System Requirements Specification

for

CyberTool

Version 6.0 approved

Prepared by Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini, Troy Neubauer

Embry-Riddle Aeronautical University

March 14th, 2024

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini	09/29/23	Creation	V1.0
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini	10/18/23	Revision	V2.0
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini	11/16/23	Revision	V3.0
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini, Troy Neubauer	02/05/24	Revision	V4.0
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini, Troy Neubauer	03/05/24	Revision	V5.0
Nicolas Rodriguez, Jeremiah Webb, Olivia Meholic, Sarah Gleixner, Joseph Alesandrini, Troy Neubauer	04/14/24	Revision	V6.0

1. Introduction

The current method of finding the perfect cybersecurity tool for commercial or personal use includes personal research and inefficient comparisons. This requires unnecessary and cost-ineffective effort that can be easily mitigated by the right application.

CyberTool is research-based software designed to help users find the right cybersecurity tools. Its goal is to make the search for security tools easy and quick. By using this tool, users (whether individuals or large companies), can find the best cybersecurity solutions for their needs. This aligns with our corporate goal of making cybersecurity accessible and understandable for everyone. It is intended for long-term commercial and personal use as it is maintainable and easily updated through administrative access.

1.1. Purpose

The purpose of this document is to define all requirements associated with the CyberTool website. It includes requirements for the user interface, data processing, authentication, and interactions between our system and other systems within the AWS architecture.

This document provides the most updated documentation of the CyberTool system as of 04/14/2024.

1.2. Document Conventions

This SRS will be broken into numerical sections with each title bolded. The glossary will contain all of the names and abbreviations of the third-party software and AWS services that are utilized.

1.3. Intended Audience and Reading Suggestions

This Software Requirements Document for CyberTool is designed for all readers. To effectively build the tool, developers should look into the "Front-End" and "Back-End" sections for technical insights. Project Managers overseeing the tool's progression should review the entire document to ensure everything is on track. Marketing Staff will benefit from the "Purpose" and "Product Scope" sections, which highlight the tool's main features. Users can learn about what the tool offers by reading the "Purpose" section. Testers should focus on the technical requirements to ensure the tool works as planned. Lastly, Documentation Writers creating guides or other materials should read the entire SRS, with special attention to the "Glossary". For a complete understanding, all readers are advised to start with the "Purpose" section and then move to sections most relevant to their role.

Section 3.1 details the User Interfaces and what functionality is accessible to the user to navigate the CyberTool application. Section 3.4 explains our communication protocols using the AWS architecture. Section 4 describes the usage of specific AWS systems such as Dynamo DB, Amazon Cognito, and Amazon API Gateway in our application.

1.4. Product Scope

This is version 6.0 of CyberTool, which is a public website that will enable users to filter through a variety of cybersecurity tools depending on the needs of the user (commercial, personal, etc.). This website will be hosted on an Amazon Web Services (AWS) server including all necessary architecture for the product. Among this architecture, there will be a database containing the cybersecurity products and the key values that were determined during research of said products. Recommendations will be generated from the database based on user preferences/requirements. Upon user request, a comparative report will be generated containing prospective tools and/or tools they already have.

1.5. References

CyberTool has extensively used AWS documentation to assist in creating requirements.
The AWS Documentation can be found at: https://docs.aws.amazon.com/

2. Overall Description

Section 2 details the product perspective (2.1), product functions (2.2), user classes (2.3), operating environment (2.4), design and implementation constraints (2.5), user documentation (2.6), and assumptions and dependencies (2.7).

2.1. Product Perspective

CyberTool is software developed to assist users in selecting appropriate cybersecurity tools. It operates by interfacing with a database hosted on AWS, which contains detailed information about various cybersecurity products. The tool then provides recommendations based on user-specific requirements. This information flow is displayed in Figure 2.1 below.

2.2. Product Functions

User Preferences/Requirements: Users input their specific needs.

- Database on AWS: The tool interfaces with a database hosted on AWS containing information about various cybersecurity products.
- Recommendations: Based on the user's preferences and the data from the database, the tool provides recommendations.
- Comparative Report: Upon user request, a report comparing prospective tools and/or tools they already have is generated.

Filter Cybersecurity Tools: Users can filter through cybersecurity tools.

- Commercial: Tools for commercial use.
- Personal: Tools for personal use

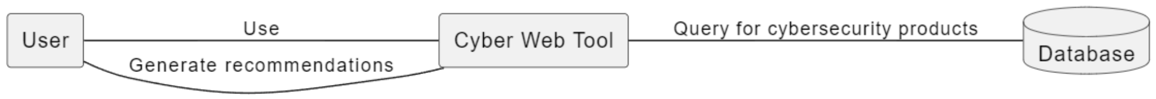


Figure 1 for 2.1 Product Perspective

Figure 1 presents a high-level perspective between entities in the product. Users research tools and can upload these tools via the Cyber Tool website itself, and the website queries the DynamoDB database for tool data and recommendations.

2.3. Operating Environment

The entire framework will operate in the AWS environment including the hosting of the website. As a result, users with a web browser (see versions in 2.7) will be able to interact with the product, CyberTool.

2.4. Design and Implementation Constraints

2.4.1. Design Constraints

- System can only be hosted by Amazon Web Services
- System can only be accessible by web browsers (See versions in 2.7).

2.4.2 Implementation Constraints

- System Speed is dependent on the quality of service paid to AWS.

2.5. User Documentation

2.5.1. A system architecture diagram will be provided on GitHub.

2.5.2. Lambda Documentation will be stored via a Github repository.

2.6. Assumptions and Dependencies

2.6.1. The developer must use Version 118.0.5993.72 (Official Build) (64-bit) Google Chrome Web Browser.

2.6.2. The developer may use Version 118.0.2 (64-bit) Firefox Web Browser.

2.6.3. The developer must have at least version 7.0.0 of npm.

2.6.4. The developer must have at least version 21.0.0 of nodejs.

2.6.5. The developer must have at least version 2.10.0 of aws-cli.

2.6.6. The developer must have internet access.

2.6.7. The developer must be on a computer running Windows 10.

2.6.8. The developer must have Administrative Access to AWS Dashboard.

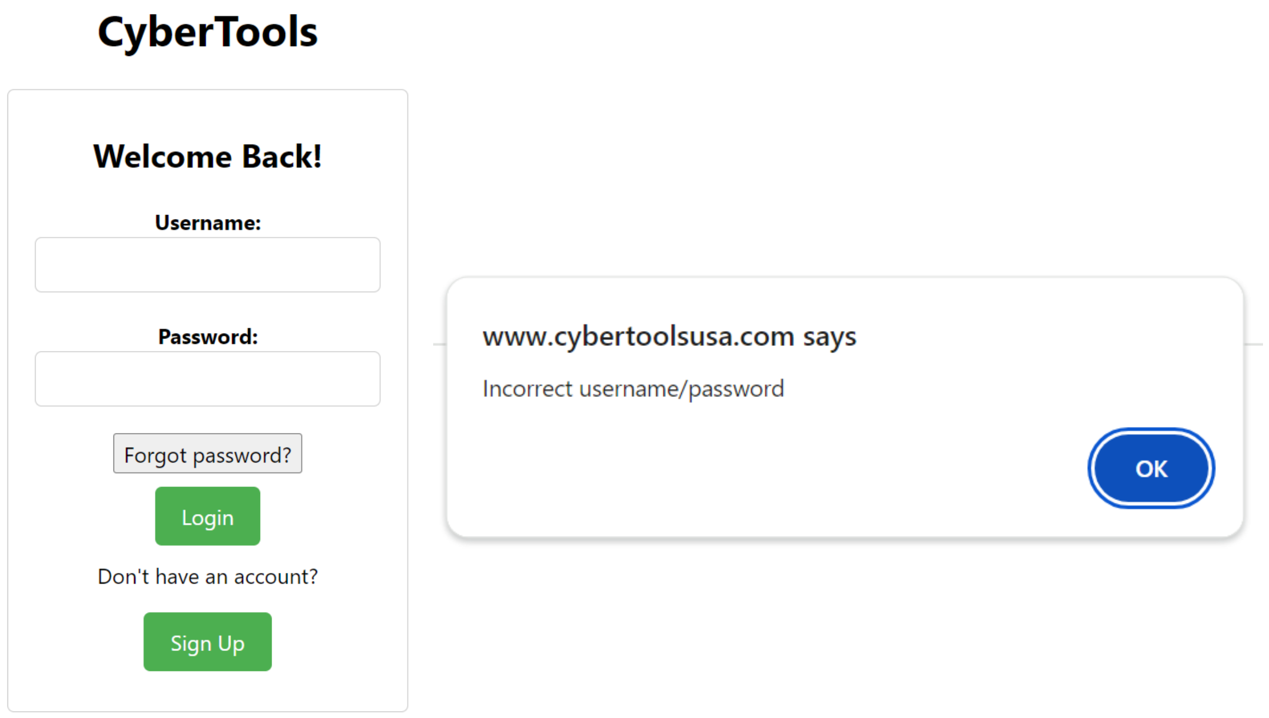
3. External Interface Requirements

Section 3 details how users interact with the system, along with how the system behaves internally through its communication protocols in Section 3.4.

3.1. User Interfaces

The user shall interface with CyberTool through our AWS-hosted web application. The following requirements detail how the user interface shall be displayed to offer application functionality to the user.

The user shall navigate CyberTool starting from the Login Page and then to the appropriate following pages according to their input.



www.cybertoolsusa.com says

Incorrect username/password

Figure 3.1.1 Login Page with Error Message

Login Page: A simple interface asking for a username and password, accompanied by a 'Forgot Password' link and a 'Sign up' button for new users.

3.1.1 Login Page: The system shall provide input fields for the username.

3.1.2 Login Page: The system shall provide input fields for the password.

3.1.3 Login Page: The system shall accept a string between 6 and 20 characters for the username field.

3.1.4 Login Page: The system shall accept a string between 6 and 20 characters for the password field.

3.1.5 Login Page: The system shall provide a 'Forgot password?' button beneath the password field.

3.1.6 Login Page: The system shall provide a 'Login' button.

3.1.7 Login Page: The systems shall provide a 'Sign Up' button underneath the 'Login' button.

3.1.8 Login Page: The systems shall provide a 'Continue as guest' button underneath the Signup button.

3.1.9 Login Page: The system shall allow authenticated users to take the survey.

3.1.10 Login Page: The system shall allow unauthenticated users to view CyberTool page.

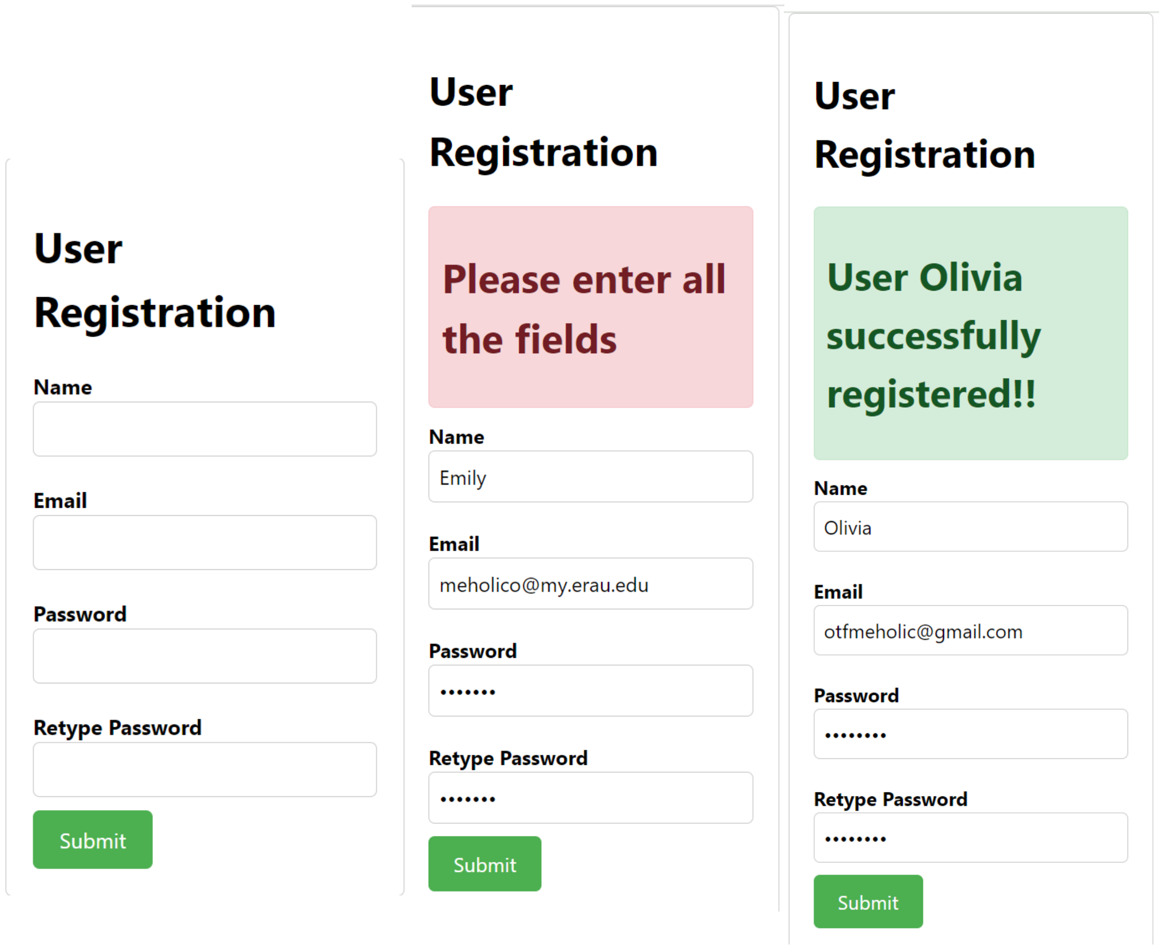
3.1.11 Login Page: The system shall allow authenticated users to be directed to survey page.

3.1.12 Login Page: The system shall allow authenticated users to view the main page.

3.1.13 Login Page: The system shall provide an error message if the inputs are not authenticated by Amazon Cognito.

3.1.14 Login Page: The system shall detect if a username is incorrect.

3.1.15 Login Page: The system shall detect if a password is incorrect.



User Registration

Name

Email

Password

Retype Password

Submit

User Registration

Please enter all the fields

Name: Emily

Email: meholico@my.erau.edu

Password: •••••••

Retype Password: •••••••

Submit

User Registration

User Olivia successfully registered!!

Name: Olivia

Email: otfmeholic@gmail.com

Password: ••••••••

Retype Password: ••••••••

Submit

Figure 3.1.2 Sign Up Page with Error and Success Messages

Signup Page: A simple interface asking for a name, email, username, and password.

3.1.16 Signup Page: The system shall provide a signup page

3.1.17 Signup Page: The system shall provide input fields for the username.

3.1.18 Signup Page: The system shall provide input fields for the password.

3.1.19 Signup Page: The system shall provide input fields for the ‘retype password’ field.

3.1.20 Signup Page: The system shall provide input fields for the email

3.1.21 Signup Page: The system shall accept a string between 6 and 20 characters for the username field.

3.1.22 Signup Page: The system shall accept a string between 6 and 20 characters for the password field.

3.1.23 Signup Page: The system shall accept a valid email for the ‘email’ input

3.1.24 Signup Page: The system shall validate that the ‘password’ and the ‘re-type password’ are the same

3.1.25 Signup Page: The system shall provide a ‘Sign Up’ button

3.1.26 Signup Page: The system shall provide a success message if the inputs are accepted by Amazon Cognito.

3.1.27 Signup Page: The system shall detect if an input field has not been filled out

3.1.28 Signup Page: The system shall provide an error message if the inputs are not accepted by Amazon Cognito.

3.1.29 Signup Page: The system shall detect if an email is already in use

3.1.30 Signup Page: The system shall detect if password is UTF8 characters

3.1.31 Signup Page: The system shall allow authenticated user to view main page

3.1.32 Signup Page: The system shall allow authenticated users to be directed to survey page

Survey Page: Multiple-choice questions designed to gather user needs regarding cybersecurity tools. Questions might revolve around user expertise, organization size, nature of the threat they face, and the technical stack they use.

3.1.33 Survey Page: The system shall provide check boxes.

3.1.34 Survey Page: The system shall provide questions for the user.

3.1.35 Survey Page: The system shall allow users to check off box.

3.1.36 Survey Page: The system shall detect if a box is not checked off.

3.1.37 Survey Page: The system shall allow authenticated users to be directed to the main page.

3.1.38 Survey Page: The system shall provide a ‘Continue’ button.

Tools Page: Displays a list of recommended cybersecurity tools based on the user’s answers, with brief descriptions.

3.1.39 Tools Page: The system shall provide a list of cybersecurity tools

3.1.40 Tools Page: The system shall provide the name of the cybersecurity tools

3.1.41 Tools Page: The system shall provide the version of the cybersecurity tools

3.1.42 Tools Page: The system shall provide the status of the cybersecurity tools

3.1.43 Tools Page: The system shall provide the launch date of the cybersecurity tools

3.1.44 Tools Page: The system shall provide a search bar on top of the cybersecurity tools

3.1.45 Tools Page: The system shall accept a string between 1 to 20 characters in search bar

3.1.46 Tools Page: The system shall show numbers of cybersecurity tools shown on page.

3.1.47 Tools Page: The system shall have pagination for cybersecurity tool display

3.1.48 Tools Page: The system shall provide left arrow button

3.1.49 Tools Page: The system shall provide right arrow button

3.1.50 Tools Page: The system shall move to next page when right arrow is pressed

3.1.51 Tools Page: The system shall move to previous page when left arrow is pressed

3.1.52 Tools Page: The system shall provide a filter checkbox menu

3.1.53 Tools Page: The system shall provide an aviation specific checkbox filter option

3.1.54 Tools Page: The system shall provide a toolbox checkbox filter option

3.1.55 Tools Page: The system shall provide a maturity level ranging from levels 1-4 as a checkbox filter option

3.1.56 Tools Page: The system shall provide an AI/ML use checkbox filter option

3.1.57 Tools Page: The system shall provide a tool function menu with checkboxes

3.1.58 Tools Page: The system shall provide a drop-down menu with different company options

3.1.59 Tools Page: The system shall filter cybersecurity tools based on filter user choice (see appendix item ---- for filter choices)

3.1.61 Tools Page: The system shall provide a link to retrieve old reports

3.1.62 Tools Page: The system shall allow the user to be directed to reports page

3.1.63 Tools Page: The system shall allow the user to download a comparative report

3.1.64 Tools Page: The system shall allow the user to click on tool

3.1.65 Tools Page: The system shall display tool information

3.1.65 Tools Page: The system shall

3.1.65 Tools Page: The system shall display tool information

3.1.65 Tools Page: The system shall display tool information

3.1.65 Tools Page: The system shall display tool information

3.1.65 Tools Page: The system shall display tool information

3.1.65 Tools Page: The system shall display tool information

Document Page: Displays a list of previously created reports with the option to view each report with an interactive link to the report

3.1.66 Document Page: The system shall display all documents previously made

3.1.67 Document Page: The system shall provide a ‘View Report’ link

3.1.68 Document Page: The system shall allow user to view a single report.

3.1.69 Document Page: The system shall allow a user to download a report via a presigned URL.

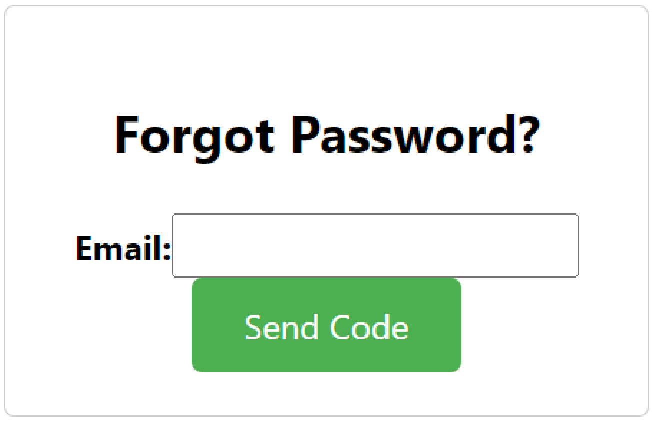


Figure 3.1.3 Forgot Password Page

Forgot Password Page: Allows user to enter email to reset password if a registered user

3.1.70 Forgot Password Page: The system shall provide input fields for the email

3.1.71 Forgot Password Page: The system shall provide a "Send Code" button

3.1.72 Forgot Password Page: The system shall provide an alert if the user inputs an invalid email.

3.1.73 Forgot Password Page: The system shall only redirect to the Reset Password Page if the user provides a valid email

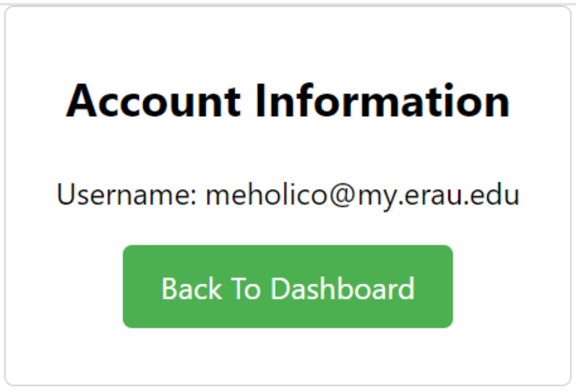


Figure 3.1.4 Account Information Page

Account Information Page: Allows users to see the username associated with their account

3.1.74 Account Information Page: The system shall display the current user's email in a text box once directed to this page.

3.1.75 Account Information Page: The system shall provide a "Back to Dashboard" button that directs the user back to the CyberTool dashboard when pressed.

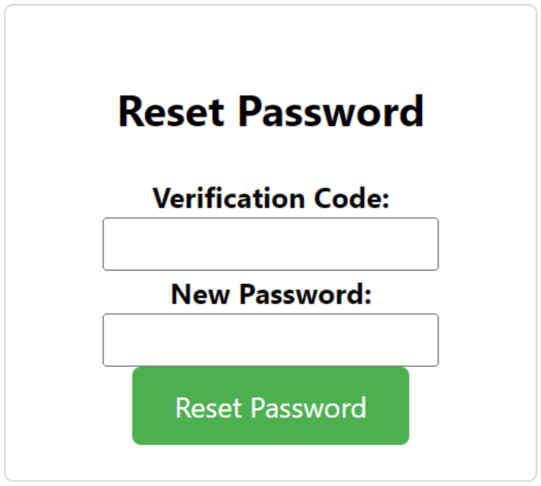


Figure 3.1.5 Reset Password Page

Reset Password Page: Allows users to enter the validation code and new password when resetting their password

3.1.76 Reset Password Page: The system shall provide input fields for the verification code.

3.1.77 Reset Password Page: The system shall provide input fields for the new password.

3.1.78 Reset Password Page: The system shall provide a "Reset Password" button that redirects the user back to the CyberTool dashboard page when pressed.

3.1.79 Reset Password Page: The system shall provide an error message if the verification code is not authenticated by Amazon Cognito.

Logical Characteristics:

Standards/Guidelines:

The GUI should ensure it is user-friendly.

Consistent font and color schemes throughout all its pages

Standard buttons should be used for login, signup, survey, and result pages.

Error messages should be clear, concise, and displayed in red font near the point of error.

3.2. Hardware Interfaces

This section discusses the hardware requirements for devices that can connect and use the CyberTool web application.

3.2.1. Users shall be able to access the CyberTool website via a laptop.

3.2.2. Users shall be able to access the CyberTool website via a personal laptop.

3.2.3. Users shall be able to access the CyberTool website via a desktop computer.

3.3. Software Interfaces

This section discusses the software interface requirements that CyberTool has for internal software functionality.

3.3.1. The system shall use AWS Amplify for FrontEnd Development.

3.3.2. The application shall be platform-independent, compatible with Windows 11, macOS, and Linux.

3.3.3. The system shall use DynamoDB Database for tool data.

3.3.4. The system shall use Amazon API Gateway for Communications between different AWS services.

3.4. Communication Interfaces

This section discusses the communication interface requirements that CyberTool has for dealing with information transfer between the React frontend and AWS backend.

3.4.1. All client-server communication within the CyberTool system shall exclusively utilize HTTPS (Hypertext Transfer Protocol Secure) over SSL/TLS (Secure Sockets Layer/Transport Layer Security).

3.4.2. The system shall use JSON for message formatting in http requests.

3.4.3. The system shall use Amazon Cognito for user Authentication and management communication.

3.4.4. The system shall use API Gateway to serve as an interface for external clients to interact with system APIs and manage API requests to AWS Lambda.

3.4.5. The system shall use Amazon Cloudfront to act as the content delivery network for static web page file delivery.

4. System Features

4.1. Comparative Report

4.1.1. Description and Priority

4.1.2. Stimulus/Response Sequences

4.1.3. Functional Requirements

4.1.3.1 The system shall develop a report in a PDF format.

4.1.3.2 The system shall develop a report in an HTML format.

4.1.3.3 The system shall store the report in an S3 Bucket. The system shall create a presigned URL for the report that is valid for 7 days.

4.1.3.4 The system shall store the source URL of the report in DynamoDB. The system shall store the fonts needed for reports in an S3 Bucket.

4.1.3.5 The system shall use AWS Lambda to run its code.

4.1.3.6 The system shall use the Rust Bootstrap for Lambda Runtime.

4.1.3.7 The system shall create error logs via CloudWatch.

4.1.3.8 The system shall create user usage logs via CloudWatch.

4.1.3.9 The system shall notify the user if an error occurs during report creation error via AWS SNS email.

4.1.3.10 The system shall notify website administrators of any report creation errors via AWS SNS email.

4.1.3.11 The report shall query from DynamoDB for tool data via Amazon API Gateway. The system shall deserialize JSON from DynamoDB Tool Database.

4.1.3.12 The system shall derive JSON output from recommendation algorithm.

4.1.3.13 The system shall use the programming language Rust for creation of the report.

4.1.3.14 The report shall display the tools selected as a table.

4.1.3.15 The report table shall display the technology's name.

4.1.3.16 The report shall contain the user's username.

4.1.3.17 The report table shall contain tools already used by the user.

4.1.3.18 The report table shall contain links to each technology's homepage.

4.1.3.19 The report table shall contain what type of technology each row contains.

4.1.3.20 The system shall generate tailored recommendations for cyber tools based on the responses provided by users in the questionnaire.

4.1.3.21The system shall enable users to generate and download reports that document the recommendations provided by the application.

4.2. Recommendation Algorithm

4.3. Fetch Tool Dashboard

4.3.1. Description and Priority

This system is designed to provide users with a dashboard view of tools stored in a DynamoDB table named "Cyber_Tools". It accepts HTTP requests containing filter criteria and pagination information, queries the database accordingly, and returns a JSON response with the relevant tool data and pagination keys for further queries. This has high prority.

4.3.2. Stimulus/Response Sequences

Stimulus: The system receives an HTTP request with an optional JSON body containing filter criteria and a last evaluated key for pagination.

Response: The system processes the request, queries the DynamoDB table based on the provided criteria, and returns a JSON response containing a list of tools that match the criteria and an optional last evaluated key for pagination.

4.3.3. Functional Requirements

4.3.3.1. The system shall be capable of processing HTTP requests that contain JSON bodies with filter criteria, including but not limited to maturity levels and other attributes of the tools.

4.3.3.2. The system shall accept the last evaluated key in the request and return a last evaluated key in the response to support pagination of the dashboard data.

4.3.3.3. The system shall recursively convert any Decimal instances found in the DynamoDB response to integers before returning the data to ensure compatibility with JSON standards and client applications.

4.3.3.4. The system shall correctly process and convert sets to lists for attributes like "Customers" and "Keywords" in the DynamoDB response to ensure proper JSON formatting.

4.3.3.5. The system shall respond if an HTTP request is received without a body, the system shall return a 400-status code with an error message indicating the absence of a request body.

4.3.3.6. The system shall include the "Access-Control-Allow-Origin" header with a wildcard value in the response to allow cross-origin resource sharing (CORS).

4.3.3.7. The system shall accept filter criteria.

4.3.3.8. The system shall return a response in JSON format, including the list of tools that match the filter criteria and an optional last evaluated key for pagination.

4.4. Upload Tool

4.4.1. Description and Priority (medium)

The upload tool functionality is triggered by an event containing a JSON-formatted string in its body. This JSON is parsed into a dictionary, which is then potentially modified before being inserted into the DynamoDB table. The insertion or failure to insert the data triggers responses with appropriate HTTP status codes and messages.

4.4.2. Stimulus/Response Sequences

Stimulus: The Lambda function receives an HTTP request with a JSON-formatted body, from API Gateway.
Response: The function parses the JSON body, potentially modifies the data, and attempts to insert it into the DynamoDB table. It then returns a response indicating success or failure.

4.4.3. Functional Requirements

4.4.3.1. The system shall initialize a client connection to AWS DynamoDB upon invocation.

4.4.3.2. The system shall parse the JSON-formatted string from the event's body into a Python dictionary.

4.4.3.3. The system shall add or update the "Approved" key in the dictionary to False before inserting the data into DynamoDB.

4.4.3.4. The system shall attempt to insert the parsed and modified dictionary as an item into the DynamoDB table named "Cyber_Tools".

4.4.3.5. The system shall return an HTTP response with a status code of 200 and a body message indicating that the data was inserted successfully, pending acceptance.

4.4.3.6. In case of an error, the system shall return an HTTP response with a status code of 500 and a body message detailing the error.

4.4.3.7. If the event body is not in valid JSON format, the system shall return an HTTP response with a status code of 400 and a body message indicating an invalid JSON format.

4.4.3.8. The system shall include HTTP headers specifying the content type as "application/json" and allowing cross-origin requests with "Access-Control-Allow-Origin" set to "*".

4.5. Cognito Authentication

4.5.1. Description and Priority (High)

The system integrates Amazon Cognito which triggers JavaScript functions to handle user authentication and access control within AWS. This feature is high priority due to its role of user security, access control, and personalization features within the application.

4.5.2. Stimulus/Response Sequences

Stimulus: User attempts to register or sign into the application.

- Response: The system initiates the authentication flow with Amazon Cognito, presenting the user with the necessary UI for the process.

Stimulus: User completes authentication details.

- Response: Amazon Cognito verifies credentials. Upon verification, the user will be granted access depending on the user pool.

Stimulus: User forgets password or login details

- The system triggers Amazon Cognito's password recovery flow, sending the user a reset link or code to their registered email

4.5.3. Functional Requirements

4.3.3.1 The system shall authenticate users through Cognito.

4.3.3.2 The system shall use user groups for user management.

4.3.3.3 The system shall monitor login attempts via Cognito.

4.3.3.4 The system shall direct authenticated users with Cognito.

4.3.3.5 The system shall confirm user email through Cognito.

4.3.3.6 The system shall use user group assignments in Cognito.

4.3.3.7 The system shall permit role-based access with Cognito.

4.3.3.8 The system shall support temporary guest credentials.

4.3.3.9 The system shall suspend user accounts with Cognito.

4.3.3.10 The system shall support password policies with Cognito.

4.3.3.11 The system shall allow for customizable user attributes in Cognito.

4.6. Database Access

4.6.1. Description and Priority

The system will utilize Amazon DynamoDB, a NoSQL database service provided by AWS with features including predictable performance and scalability, to store and query information related to each tool covered by the project. Partition keys will be set based on the primary purpose of each tool (i.e., its function), and a sort key with a unique identifier. Each item (or tool) in the table will contain attributes or additional information such as the name of the tool, its features, accessibility and requirements, etc. A full list of attributes is available in Appendix E.

4.6.2. Stimulus and Response Sequences

Stimulus: User makes a search in a search field on the website associated with a particular attribute
- Response: The system shall display each tool where that tool's particular attribute matches the search term

Stimulus: Developer updates a database item through AWS
- Response: DynamoDB table updates and is reflected on the website

4.6.3. Functional Requirements

4.4.3.1 The system shall store tables that each contain a unique string-type partition key that is associated with the function the user wants the tool to serve. (See Appendix D for partition keys)

4.4.3.2 The system shall store tables that contain string-type global secondary indexes (GSI) that will contain a list of attributes. (See Appendix E for GSIs)

4.4.3.3 The system shall perform queries based on desired partition keys and GSIs designated by the user.

4.4.3.4 The system shall allow the user to form complex queries equivalent to AND, OR, or NOT operations.

4.4.3.5 The system shall provide the user with information about a tool based on queries to the database.

4.4.3.6 If the user enters a query and no tool matches the input given, the system shall display an error message that there are no tools available with the given criteria.

4.4.3.7 The system shall generate tables with columns created by the GSIs the user has made selections for.

4.4.3.8 The system shall generate an additional column with a personal recommendation rating for each tool based on analysis from the developer team.

4.4.3.9 Information provided by the database shall be accessible on the website.

4.4.3.10 Information provided by the database shall be able to be integrated into the recommendation reports.

4.4.3.11 The system shall allow the developer team to view and modify the database through AWS.

4.7. Application Programming Interface (API)

4.7.1. Description and Priority (

The system will utilize Amazon API Gateway, a service used to facilitate communication between different AWS services including AWS Lambda, AWS Amplify, and DynamoDB. Amazon API Gateway allows for the development of a RESTful API and uses URL endpoints to invoke different HTTP methods to access or modify resources.

This feature is high priority, as it allows for developers to continue developing the website with integration between the front-end and back-end and enables end-user features including searching the tool database and storing user account information or reports.

4.7.2. Stimulus and Response Sequences

Stimulus: Developer invokes an API method to update a database item

- Response: The API sends a command to DynamoDB to update the item in the database

Stimulus: User queries a search on the available tools

- Response: The API sends a command to DynamoDB to retrieve information on the available tools

4.7.3. Functional Requirements

4.5.3.3 The system shall provide logging and monitoring capabilities for API usage via AWS CloudWatch

4.5.3.5 The system shall define and expose endpoints for main resources, such as /tools, /users, etc.

4.5.3.6 The system shall implement API authentication using Cognito User Pools

4.5.3.7 The system shall support JSON as the standard format for request and response bodies

4.5.3.8 Each API request shall require appropriate headers, including Content-Type, Authorization, and Accept.

4.5.3.9 The system shall use JSON for data serialization in API communication

4.5.3.10 The system shall support the HTTP methods GET, POST, PUT, and DELETE

4.5.3.11 The system shall follow RESTful principles for a consistent and predictable API

4.5.3.12 HTTP status codes shall be used appropriately to indicate the result of API requests

4.5.3.13 The system shall provide clear error messages in the API response, including error codes and descriptions.

4.5.3.14 Additional error details and a trace ID shall be included for debugging purposes.

4.5.3.15 The system shall maintain comprehensive API documentation that includes details on endpoints, request/response formats, authentication, and examples.

4.5.3.16 The system shall utilize AWS Lambda with a Python 3.10 runtime to run code through API Gateway.

5. Other Nonfunctional Requirements

5.1. Performance Requirements

5.1.1. To facilitate an optimal user experience, the CyberTool web application shall utilize a Content Delivery Network (CDN), with Amazon CloudFront.

5.1.2. The system shall deliver dashboard tools in under 10 seconds.

5.2. Safety & Security Requirements

5.2.1. The system shall exclusively use the HTTPS (Hypertext Transfer Protocol Secure) protocol to guarantee the confidentiality and integrity of data during transmission.

5.2.2. The SSL/TLS certificate required for HTTPS shall be obtained from AWS Certificate Manager.

5.2.3. The system shall use Server-Side Encryption (SSE) with the AES-256 (Advanced Encryption Standard with a 256-bit key) encryption standard, in both S3 and DynamoDB.

5.2.4. The system shall use AWS CloudWatch to log all pertinent data transactions.

5.2.5. All log data shall be securely stored in Amazon S3 (Simple Storage Service) with Server-Side Encryption (SSE) enabled.

5.2.6. The system shall use AWS Web Application Firewall (WAF) for filtering and monitoring API Gateway HTTP traffic.

5.2.7. The system shall use AWS Shield to ensure availability of API Gateway.

5.3. Software Quality Attributes

5.3.1. The system shall maintain high availability 99.9999% uptime, ensuring that services are continuously accessible and reliable despite the occurrence of hardware failures or other disruptions.

5.3.2. The system shall leverage AWS's multi-AZ and serverless technologies to provide uninterrupted data and service availability to customers.

5.3.3. The CyberTool shall be deployed across multiple AWS Availability Zones (AZs) to achieve redundancy and fault tolerance.

5.3.4. In the event of a disruption affecting one AZ, the CyberTool shall automatically redirect traffic to a functioning AZ, thus minimizing downtime and service interruptions.

5.4. Business Rules

5.4.1. The system shall require website users to create an individual account using a valid email address for registration to uniquely identify each user.

5.4.2. The system shall enable personalized interactions with the application through the account-based identification of users.

5.4.3. The system shall provide registered users with the ability to complete a questionnaire designed to gather user preferences and requirements.

6. Database

6.1 The system shall use a NoSQL DynamoDB database.

Appendix A: Glossary

AWS – Amazon Web Services
WAF – Web Application Firewall
SSE – Server-Side Encryption
IAM – Identity and Access Management
OU - Organization Unit
SCP - Service Control Policy
RBAC – Role-based Access Control
CIAM – Customer Identity and Access Management

Appendix B: Partition Keys

In AWS, partition keys are essential components of services like Amazon DynamoDB. In DynamoDB, partition keys are used to distribute data across partitions for scalability and performance. Each item in a DynamoDB table is uniquely identified by its partition key, and DynamoDB uses this key to determine the partition in which the item will be stored. This enables efficient querying and retrieval of data.

- Aircraft Log & Anomaly Detection Tools
- Firewall and Network Security
- Endpoint Security
- Secure Communication and Data Encryption
- Access Control and Identity Management
- Security Information and Event Management (SIEM) Systems
- Vulnerability Management
- Threat Intelligence Platforms
- Supply Chain Risk Management Solution
- Regulatory Compliance Tools
- Aviation Focused Tools
- General Tools
- Industrial Control Systems

Appendix C: Attributes

The following list are the attributes that each tool may have. Currently, for reference, a tool to be stored in the DynamoDB database, they must have a Tool_Function, Tool_ID and Tool_Name. This list may continue to change over time due to research.

- Tool_Function: String - Describes the primary function or purpose of the tool.
- Tool_ID: String - A unique identifier for the tool.
- Tool_Name: String - The name of the tool.
- Company: String - The company or developer behind the tool.
- Tool_URL: String - Web URL where more information about the tool can be found.
- Phone: String - Contact phone number for customer support or inquiries.
- Device: String - Types of devices that can use the tool (e.g., mobile, desktop).
- Launched: String - The release date of the tool.
- Requirements: String - System or other requirements necessary to use the tool.
- Features: String - Key features or functionalities of the tool.
- Drawbacks: String - Known limitations or disadvantages of the tool.
- Pricing: JSON - Cost structure for using the tool, assuming multiple components might need to be described in a JSON format.
- Compliance: String - Information on compliance with standards or regulations.
- Accuracy: String - The expected accuracy or reliability of the tool.
- Documentation_Link: String - A URL linking to the tool's official documentation.
- Keywords: String Set - Keywords associated with the tool for search optimization.
- Description: String - A brief description of the tool.
- AI/ML_Use: Bool - Indicates whether the tool uses artificial intelligence or machine learning technologies.
- Cloud_Capable: Bool - Specifies if the tool can be deployed in a cloud environment.
- Customers: String Set - Types or categories of customers who typically use the tool.
- Maturity Level: String - Describes the developmental stage of the tool (e.g., Beta, Released, Mature).
- ToolBox: String Set - Indicates if the tool is part of a larger suite or toolbox.
- Approved: Bool - Indicates whether the tool has been officially approved for use.
- Aviation_Specific: Bool - Indicates if the tool is specifically designed for use in the aviation industry.