**Software Requirements Specification (SRS)**

**Version 1.0**

DevSecOps Team

Spring 2022

University of Maryland Global Campus

SWEN 670

Prepared by Andrew Nicolette and Robert Wren

For Approval by Dr. Mir Assadullah

# **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 01/13/2022 | 1.0 | Initial DevSecOps Software Requirements Specification | Andrew Nicolette  Robert Wren |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Table of Contents**

[**Revision History 2**](#_Toc93578354)

[**Table of Contents 3**](#_Toc93578355)

[**1 Introduction 5**](#_Toc93578356)

[**1.1 Purpose 5**](#_Toc93578357)

[**1.2 Document Conventions 5**](#_Toc93578358)

[**1.3 Intended Audience and Reading Suggestions 5**](#_Toc93578359)

[**1.4 Scope 5**](#_Toc93578360)

[**1.5 References 5**](#_Toc93578361)

[**2 Overall Description 6**](#_Toc93578362)

[**2.1 Product Perspective 6**](#_Toc93578363)

[**2.2 Product Functions 6**](#_Toc93578364)

[**2.3 User Classes and Characteristics 6**](#_Toc93578365)

[**2.4 Operating Environment 7**](#_Toc93578366)

[**2.5 Design and Implementation Constraints 7**](#_Toc93578367)

[**2.6 User Documentation 7**](#_Toc93578368)

[**2.7 Assumptions and Dependencies 7**](#_Toc93578369)

[**3 System Features 8**](#_Toc93578370)

[**3.1 Code Repository 8**](#_Toc93578371)

[**3.2 Automated Application Build Pipeline 8**](#_Toc93578372)

[**3.3 Automated Application Testing 8**](#_Toc93578373)

[**3.4 Automated Merge Request Reviewers 8**](#_Toc93578374)

[**3.5 Automated Application Deployment 8**](#_Toc93578375)

[**4 External Interfaces 9**](#_Toc93578376)

[**4.1 Hardware Interfaces 9**](#_Toc93578377)

[**4.1.1 Deployment Host 9**](#_Toc93578378)

[**4.2 Software Interfaces 9**](#_Toc93578379)

[**4.2.1 GitHub 9**](#_Toc93578380)

[**4.3 Communications Interfaces 9**](#_Toc93578381)

[**4.3.1 HTTPS 9**](#_Toc93578382)

[**4.3.2 Email 9**](#_Toc93578383)

[**5 System Features/Modules 10**](#_Toc93578384)

[**5.1 Code Repository 10**](#_Toc93578385)

[**5.1.1 Description and Priority 10**](#_Toc93578386)

[**5.1.2 Stimulus/Response Sequences 10**](#_Toc93578387)

[**5.1.3 Functional Requirements 10**](#_Toc93578388)

[**5.2 Automated Application Build Pipeline 11**](#_Toc93578389)

[**5.2.1 Description and Priority 11**](#_Toc93578390)

[**5.2.2 Stimulus/Response Sequences 11**](#_Toc93578391)

[**5.2.3 Functional Requirements 11**](#_Toc93578392)

[**5.3 Automated Application Testing 12**](#_Toc93578393)

[**5.3.1 Description and Priority 12**](#_Toc93578394)

[**5.3.2 Stimulus/Response Sequences 12**](#_Toc93578395)

[**5.3.3 Functional Requirements 12**](#_Toc93578396)

[**5.4 Automated Merge Request Reviewers 13**](#_Toc93578397)

[**5.4.1 Description and Priority 13**](#_Toc93578398)

[**5.4.2 Stimulus/Response Sequences 13**](#_Toc93578399)

[**5.4.3 Functional Requirements 13**](#_Toc93578400)

[**5.5 Automated Application Deployment 14**](#_Toc93578401)

[**5.5.1 Description and Priority 14**](#_Toc93578402)

[**5.5.2 Stimulus/Response Sequences 14**](#_Toc93578403)

[**5.5.3 Functional Requirements 14**](#_Toc93578404)

# **Introduction**

## **Purpose**

The purpose of this document is to define both the functional and nonfunctional requirements for standing up and enhancing UMGC’s CI/CD development pipeline. This document is to be referred by the development teams, DevSecOps team, and other stakeholders to understand the security controls around their repositories, how to conform to the pipeline standards, and how to build a secure application for production level use. All requirements presented in this document are high priority and committed for release 1.0.

## **Document Conventions**

This document conventions are currently based on the IEEE 830 format which is utilized in the decomposition of functional requirements. The priorities for each of the requirements are written in such a way as to assume that they correspond to the priority level of the parent system feature.

The references follow APA 7th edition format. Each of the headers of the sections in this document shall be in Bold Times New Roman 14-point font. Headers for sub-sections shall be 12-point font. The content for each of the sections shall be in Times New Roman 12-point font. Topics shall be italicized in Times New Roman 12-point font.

## **Intended Audience and Reading Suggestions**

The intended audience for this document is the internal and future members of each product team. This document will be used in parallel with other future documents like the runbook guide, programmer’s guide, and readme files to refer to when trying to understand the build pipeline and repository security. Specific team members shall find different parts of the document more useful than others. Project managers and business analysts are more likely to refer to sections 1 and 2 whereas sections 3,4,5 are more useful for the developers and testers. As of version 1, the project has been implemented under the guidance of UMGC professors and teaching assistants/mentors.

## **Scope**

The purpose of this project is to provide a continuous line of integration, communication, and automation across the development within the capstone project. A more enhanced pipeline and repository will standardize the environment and provide a dynamic way to ensure the code is being developed to a high-quality standard. The pipeline will increase both the quality and rate at which the software is being developed. It also will require project teams to adhere to all the tests and only allow the application to be deployed and built if all the automated tests and code coverage results passed at certain development points. Depending on the timeline and future requirements, each build artifact will also be pushed to the App Store and Play Store for beta testing and/or production level use.

## **References**

Pham, I., Leung, V. (2021, February 04). Software Requirement Specification. University of Maryland Global Campus.

# **Overall Description**

## **Product Perspective**

Utilizing previous research and development, the two product teams will further enhance an application that uses natural language processing and cloud-based technology to assist those that suffer from short term memory loss. Overtime, those who have suffered from this condition have proven notetaking and future reminders is a key activity to managing this condition. These notes range from a multitude of things. Depending on the patient’s condition, it can be simple reminders like getting certain things at the grocery store or important reminders like taking medication or remembering people’s names. With the primary goal being to extend functionality and build separate user interfaces for the admin and patient, the product team will also improve the functionality for reminders and status reports.

The DevSecOps project focuses on the creation and the enhancements to the CI/CD pipeline and repository security that houses the code for the Short-Term Memory Loss application. As there are a multitude of different deployments and build methodologies, the DSO team will assist in the rapid delivery of high-quality software on a frequent basis and will continue to improve the work the Fall 2021 semester completed. More specifically, this SRS will concentrate primarily on the feature adds. This will allow the product teams to spend more time on development rather than ensuring compliance to the build and deployment guidelines.

## **Product Functions**

The DSO team will add features to do the following:

* Setup branch protection
* Incorporate branch protection policies
* Incorporate code owner pull request reviews
* Further enhance code quality and security vulnerability checks
* Further enhance automated tests on pushes, merge request, and builds
* Publish stable application to Play Store and/or App Store

## **User Classes and Characteristics**

**Developers**: This user class is the main class of users that will leverage the DSO pipeline. Developers shall use the technologies within the scope of which the DSO team provides to facilitate the deployment and integration activities.

**Lead** **Developer**: A member shall be designated as the lead developer who will work with the DSO team to merge changes from the development branches to the main branch. This lead developer will also provide guidance to other team members and approve merge requests from the feature branches to the development branch.

**DSO Team**: This user class is a class of users that will be the main contact point for the development teams for any related DSO matters. Each member of this team will wear multiple hats. They will configure the developer’s repositories, create the build and test pipelines, and triage any questions from the developers regarding the maintenance of the pipeline or repository. The DSO team will not be involved in how the content of the repository is organized or with any of the design decisions the product team comes up regarding building the overall application.

## **Operating Environment**

The following are operating environment considerations:

OE-1: The DSO project shall support multiple operating systems as developers use their personal machines with innumerable configurations. The project utilizes industry tools that have clients for desktop systems such as Windows, MacOS, and Linux. Support shall be with development machines that are of relatively recent vintage.

OE-2: Much of the access to resources shall be via the Internet through modern mobile operating systems like iOS and/or Android.

OE-3: Feedback is solicited from development teams for the coding languages and frameworks that shall be used so that DevSecOps team could design the pipeline while proactively considering these requirements.

OE-4: For cost effectiveness, the DSO project shall leverage open source tools such as GitHub and others where it is applicable to deploy and test components of the application.

## **Design and Implementation Constraints**

The DSO team shall utilize GitHub as their hosting software for version control. For mobile application implementation, the DSO expects the product teams to use Dart/Flutter. The DSO team will potentially utilize GitHub actions, Azure DevOps, or other open-source services with no associated costs to build and improve their pipeline.

## **User Documentation**

In addition to this SRS, the DSO team will deliver the following documentation along with the software:

* Project Plan
* Technical Design Document
* User Guide
* Test Report
* Programmer Guide
* Deployment and Operations Runbook

## **Assumptions and Dependencies**

The DSO team is assuming that the lead developer and development team leads will maintain all back-end and front-end server changes that are necessary to the development branch. The DSO will primarily focus on managing the repository and build pipeline and not with the application software development. Furthermore, the DSO team is dependent on the existing code working as expected. All changes made by the Spring 2022 DSO team rely on code written by previous semester DSO teams.

# **System Features**

Code Repository

The feature utilizes the free tier of a distributed version-control code repository for tracking and collaboration which allows developers to retrieve and contribute to their project’s source code in collaboration with team members.

Automated Application Build

This feature allows the DSO team to work with developers to add the build solution for each component of the application to the DSO pipeline where the pipeline is integrated with the remote code repository and triggered by new merges or pushes. The feature also allows the pipeline to automatically compile the code according to specific programming languages. The notification is sent to the appropriate team for the status of each build. The build task can also be run manually as needed.

Automated Application Testing

This feature allows DSO team to work with developers to add the testing solution for each component of the application to the DSO pipeline. It is triggered before the build is successful. The feature also allows the pipeline to automatically run the selected functional testing (Unit/Integration). The notification is sent to the appropriate team for the status of each run. The testing task can also be run manually as needed.

Automated Merge Request Reviewers

This feature allows the DSO team to configure stringent repository security and branch protection. It is triggered after a merge is instantiated and allows certain members of the team to review updated files in the repository based on the files’ location. A notification is sent to the appropriate team/person for the reviewal of each change.

Automated Application Deployment

This feature allows the DSO team to release the application to an environment for various purposes after the code has passed the build, test, and code analysis. Depending on future requirements, this could involve deploying future releases to the App Store and Play Store.

# **External Interfaces**

Hardware Interfaces

### **Deployment Host**

For development purposes, the application will be retrieved from the Spring 2022 GitHub Repository and deployed on each developer’s localhost for feature improvements and additions.

Software Interfaces

### **GitHub**

The system will use a GitHub repository for committing, pushing, pulling, and cloning the source code.

Communications Interfaces

### **HTTPS**

The interface portion of the code will be accessible via an HTTPS link such that it can be cloned, forked, and merged from any system that has access to the internet.

### **Email**

Build notifications shall be sent via email communication to the configured repository admins and associated users of interest.

# **System Features/Modules**

Code Repository

### **Description and Priority**

A developer who has been invited to the organization and has the required access to the repository will retrieve and contribute to their project’s source code in collaboration with other team members. They will create feature branches and update the code as necessary. After that, a merge request will be initiated and reviewed back into the main and development protected branches.

### **Stimulus/Response Sequences**

Stimulus: A developer requests to merge changes to the main or development protected branch.

Response: The system displays the restriction of the protected branch with rules and the information to merge. The lead developer/DSO POC or a DSO team member (the approving authority) who are configured to have the permission to merge is also notified via email.

Stimulus: The approving authority approves the merge.

Response: The system merges new changes to the protected branch and tracks those changes. The system also notifies the requester for the approval or rejection.

### **Functional Requirements**

REQ-1.1: After a merge request has been initiated, the system shall display information on who the reviewers are.

REQ-1.2: After a merge request has been initiated, the system shall notify the reviewers that they have a merge request needing approval.

REQ-1.3: After a merge request has been initiated, the system shall inform the review of the potential conflicts.

REQ-1.4: After a merge request has been initiated, the system shall allow the reviewer to approve or deny it along with some comments why.

REQ-1.5: After a merge request has been approved, the system shall merge the changes into the protected branch.

REQ-1.6: After a merge request has been approved, the system shall track the changes and have a revert option if necessary.

REQ-1.7: After a merge request has been approved, the system shall notify the developer with the reviewer’s comments.

REQ-1.8: After a merge request has been denied, the system shall notify the developer with the reviewer’s comments.

Automated Application Build

### **Description and Priority**

This feature allows the DSO team to create a build solution and artifact for the application. It will be triggered by new merges or pushes to the DSO pipeline and allows the pipeline to automatically compile the code and create a build artifact. The notification is sent to the appropriate team for the status of each build. The build task can also be run manually as needed. Priority=High.

### **Stimulus/Response Sequences**

Stimulus: Changes are merged to either the main or development protected branches and are approved by the approving authority in CODEOWNERS file.

Response: The pipeline triggers the task to build the application/solution. The pipeline also notifies the leads and merge requestor of the success or failure status of the build.

### **Functional Requirements**

REQ-2.1: The pipeline shall be integrated with the remote code repository.

REQ-2.2: Upon changes merged or pushed to the main or development protected branches, the pipeline shall monitor changes in the corresponding branch.

REQ-2.3: Upon changes merged or pushed to the main or development protected branches, the pipeline shall trigger the build task of the application.

REQ-2.4: Upon changes merged or pushed to the main or development protected branches, the pipeline shall notify the approving authority and the merge requestor of the success or failure status of the build.

Automated Application Testing

### **Description and Priority**

This feature allows DSO team to work with developers to add an automated testing solution for the application to the DSO pipeline. It is triggered before the build and only creates build artifacts if all the tests have passed.

### **Stimulus/Response Sequences**

Stimulus: Changes are merged or pushed to either the main or development protected branches and are approved by the approving authority in CODEOWNERS file.

Response: The pipeline triggers and runs all the unit, integration, and functional tests configured and notifies the leads and merge requestor of the status.

### **Functional Requirements**

REQ-3.1: Upon changes merged to the main or development protected branches, the pipeline shall trigger multiple unit and integration tests.

REQ-3.2: Once the tests complete, the pipeline shall notify the approving authority and the merge requestor of the status via email.

Automated Merge Request Reviewers

### **Description and Priority**

This feature is triggered after a merge is instantiated and allows certain members of the team to review updated/added files in the repository based on the files’ location.

### **Stimulus/Response Sequences**

Stimulus: A merge request is instantiated to one of the protected branches.

Response: Depending on the location of the changes, one to many reviewers will be notified and required to approve the change.

### **Functional Requirements**

REQ-4.1: Upon changes requesting to be merged to the main or development protected branches, the pipeline shall prepopulate one to many reviewers based on the directory location of the files.

REQ-4.2: Upon changes requesting to be merged to the main or development protected branches, the pipeline shall notify one to many reviewers based on the directory location of the files and ask for their approval.

Automated Application Deployment

### **Description and Priority**

This feature allows the DSO team to release the application to an environment for various purposes after the code has passed the build and tests. Depending on future requirements, this could involve deploying future releases to the App Store and Play Store.

### **Stimulus/Response Sequences**

Stimulus: A merge request is instantiated to one of the protected branches. The pipeline runs and passes all tests and builds and triggers the deployment task.

Response: The pipeline runs the deployment tasks and sends build artifact to specified location.

### **Functional Requirements**

REQ-5.1: Upon the test and builds being successful after a merge request has been instantiated, the pipeline shall trigger the deployment task to run.

REQ-5.2: Upon the test and builds being successful after a merge request has been instantiated, the pipeline shall deploy the build artifact to a specified location.