation, detection, analysis, containment, recovery, and user response activities.	MANDATORY	Fully Implemented	2.1.22
Track, document, and report incidents to designated officials and/or authorities both internal and external to the organization.	MANDATORY	Fully Implemented	2.1.23
Provide controls on the tools, techniques, mechanisms, and personnel used to conduct system maintenance.	5	Fully Implemented	2.1.24
Require multifactor authentication to establish nonlocal maintenance sessions via external network connections and terminate such connections when nonlocal maintenance is complete.	5	Fully Implemented	2.1.25
Control the use of removable media on system components.	MANDATORY	Fully Implemented	2.1.26
Ensure that organizational systems containing sensitive data are protected during and after personnel actions such as terminations and transfers.	MANDATORY	Fully Implemented	2.1.27
Limit physical access to organizational systems, equipment, and the respective operating environments to authorized individuals.	MANDATORY	Fully Implemented	2.1.28
Protect and monitor the physical facility and support infrastructure for organizational systems.	5	Fully Implemented	2.1.29
Scan for vulnerabilities in organizational systems and applications periodically and when new vulnerabilities affecting those systems and applications are identified.	MANDATORY	Fully Implemented	2.1.30
Periodically assess the security controls in organizational systems to determine if the controls are effective in their application.	5	Fully Implemented	2.1.31
Monitor security controls on an ongoing basis to ensure the continued effectiveness of the controls.	5	Fully Implemented	2.1.32
Monitor, control, and protect communications (i.e., information transmitted or received by organizational systems) at the external boundaries and key internal boundaries of organizational systems.	MANDATORY	Fully Implemented	2.1.33
Employ architectural designs, software development techniques, and systems engineering principles that promote effective information security within organizational systems.	MANDATORY	Fully Implemented	2.1.34
Implement subnetworks for publicly accessible system components that are physically or logically separated from internal networks.	MANDATORY	Fully Implemented	2.1.35
Deny network communications traffic by default and allow network communications traffic by exception (i.e., deny all, permit by exception).	MANDATORY	Fully Implemented	2.1.36
Protect the authenticity of communications sessions.	5	Fully Implemented	2.1.37
Identify, report, and correct system flaws in a timely manner.	MANDATORY	Fully Implemented	2.1.38

Provide protection from malicious code at designated locations within organizational systems.	MANDATORY	Fully Implemented	2.1.39
Monitor system security alerts and advisories and take action in response.	MANDATORY	Fully Implemented	2.1.40
Update malicious code protection mechanisms when new releases are available.	MANDATORY	Fully Implemented	2.1.41
Monitor organizational systems, including inbound and outbound communications traffic, to detect attacks and indicators of potential attacks.	MANDATORY	Fully Implemented	2.1.42
Employ the principle of least privilege, including for specific security functions and privileged accounts.	3	Fully Implemented	2.1.43
Ensure that the actions of individual system users can be uniquely traced to those users so they can be held accountable for their actions.	3	Fully Implemented	2.1.44
Perform maintenance on organizational systems.	3	Fully Implemented	2.1.45
Check media containing diagnostic and test programs for malicious code before the media are used in organizational systems.	3	Fully Implemented	2.1.46
Prohibit the use of portable storage devices when such devices have no identifiable owner.	3	Fully Implemented	2.1.47
Screen individuals prior to authorizing access to organizational systems containing sensitive data	3	Fully Implemented	2.1.48
Periodically assess the risk to organizational operations (including mission, functions, image, or reputation), organizational assets, and individuals, resulting from the operation of organizational systems and the associated processing, storage, or transmission of sensitive data.	3	Fully Implemented	2.1.49
Develop and implement plans of action designed to correct deficiencies and reduce or eliminate vulnerabilities in organizational systems.	3	Fully Implemented	2.1.50
Implement cryptographic mechanisms to prevent unauthorized disclosure of sensitive data during transmission unless otherwise protected by alternative physical safeguards.	3	Fully Implemented	2.1.51
Employ FIPS-validated cryptography when used to protect the confidentiality of sensitive data.	3	Fully Implemented	2.1.52
Perform periodic scans of organizational systems and real-time scans of files from external sources as files are downloaded, opened, or executed.	3	Fully Implemented	2.1.53
Identify unauthorized use of organizational systems	3	Fully Implemented	2.1.54
Separate the duties of individuals to reduce the risk of malevolent activity without collusion.	1	Fully Implemented	2.1.55
Prevent non-privileged users from executing privileged functions and capture the execution of such functions in audit logs.	1	Fully Implemented	2.1.56
Limit unsuccessful logon attempts.	1	Fully Implemented	2.1.57
Use session lock with pattern-hiding displays to prevent access and viewing of data after a period of inactivity.	1	Fully Implemented	2.1.58
Terminate (automatically) a user session after a defined condition.	1	Fully Implemented	2.1.59
Route remote access via managed access control points.	1	Fully Implemented	2.1.60

Verify and control/limit connections to and use of external systems.	1	Fully Implemented	2.1.61
Limit use of portable storage devices on external systems.	1	Fully Implemented	2.1.62
Provide security awareness training on recognizing and reporting potential indicators of insider threat.	1	Fully Implemented	2.1.63
Review and update logged events.	1	Fully Implemented	2.1.64
Alert in the event of an audit logging process failure.	1	Fully Implemented	2.1.65
Protect audit information and audit logging tools from unauthorized access, modification, and deletion.	1	Fully Implemented	2.1.66
Limit management of audit logging functionality to a subset of privileged users.	1	Fully Implemented	2.1.67
Analyze the security impact of changes prior to implementation.	1	Fully Implemented	2.1.68
Control and monitor user-installed software.	1	Fully Implemented	2.1.69
Enforce a minimum password complexity and change of characters when new passwords are created.	1	Fully Implemented	2.1.70
Allow temporary password use for system logons with an immediate change to a permanent password.	1	Fully Implemented	2.1.71
Obscure feedback of authentication information.	1	Fully Implemented	2.1.72
Test the organizational incident response capability.	1	Fully Implemented	2.1.73
Supervise the maintenance activities of maintenance personnel without required access authorization.	1	Fully Implemented	2.1.74
Protect the confidentiality of backup sensitive data at storage locations.	1	Fully Implemented	2.1.75
Escort visitors and monitor visitor activity.	1	Fully Implemented	2.1.76
Remediate vulnerabilities in accordance with risk assessments.	1	Fully Implemented	2.1.77
Terminate network connections associated with communications sessions at the end of the sessions or after a defined period of inactivity.	1	Fully Implemented	2.1.78
Establish and manage cryptographic keys for cryptography employed in organizational systems.	1	Fully Implemented	2.1.79
Develop, document, and periodically update system security plans that describe system boundaries, system environments of operation, how security requirements are implemented, and the relationships with or connections to other systems.	MANDATORY	Fully Implemented	2.1.80
Total Possible Points	**116**	**0**	**0**

Attachment F

POA&M Tracker

System Name:	OmniBallot

POA&M ID	Type	Governing Control	Weakness Description	Source Identifying Weakness	Original Detection Date	Scheduled Completion Date	POC	Resources Required	Vendor Dependency	Last Vendor Check-in Date	Vendor Dependent Product Name	Planned Milestones	Milestone Changes	Status Date	Original Risk Rating	Adjusted Risk Rating	Mitigating Factors	Comments
2.5.5	WA	OWASP ASVS	Verify that if an authentication factor is changed or replaced, that the user is notified of this event. Changes to a users MFA authenticator does not inform the user.	Self-assessment	11/1/2023	TBD	Democracy Live Development	Democracy Live	No		Cognito				Low		Small user population.	
14.2.3	WA	OWASP ASVS	Verify that if application assets, such as JavaScript libraries, CSS or web fonts, are hosted externally on a Content Delivery Network (CDN) or external provider, Subresource Integrity (SRI) is used to validate the integrity of the asset. OmniBallot does not use SRI integrity validation.	Self-assessment	11/1/2023	TBD	Democracy Live Development	Democracy Live	No		N/A				Low		ReCAPTCHA and Google Fonts do not use SRI checks. However, they are very commonly used and a compromise of them would be difficult at best.	

Appendix 1

OmniBallot Accessibility Conformance Report

(based on VPAT 2.4 Rev)

Democracy Live Accessibility Conformance Report

WCAG Edition

(Based on VPAT® Version 2.4Rev)

Name of Product/Version: OmniBallot v10

Report Date: November 2023

Product Description: Online balloting software

Contact Information: support@democracylive.com

Notes:

Evaluation Methods Used:

Testing based on knowledge of product and functionality. Testing included use of WebAIM, Developer Accessibility tools, Screen Readers, and keyboard to validate each requirement. Testing included MacOS VoiceOver, MacOS Voice Control, Windows NVDA, and Windows JAWS, on Edge, Chrome, Firefox, and Safari web browsers.

Applicable Standards/Guidelines

This report covers the degree of conformance for the following accessibility standard/guidelines:

Standard/Guideline	Included In Report
Web Content Accessibility Guidelines 2.0	Level A (Yes) Level AA (Yes)

Standard/Guideline	Included In Report
	Level AAA (Yes)
Web Content Accessibility Guidelines 2.1	Level A (Yes) Level AA (Yes) Level AAA (Yes)

Terms

The terms used in the Conformance Level information are defined as follows:

- **Supports**: The functionality of the product has at least one method that meets the criterion without known defects or meets with equivalent facilitation.
- **Partially Supports**: Some functionality of the product does not meet the criterion.
- **Does Not Support**: The majority of product functionality does not meet the criterion.
- **Not Applicable**: The criterion is not relevant to the product.
- **Not Evaluated**: The product has not been evaluated against the criterion. This can be used only in WCAG 2.0 Level AAA.

WCAG 2.1 Report

Note: When reporting on conformance with the WCAG 2.x Success Criteria, they are scoped for full pages, complete processes, and accessibility-supported ways of using technology as documented in the WCAG 2.0 Conformance Requirements.

Table 1: Success Criteria, Level A

Notes:

Criteria	Conformance Level	Remarks and Explanations
1.1.1 Non-text Content (Level A)	Supports	Limited use of graphic content. Text alternatives provided for all graphics and icons when necessary.
1.2.1 Audio-only and Video-only (Prerecorded) (Level A)	Supports	No audio or video components
1.2.2 Captions (Prerecorded) (Level A)	Supports	No audio or video components
1.2.3 Audio Description or Media Alternative (Prerecorded) (Level A)	Supports	No audio or video components
1.3.1 Info and Relationships (Level A)	Supports	Headings, aria landmarks, aria labels and semantic markup are used according to best practices.
1.3.2 Meaningful Sequence (Level A)	Supports	Content programmatically follows logial sequence from top to bottom
1.3.3 Sensory Characteristics (Level A)	Supports	When colors are used, alternative icons or text are also included
1.4.1 Use of Color (Level A)	Supports	When colors are used, alternative icons or text are also included. Example: Warning text is accompanied by a graphic icon, bold typeface, the word warning. SS is used to change the visual representation of items with focus.
1.4.2 Audio Control (Level A)	Supports	No audio components
2.1.1 Keyboard (Level A)	Supports	All elements can be accessed via keyboard
2.1.2 No Keyboard Trap (Level A)	Supports	No components trap keyboard focus
2.1.4 Character Key Shortcuts (Level A 2.1 only)	Supports	No keyboard shortcuts are used
2.2.1 Timing Adjustable (Level A)	Supports	No time limits in application
2.2.2 Pause, Stop, Hide (Level A)	Supports	No moving, scrolling, or blinking content
2.3.1 Three Flashes or Below Threshold (Level A)	Supports	No flashing content
2.4.1 Bypass Blocks (Level A)	Supports	Skip to main content, skip to review, and skip to bottom links provided when applicable/content repeats
2.4.2 Page Titled (Level A)	Supports	All pages have a descriptive title
2.4.3 Focus Order (Level A)	Supports	Focus order follows visual page order

Criteria	Conformance Level	Remarks and Explanations
2.4.4 Link Purpose (In Context) (Level A)	Supports	Links include context (no "click here" links). Buttons also include additional context ("Continue to ___")
2.5.1 Pointer Gestures (Level A 2.1 only)	Supports	No multipoint or guesture functionality
2.5.2 Pointer Cancellation (Level A 2.1 only)	Supports	No drag and drop or pointer down specific functionality in application. All interactivity can be used with single pointer.
2.5.3 Label in Name (Level A 2.1 only)	Supports	A warning icon includes warning text in label. Continue buttons use aria-labelledby to use visible label
2.5.4 Motion Actuation (Level A 2.1 only)	Supports	No motion actuated components
3.1.1 Language of Page (Level A)	Supports	Lang attribute is applied to html element in code
3.2.1 On Focus (Level A)	Supports	Focus is always shown but does not change context or content
3.2.2 On Input (Level A)	Supports	Changing any input value does not change focus or context
3.3.1 Error Identification (Level A)	Supports	Errors are clearly identified using color, style changes, and icons when applicable.
3.3.2 Labels or Instructions (Level A)	Supports	All user inputs include labels and descriptions if additional context is necessary (such as format)
4.1.1 Parsing (Level A)	Supports	Application has valid HTML, uses unique IDS, and has hierarchal structure
4.1.2 Name, Role, Value (Level A)	Supports	All elements use semantic markup or define role, aria-label (and additional aria-* attributes) when custom components are defined

Table 2: Success Criteria, Level AA

Notes:

Criteria	Conformance Level	Remarks and Explanations
1.2.4 Captions (Live) (Level AA)	Supports	No live audio content
1.2.5 Audio Description (Prerecorded) (Level AA)	Supports	No prerecorded audio content

Criteria	Conformance Level	Remarks and Explanations
1.3.4 Orientation (Level AA 2.1 only)	Supports	All functionality works regardless of orientation. Note: Signature can be performed in either orientation, however, signature component currently functions *better* when rotated. Development scheduled to make signature component rotation independent.
1.3.5 Identify Input Purpose (Level AA 2.1 only)	Supports	All inputs either use semantic markup for native input descriptions, or use roles to accurately identify input (ie checkbox for an on/off type of component)
1.4.3 Contrast (Minimum) (Level AA)	Supports	All contrast meets 7:1 ratio
1.4.4 Resize text (Level AA)	Supports	All text can be resized to 300% without loss of functionality
1.4.5 Images of Text (Level AA)	Supports	No images of text used
1.4.10 Reflow (Level AA 2.1 only)	Supports	Application is fully responsive and reflows to only scroll vertically
1.4.11 Non-text Contrast (Level AA 2.1 only)	Supports	User interface components and states meet a 3.54:1 ratio
1.4.12 Text Spacing (Level AA 2.1 only)	Supports	Line height and spacing is defined in CSS and can be adjusted. No letter or word spacing are defined (can be adjusted as needed).
1.4.13 Content on Hover or Focus (Level AA 2.1 only)	Supports	No use of hover or dismissible content
2.4.5 Multiple Ways (Level AA)	Supports	Simple site structure takes user through voter flow. Links are provided in menu and on home page. Content/pages are not deeply nested
2.4.6 Headings and Labels (Level AA)	Supports	Headings are used on every page (with sub headings when necessary). Labels are used on all form components.
2.4.7 Focus Visible (Level AA)	Supports	Focus is visually indicated with a clear outline using CSS
3.1.2 Language of Parts (Level AA)	Supports	Interface is fully translatable so all components can be translated as needed.
3.2.3 Consistent Navigation (Level AA)	Supports	All pages have menus at the top, action items near the bottom, and a footer at the end of the page for navigation providing consistent navigation.

Criteria	Conformance Level	Remarks and Explanations
3.2.4 Consistent Identification (Level AA)	Supports	Navigation between pages always use buttons. Buttons always include consistent identification of where the button takes you. All icons use a text alternative that matches the visible text label.
3.3.3 Error Suggestion (Level AA)	Supports	All form fields include required attributes when required. If format is invalid an invalid label is provided. Form errors include a link to focus on the component with an error.
3.3.4 Error Prevention (Legal, Financial, Data) (Level AA)	Supports	All user input is accompanied by inline error identification, a/or review screen, or review capability depending on context. Users can make changes before final submission action.
4.1.3 Status Messages (Level AA 2.1 only)	Supports	Arial-live regions used to portray updated content.

Table 3: Success Criteria, Level AAA

Notes:

Criteria	Conformance Level	Remarks and Explanations
1.2.6 Sign Language (Prerecorded) (Level AAA)	Support	No video content
1.2.7 Extended Audio Description (Prerecorded) (Level AAA)	Supports	No video content
1.2.8 Media Alternative (Prerecorded) (Level AAA)	Supports	No video content
1.2.9 Audio-only (Live) (Level AAA)	Supports	No audio content
1.3.6 Identify Purpose (Level AAA 2.1 only)	Supports	Aria landmarks and semantic markup used for page structure
1.4.6 Contrast (Enhanced) (Level AAA)	Supports	All text meets a 7:1 contrast ratio
1.4.7 Low or No Background Audio (Level AAA)	Supports	No background audio
1.4.8 Visual Presentation (Level AAA)	Supports	No styles are hard coded (CSS used for all visual presentation) allowing for user overrides. Page width is more than 80 characters and is fully responsive. Line spacing is 1.5 by default and can be increased. Text can

Criteria	Conformance Level	Remarks and Explanations
		be increased to 300% without impacting functionality.
1.4.9 Images of Text (No Exception) (Level AAA)	Supports	No images of text used
2.1.3 Keyboard (No Exception) (Level AAA)	Supports	All functionality is available through common keyboard controls (tab, shift tab, space/enter)
2.2.3 No Timing (Level AAA)	Supports	No timing restrictions
2.2.4 Interruptions (Level AAA)	Supports	Application does not interrupt user (no alerts for example). Warnings are inline and can be moved passed or addressed on user's timeframe.
2.2.5 Re-authenticating (Level AAA)	Supports	User sessions do not expire
2.2.6 Timeouts (Level AAA 2.1 only)	Supports	There are no timeouts
2.3.2 Three Flashes (Level AAA)	Supports	No flashing content
2.3.3 Animation from Interactions (Level AAA 2.1 only)	Supports	No animations used
2.4.8 Location (Level AAA)	Supports	Simple site architecture does not provide deeply nested logic. Progress through user flows not possible due to complicated nature of business logic. With three steps A, B, C: Step A may cause Step B to be skipped. Providing "Step 1 of 3" is then misleading if the user is skipped directly from 1 to 3. Final step is always indicated by "End Session" rather than "Continue"
2.4.9 Link Purpose (Link Only) (Level AAA)	Supports	Links always include full text. For example: "View more information about absentee ballot requests" rather than "click here"
2.4.10 Section Headings (Level AAA)	Supports	All pages can only have one h1 tag. H2 and h3 tags are used to identify sub sections.
2.5.5 Target Size (Level AAA 2.1 only)	Supports	All targets are larger than 44x44px (menus, buttons, checkboxes, etc)
2.5.6 Concurrent Input Mechanisms (Level AAA 2.1 only)	Supports	Any input that sends common key commands to the web browser may be used (no custom input handling logic)

Criteria	Conformance Level	Remarks and Explanations
3.1.3 Unusual Words (Level AAA)	Supports	Clear and simple wording used by default. All text is customizable by customer as needed. No jargon or slang used.
3.1.4 Abbreviations (Level AAA)	Supports	No abbreviations used
3.1.5 Reading Level (Level AAA)	Supports	Clear and simple wording used by default. All text is understandable at a lower secondary level. Note: This applies to default text. All text is customizable by customers and is the customer responsibility to maintain this standard. Ballot text is also customer responsibility to keep simple to understand.
3.1.6 Pronunciation (Level AAA)	Supports	No complex words requiring pronunciation are used by default.
3.2.5 Change on Request (Level AAA)	Supports	No pop up windows, redirects, or automatic updates are performed. All changes within the application are triggered by direct user interaction.
3.3.5 Help (Level AAA)	Supports	Context help is provided on each page before the related content (ex: ballot marking instructions before ballot, ballot review instructions before review)
3.3.6 Error Prevention (All) (Level AAA)	Supports	All user input is accompanied by inline error identification, a/or review screen, or review capability depending on context. Users can make changes before final submission action.

Appendix 2

University of Washington Accessibility Test Report

ADII Accessible Design & Innovative Inclusion

6912 220th Street SW Suite 105
Mountlake Terrace, WA 98043

October 31, 2023

Subject: OmniBallot by Democracy Live

To Whom It May Concern,

Since 2018, Accessible Design and Innovative Inclusion (or ADII) has assisted public agencies and private businesses in the United States in making their facilities, programs, activities, goods and services more accessible to people with disabilities. With specific expertise in architectural and website accessibility, ADII has partnered with a variety of organizations, such as museums, libraries, schools, parks, and zoos, to conduct evaluations, remediation, staff training, and technical assistance. ADII is comprised of specialists with over 100 years of combined experience (as a team) who are trained in disability rights laws, including the Americans with Disabilities Act (ADA), disability access issues, inclusive design and universal design principles of built environment, employment accommodations, and targeted public engagement with diverse and disability community. The ADII Director is a certified ADA coordinator and a Certified Rehabilitation Counselor. ADII is a program of the Center for Continuing Education in Rehabilitation (CCER), which is housed within the University of Washington, Department of Rehabilitation Medicine. CCER has done a significant amount of work within the disability community for over 40 years, promoting access and inclusion for all through education, collaboration, and connection to resources.

On October 30 and 31, 2023, ADII's human evaluation team reviewed the OmniBallot by Democracy Live, and concluded that it met user acceptance and Section 508 conformity for voters living with disabilities. The review was conducted by individuals who use assistive technology for everyday computing tasks.

Thank you for your commitment to accessible websites and digital products. Thank you, also, for this opportunity to report on OmniBallot.

Sincerely,

Eva L. de Leon

Eva L. De Leon, MA, CRC, ADAC
ADII Director
Assistant Director of Programs, CCER
The University of Washington, Department of Rehabilitation Medicine

UWCTDS
Center for Technology and Disability Studies
university of washington
(206) 685-4181-V
(206) 616-1396 TTY • (206) 543-4779 Fax
Box 357920 Seattle, WA 98195-7920

Evaluation Date: 5/6/2019

Web Accessibility Testing and Report hours: 4 hours @ $125/hr

Website Evaluated: https://sites.omniballot.us/UW/app/home

Testing Technology and Evaluative Tools Used

The website was tested on Windows 10 using JAWS 2018 and Chrome browser. While this is a typical assistive technology configuration, it should be noted that not all operating systems, screen readers, or browsers are the same. Results may vary somewhat between platforms and circumstances, as with any internet technology.

The following content and tools were used to test for WCAG 2.0AA and Section 508 compliance:

- Web Content Accessibility Guidelines: http://www.w3.org/rlVWCAG20/
- WAVE Browser Extensions (For Chrome and Firefox): http://wave.webaim.org/extension/
- WebAIM color checker: http://wave.webaim.org/extension/
- WebAIM Web Accessibility Checklist (WCAG 2.0 Guidelines): webaim.org/standards/wcag/checklist/
- Markup Validation Service: http://validator.w3.org/
- US Dept of Health & Human Services: HTML 508 Compliance Checklist: https://www.hhs.gov/web/section-508/making-files-accessible/checklist/html/index.html
- ICT refresh: https://www.access-board.gov/guidelines-and-standards/communications-and-it/about-the-ict-refresh/final-rule/single-file-version#E205-content

Executive Summary

Overall, the site is very accessible in its current form. Keyboard access was effective in all web pages tested. Screen reader access was also strong. Screen reader navigation was mostly consistent and easy to understand. When marking ballots the state of the checkboxes was announced before and after making a selection. One recommendation is made below to improve usability.

Due to the fact that WCAG 2.0 AA Success Criteria are more explicit than the current 508 Standards, focus was directed to identifying compliance with WCAG 2.0 AA Success Criteria. https://www.access-board.gov/guidelines-and-standards/communications-and-it/about-the-ict-refresh/background/comparison-table-of-wcag2-to-existing-508-standards
Unless mentioned specifically below, WCAG 2.0 and Section 508 compliance was found to be met.

General Evaluation and Commentary

1.3.2 Meaningful Sequence

When reading the Sierra County Elections Department PDF with Jaws, the reading order does not align with the visual order in which information is presented. The line "I declare that:" is read before "Voter's Declaration/Oath of Voter". Taking a look in Adobe Acrobat Pro, the reading order is incorrect for these two lines although the document is tagged in the correct order. Correcting the reading order and then running the auto tag utility fixes this issue.

1.4.3 Contrast

There are several different color combinations on this website that did not meet Level AA Criterion for contrast. In order to comply with WCAG 2.0 AA criterion one of the following conditions must be met:

- Text and images of text have a contrast ratio of at least 4.5:1.
- Large text - at least 18 point (typically 24px) or 14 point (typically 18.66px) and bold - has a contrast ratio of at least 3:1.

On most of the pages the buttons have a background color of #007fb6 with white text color #eeeeee which has a contrast ratio of 3.84:1. The font size was found to be typically 16px which is equivalent to 12 pt font.

The links such as "Skip to Bottom" in the County Voter Information Guide or "How to Vote" under Additional Info are in #007fb6 with a white background #eeeeee, contrast ratio 3.84:1 which do not provide adequate contrast for the 16px font. On mouse hover the links are even lighter # 00a3e5, contrast ratio of 2.46:1.

Recommendation is to change the background color to increase the contrast ratio to 4.5:1 or increase the font size to 18 point (typically 24px) or 14 point (typically 18.66px) and bold.

2.4.1 Bypass Blocks

The "Skip to Bottom" links in the Reference Ballot do not function.

2.4.2 Page Titled

Web pages are not titled <title></title>

2.4.6 Headings and Labels

Inconsistent use of Headings in the Additional Info- How to Vote page. There are two "How to Vote" level 2 headings. Also one instance of the heading "How to Vote" is in all caps whereas the other is not. Towards the bottom of the page "Spanish Language Assistance" is heading level 1 which is inconsistent with the Heading structure above.

2.4.7 Focus Visible

On the Print Your Choices page of the Ballot Marking Application, the "Print Selection" button does not change to yellow on hover or on focus with keyboard navigation. The blue focus rectangle does not provide enough contrast to the blue button background color.

3.2.4 Consistent Identification

The "Continue" buttons at the bottom of the Welcome Voters page are recognized as links by Jaws whereas the "Continue" buttons on other pages are recognized as buttons. Recommendation to change the links to buttons on the Welcome page to improve consistency.

3.3.2 Labels or Instructions

In the Ballot Marking Instructions page the instructions state to "click on the following link" however there are no links following. "Go Back" and "Continue to Ballot" are both buttons. Also in the tab order and using Jaws, the next object the user comes to is the "Go Back" button. Recommendation to provide more specific instructions for improved understanding of next steps.

4.1.1 Parsing

The W3 Validator tool was used for these findings: http://validator.w3.org/
On all the pages, the below error was identified. This is also mentioned under section 2.4.2.

Error: Element `title` must not be empty.

From line 5, column 10; to line 5, column 17

```
↩ <title></title>↩ <ba
```

On the live ballot application page the following errors were identified. Despite these errors, the page was completely accessible using Jaws indicating that the page has appropriate tag structure.

Error: Forbidden code point U+0085.

At line 89, column 7902

```
="["+a+"]",s="…",c=RegExp("^"
```

Error: Forbidden code point U+007f.

At line 724, column 6200

```
e","\t":"Tab","□":"Delete",""
```

Error: Forbidden code point U+001b.

At line 724, column 6213

```
,"□":"Delete","":"Escape",Del
```

Error: Forbidden code point U+0090.

At line 724, column 6462

```
O:"/","`":"0","□":"NumLock"};j
```

Error: Forbidden code point U+0085.

At line 773, column 8977

```
="["+a+"]",c="…",u=RegExp("^"
```

Usability

Hidden headings are used for Language and Main Menu to provide information and quick navigation access to screen reader users. For consistency, consider adding a similar heading to the list of links for Instructions, Candidate Statements, Measures, etc.

Appendix 3

See TRADE SECRETS Envelope

Disaster Recovery Plan

(aka Emergency Response Plan)

CONFIDENTIAL

Appendix 4

See TRADE SECRETS Envelope

Synack 2023 Summary Report

CONFIDENTIAL

Appendix 5

See TRADE SECRETS Envelope

StateRAMP Requirements

CONFINDENTIAL

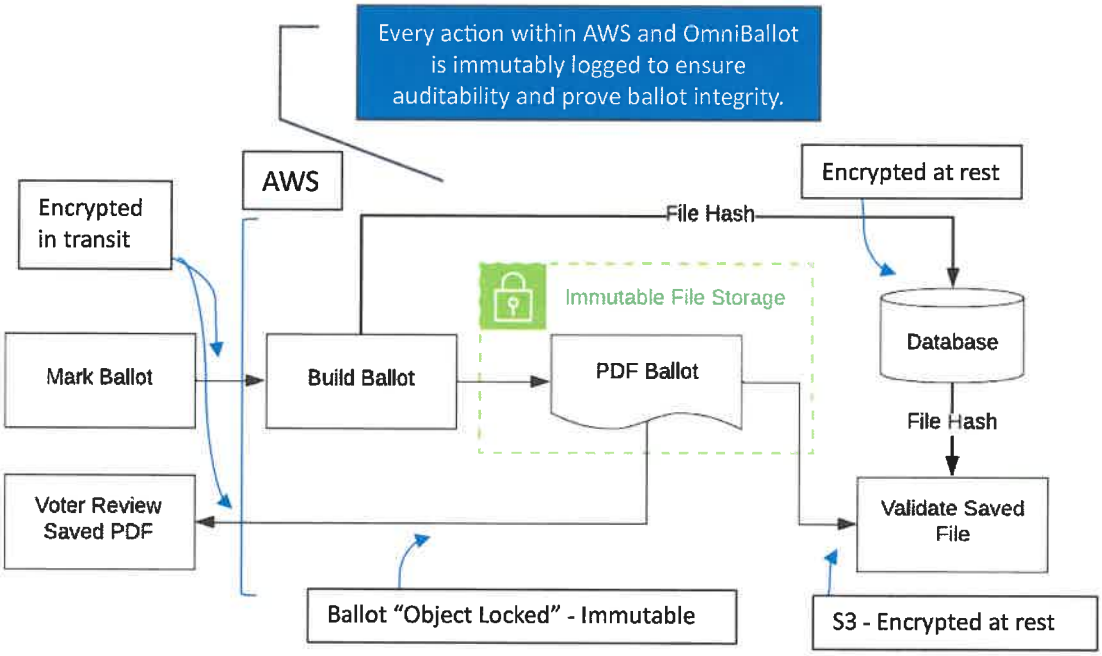
Appendix 6

Diagram: Auditable Proof of Secure Ballot Transmission

OMNIBALLOT

OmniBallot: A Voter's Voted Ballot Auditable Proof

After the voter marking process is completed, the voter's ballot selections are securely encrypted and sent to the OmniBallot backend (shown in the below diagram "Build Ballot") where they are populated onto an official ballot PDF. The PDF is stored in immutable file storage using AWS S3 Object Lock (it is encrypted at rest). A hash of the PDF file is calculated prior to saving the file and is then validated after saving to ensure the file was written successfully and without modification. This hash is a unique digital signature (like a fingerprint) of the file and is stored in the database (the database is fully encrypted). Any time OmniBallot loads the ballot file, the file is validated against the known hash as a secondary level of assurance the file has not been modified (an AWS insider attack prevention control). A pre-signed URL is generated with a 5-minute expiration which allows the voter to directly view the file **that has already been immutably saved**. At no time during this process (or within any OmniBallot function) can network traffic or "server" (AWS Fargate) services be accessed, viewed, monitored, or modified. The voter's ballot is completely hidden and unknown to anyone.



After confirming the saved ballot represents their selections, the voter can complete the electronic ballot submission. If a concern about the integrity of the system or a voter's ballot is raised, the following data points provide auditable proof the ballot has not been modified:

1. A creation timestamp and hash of the original ballot submission
2. A timestamp when the voter approved the ballot stored in object lock
3. A timestamp, hash, and second copy of the voter's ballot in Verifier (the voter was also given an opportunity to view this ballot)
4. A timestamp, user, and IP address of the EO who downloaded the voter's ballot

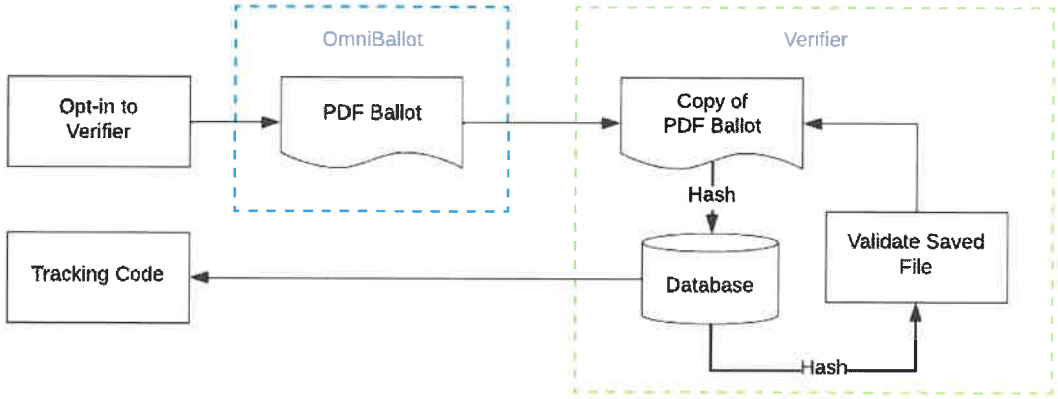
5. A timestamp and hash of the ballot the EO printed that has been uploaded into Verifier
6. CloudTrail logs proving
    a. No database access has occurred
    b. No security policies have been modified
    c. Object Lock has remained enabled
    d. No direct ballot access has occurred on the server
7. A copy of all logs have been sent (automatically) and are monitored by a third party

Additional OmniBallot supporting infrastructure assurances to ensure ballots remain private and unchanged:

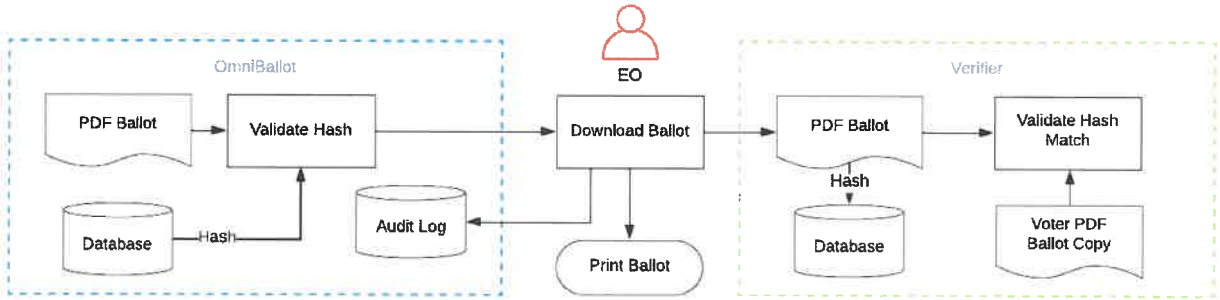
1. **CloudTrail** is an immutable audit log that tracks every action performed within AWS. Any changes to, or attempts to change, the following controls will be immutably stored and delivered to a 3rd party Managed Detection and Response (MDR) provider.
2. **Read Only Mode** is enabled during an election to prevent modification to any infrastructure elements. If an administrator needs access to perform maintenance or perform an urgent task, an access request is submitted. Logins to the infrastructure will trigger a notification to customers (Stakeholders).
3. **Role Based Permissions** ensure server administrators or developers are not able to access files within the ballot storage location during troubleshooting or maintenance windows.
4. **Service Control Policies** at the Organization level prevent modification to permissions models and Object Lock settings.
5. **S3 Object Lock** guarantees ballots are immutably stored with all direct access (or attempted access) logged.
6. **S3 Sensitive File** access, view, delete, and modify is prevented.
7. **EO Action Audit Log** stores all ballot access to an application audit log.
8. **Strict Database Control** requires access through Systems Manager. All database sessions are logged to CloudTrail and are prevented (or in the case of troubleshooting) during an election (Read Only Mode).
9. **GuardDuty** monitors the enter enterprise for anomalous behavior including modification of the above services and settings. Findings generate notifications to both internal resources and Democracy Live's Managed Detection and Response (MDR) service (24 x7).
10. **MDR Account Monitoring** is a 24x7x365 service performed by a third-party. AWS logs and events are automatically forwarded to a centralized security incident and event correlator (SIEM) which captures, alerts, and notifies upon detection of suspicious and anomalous activity.

There are multiple protection and tracking tiers throughout this process that prove ballot integrity. Tier one, every action, and movement have some form of encryption to ensure the integrity of the ballot. Tier two, it is immutably locked to guarantee that the ballot can never be modified or deleted by anyone during its lifetime. Tier three, everything supporting the process is traced, monitored, audited (with immutable logs), and designed (service and file preventions) to prohibit both external and internal attacks.

Tier four. Voters can opt into an additional ballot verification process through another system called Verifier, which is an independently hosted solution. This system allows the voter to verify the ballot submitted to OmniBallot is the exact same ballot received and printed by their Elections Official (EO). When the voter opts-in to Verifier, a copy of their ballot is sent to Verifier and stored using immutable file storage (this is a different AWS account and file location than OmniBallot). The PDF hash is calculated and stored in a separate location as the PDF. The voter receives a tracking number which they can use to check the status of their submission (see below).



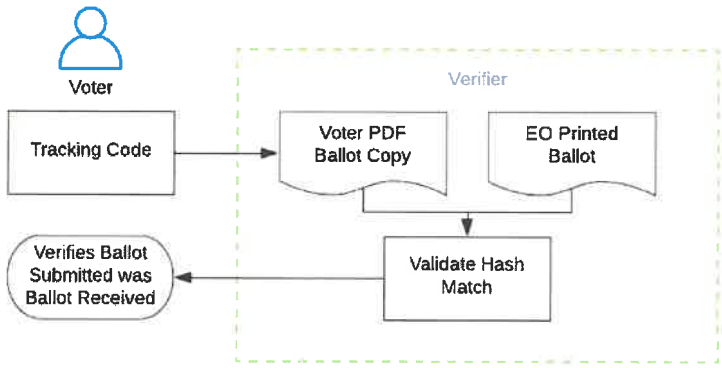
When the voter submits their ballot, the EO will receive an email notification about the new submission. The EO can log into OmniBallot to download and print the voter's ballot. Prior to downloading the Voter's ballot, OmniBallot performs an additional hash check against the file to confirm no changes have been made to the file while stored in Object Lock. If the hash of the file matches the stored hash, the ballot is downloaded and can be printed. Within OmniBallot, every action performed on a ballot submission is logged and clearly displayed with the submission information. Logged information includes EO's email, IP address, action performed, and a timestamp.



After printing the Voter's ballot, the EO logs into Verifier and uploads the PDF of the ballot they printed. When the ballot PDF is uploaded, Verifier calculates a file hash and the file in immutable

file storage. Verifier then compares the hash of the Voter ballot with the uploaded ballot and clearly identifies if the hashes do not match (indicating the files are not identical).

When the Voter returns to Verifier, they enter the tracking number to load their ballot submission. They can now see their ballot (and hash), and the ballot the EO printed (and hash). Verifier compares the hash values of the two files and displays a clear message that the hashes match and that the files are identical.



Appendix 7

See TRADE SECRETS Envelope

OWASP Application Security

CONFIDENTIAL

Appendix 8

See TRADE SECRETS Envelope

Security Requirements for Databases

CONFIDENTIAL