**UMGC Capstone Project**

**Form Scriber**

**DevSecOps Project Plan**

**Version 4.0**

**Prepared by:**

**Ivy Pham**

**Vincent Leung**

**SWEN670 – Spring 2021**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 1/20/2021 | 1.0 | Initial document | Ivy Pham |
| 1/21/2021 | 1.1 | Added overview, definitions, roles, communication management, deliverables and milestones | Vincent Leung |
| 1/23/2021 | 1.2 | Added project summary and purpose | Vincent Leung |
| 1/24/2021 | 1.3 | Added assumptions and constraints | Vincent Leung |
| 1/26/2021 | 1.4 | Modified Overview and Organization Overview | Ivy Pham |
| 1/27/2021 | 1.5 | Various edits and additions | Ivy Pham |
| 1/27/2021 | 1.6 | Added risk analysis | Vincent Leung |
| 1/31/2021 | 1.7 | Various edits and additions | Ivy Pham |
| 2/1/2021 | 1.8 | Incorporated Johnny’s comment | Vincent Leung |
| 2/2/2021 | 1.9 | Incorporated Johnny’s comment | Ivy Pham |
| 2/21/2021 | 1.10 | Incorporated changes from post Milestone 1 | Vincent Leung |
| 2/25/2021 | 2.0 | Review and submit for 2nd milestone | Ivy Pham |
| 3/15/2021 | 3.0 | Added test plan | Vincent Leung |
| 3/16/2021 | 3.1 | Review and submit for 3rd milestone | Ivy Pham |
| 3/28/2021 | 4.0 | Prepare final version | Vincent Leung |

**Table of Contents**

[1. Executive Summary 5](#_Toc66798830)

[1.1 Business Need/Opportunity 5](#_Toc66798831)

[1.2 Purpose 5](#_Toc66798832)

[1.3 Objectives 5](#_Toc66798833)

[1.4 Scope 6](#_Toc66798834)

[1.5 Definitions, Acronyms, and Abbreviations 6](#_Toc66798835)

[2. Organization 7](#_Toc66798836)

[2.1 Internal Structure 7](#_Toc66798837)

[2.2 External Structure 7](#_Toc66798838)

[2.3 Roles and Responsibilities 8](#_Toc66798839)

[2.4 Communication Management Plan 9](#_Toc66798840)

[2.5 Cost Analysis 11](#_Toc66798841)

[3. Execution 11](#_Toc66798842)

[3.1 Methodology 12](#_Toc66798843)

[3.2 Key Deliverables and Milestones 12](#_Toc66798844)

[3.3 Change Management 13](#_Toc66798845)

[3.4 Assumptions and Constraints 14](#_Toc66798846)

[3.5 Risk Analysis 15](#_Toc66798847)

[4. Test Plan 16](#_Toc66798848)

[4.1 Introduction 16](#_Toc66798849)

[4.1.1 Purpose 16](#_Toc66798850)

[4.1.2 Coverage 16](#_Toc66798851)

[4.2 Requirements 16](#_Toc66798852)

[4.2.1 Software Requirements 17](#_Toc66798853)

[4.2.2 Hardware Requirements 17](#_Toc66798854)

[4.3 Testing 17](#_Toc66798855)

[4.3.1 Test Strategy 17](#_Toc66798856)

[4.3.2 Proposed Test Cases 17](#_Toc66798857)

[4.4 Assumptions and Constraints 22](#_Toc66798858)

[4.4.1 Assumptions 22](#_Toc66798859)

[5. Future Work 22](#_Toc66798860)

[6. Project Schedule 23](#_Toc66798861)

# 1. Executive Summary

## 1.1 Business Need/Opportunity

The Software Engineering Capstone project requires the creation of a prototype application called Form Scriber, which is the development of an artificial intelligence (AI) software to support doctors, medical technicians, police officers and others. The application primarily assists professionals in public service to capture and prepare text reports during conversations with their clients. For example, the conversation between a physician and patient can be recorded to automatically prepare reports that can be edited and saved to an Electronic Medical Record (EMR). To support the development of The Form Scriber application, the DevSecOps (DSO) team was formed. The DevSecOps Project Plan is the initial document to outline the schedule, techniques, and tools to support the fast-paced software development project with efficiency.

The Form Scriber DevSecOps project will enable the development teams to utilize a set of best practices and automation tools to build, test, secure, publish and deploy their services and applications. A continuous integration (CI) pipeline with a centralized code repository will be provided to the three development teams with the test automation and static code analysis tools to verify functionalities and ensure industry standards are followed, including security. A continuous delivery (CD) pipeline will also be provided to have the code built and deployed to the assigned environment.

## 1.2 Purpose

The project aims to streamline software development and deployment via automated processes that may otherwise require manual intervention that can be time-consuming and/or error-prone. The process ensures a high-quality end product that can be deployed in a fast and reliable manner and on a consistent and frequent basis. In addition to development and deployment operations, security will also be emphasized as a requirement in code quality by utilizing security practices to identify vulnerabilities earlier in the feedback cycle. The CI/CD pipeline aims to ensure environment stability, enable high release velocity, reduce defects with automated testing, monitor processes, and provide metrics for tracking.

## 1.3 Objectives

Within the course context, the DevSecOps team exists to provide cohesion between the individual project teams and to serve as a unifying agent for the selection of industry-standard Tools, Techniques, and Practices (TTP). Guided by the Project Coordinator and Project Mentors, the DevSecOps team primary objectives are to

* Develop an internal structure for software development across the segregated project teams
* Advise project teams through the respective Project Manager of best practices discovered through both external research and intra-team experiences
* Create standardized instructional documentation of each team’s project startup
* Provide technical design documents to aid in continuity across the business
* Lessons learned from this project will benefit future projects.

## 1.4 Scope

The guidelines of the Project Coordinator and, to the extent possible, the guidance of the DevSecOps team, are recommendations provided to the development teams representing an opinionated architecture structure for the development, testing, and delivery of each of the client’s requirement. Within the DevSecOps scope of operations, the team contributes to the term’s project execution by supporting the following tasks:

* + Create and manage the centralized code repository for each development team
  + Provide guidelines for industry standards and best practices for tech stacks
  + Support project development teams through setup and utilization of recommended development stacks
  + Provide CI/CD pipeline for each development to build, test, analyze and deploy.
  + Provide overall code reviews to establish and enforce an intra-team, agreed-upon code standards
  + Provide strategies to support CI/CD pipeline with limited resources with no cost solution

## 1.5 Definitions, Acronyms, and Abbreviations

The below list defines popular project terms and acronyms.

* CI/CD - Continuous integration and continuous deployment, or CI/CD pipeline, a set of operating principles, practices, and tools used to deliver code often and reliably.
* DialogFlow Team – The project development team creating the back-end services and REST APIs utilizing Google’s Dialogflow that understands natural language to process conversations.
* DSO - DevSecOps, Development Security & Operations, or the set of practices that combines software development, IT security, and IT operations in the integration and deployment of software applications.
* DSO POC – DevSecOps point of contact from each team.
* Form Scriber - The application under development, an AI-based voice recognition software for mobile devices that analyzes conversations and transcribes key information into user predefined forms.
* Git – The project’s chosen version control system for source code.
* Mobile Team – The project development team creating the mobile interface that the user would use to initiate recordings and analyses of spoken conversations, and to interface with the Dialogflow implementation and web interface.
* PM – Project manager
* Repo - Source code repository to store code and documentation for software hosted on the web, such as GitHub.

# 2. **Organization**

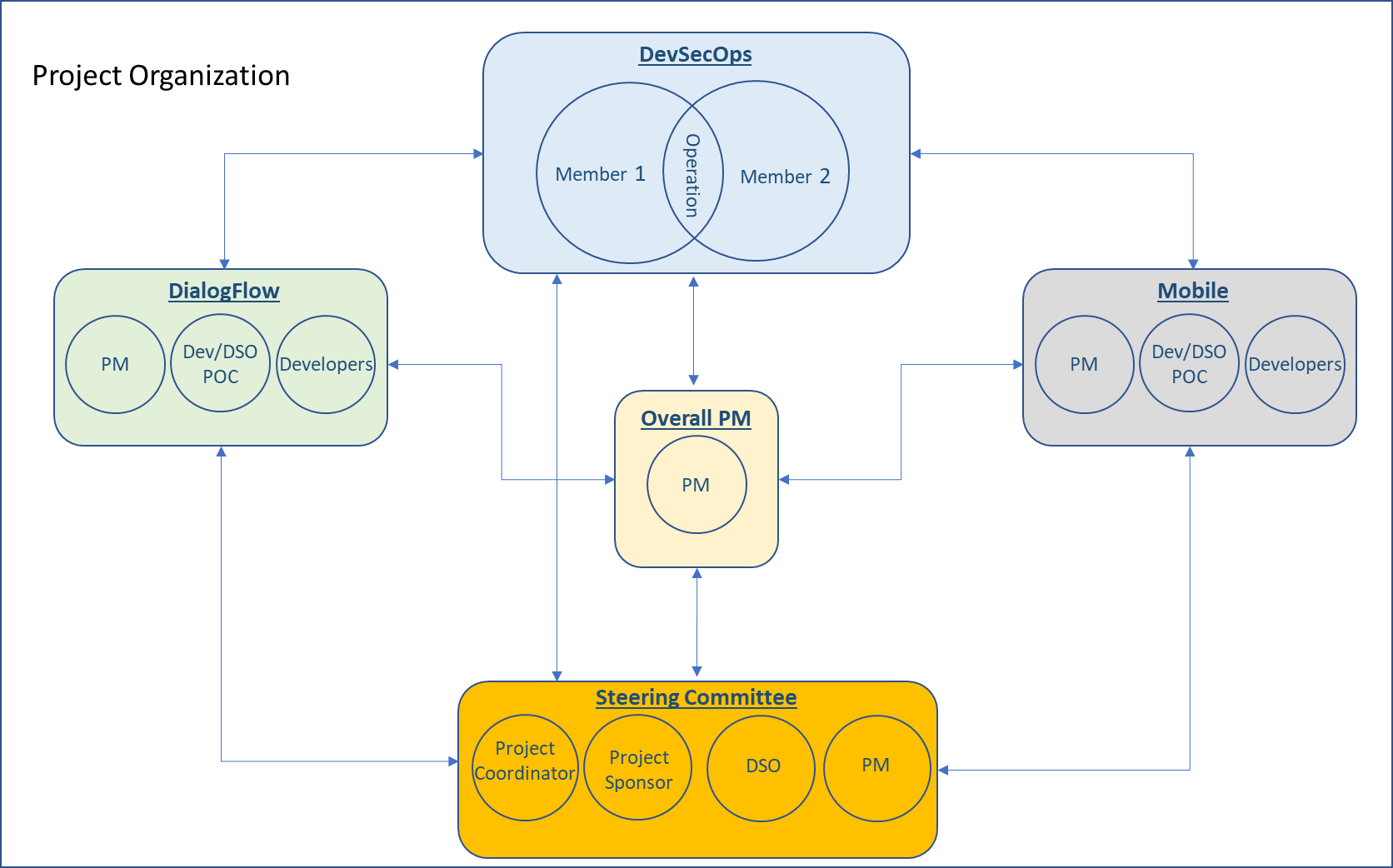
The Form Scriber project consists of an overall project manager, DSO team, DialogFlow team and Mobile team. Each application team has a project manager, developers and testers. All teams are led by the project coordinator and project sponsors.

## 2.1 Internal Structure

As the DSO team is two members strong, each member shares all roles and responsibilities to ensure the continuity of the project. DSO team collaborates within itself throughout the life of the project to provide consistent recommendations and best practices to all application teams.

## 2.2 External Structure

The DSO team main responsibility is to serve all teams where it is deemed necessary. Therefore, DSO team can be reached via Teams and email from any stakeholders to provide technical assistance and DSO guidelines. DSO team also reaches out to any stakeholders as needed per the communication plan. While the DevSecOps team aims to provide technical assistance to all project team members, the primary external interface between each of the development teams is through or in coordination with the respective Project Manager and Lead Developer/DSO POC for any given team. This ensures that any DevSecOps requests or decisions have the vetting of all team members, and between project PMs when appropriate.



**Figure 1.** Project Organization

*Notes*. This visualizes the overall structure and communications flow of the internal and external teams and stakeholders involved with the Form Scriber project.

## 2.3 Roles and Responsibilities

The responsibilities of the DSO team include, but are not limited to, the planning, designing, and implementation of a continuous integration/continuous deployment pipeline to support the development and deployment of the Form Scriber application for the Capstone Project. The team members will work together to document and implement a working CI/CD pipeline for eventual deployment of said application under development, as well as be liaison and support for development team members, its customers. The DSO team will set and uphold such things as best practices, coding standards, and offer technical expertise and assistance to development teams using its framework, in addition to soliciting feedback and suggestions for improvement.

At the time of this document inception, from a DevSecOps perspective, there are several key roles that are recognized and assigned to key Capstone participants:

* + Project Sponsor: Dr. Mir Assadullah is currently the sole proprietary of this role and serves as chief advisor of all projects. As such, Dr. Assadullah may exercise authority to alter, limit, or expand any of the perimeters of the project(s) to meet Capstone requirements.
  + Project Mentors: are the experts in the information technology who provide guidance and advisory for any impediments, issues and improvements.
  + Project Managers: There is an overall project manager who works closely with team project managers to track the timeline and deliverables. Team PMs are liaisons between their assigned team and all stakeholders. The PM communicates any needs or concerns from the teams to affected stakeholders and vice versa. Each team has identified a primary Project Manager; however, to allow for flexibility in team roles, teams may collectively decide to reassign the role during any particular phases of the project. Such a change must adhere to the communication plan to notify the core steering committee members.
  + Software Developers: Identified developers are responsible for utilizing Capstone-specified programming languages, tools, and infrastructure to write, test, and implement client specified applications. At times during the production effort, it may be necessary for DSO team members to interface directly with a given project's assigned developers outside of the specified communication plan to establish processes, clarify requirements, or perform similar related items to meet the overall Capstone goals.
  + Steering Committee: Refers to those critical roles needed to make the majority of critical decisions that result in an impact on multiple team's internal processes or deliverables. As of this writing, those roles are:
    - Project Coordinator
    - Project Mentor
    - Overall Project Manager
    - Team Project Manager
    - DevSecOps

## 2.4 Communication Management Plan

The DevSecOps team will strive to communicate frequently to all project stakeholders and respond in a timely manner to requests and incidents. DSO project status will be tracked and updated on a daily basis to ensure progress towards milestone goals and fulfillment of deliverables. It is preferable that development and project management team members message DSO members through the project’s Microsoft Teams channels and chat sessions. However, other options are also available such as email or phone. The DSO team will liaison with individual development teams via points of contact of respective teams as noted as Lead developer/DSO POC. The DSO team will escalate issues to team project managers and business management as needed.

Approach

With interface requirements for each of the project teams, each DSO team member leads the effort of ensuring effective communication within the Capstone construct. They communicate with the steering committee members to determine their preferred frequency and method of communication. At the current time, the steering committee is notified by weekly update emails to ensure projects on individual schedules. In addition to identifying communication frequency and method, the project's communication channels also identify and confirm that stakeholders have access to respective channels. After changes get approved, the plan is updated, and supporting documentation is distributed to update the project team and all stakeholders.

Project Team Directory

|  |  |  |
| --- | --- | --- |
| **Role** | **Team Member** | **Contact** |
| Project Advisor | Dr. Mir Assadullah | mir.assadullah@faculty.umgc.edu |
| Project Mentor | Johnny Lockhart | janthony@umgc.dev |
| Project Mentor | Roy Gordon | uspsrgordon@aol.com |

**Table 1.** Project Advisor/Mentor

|  |  |  |
| --- | --- | --- |
| **Role** | **Team Member** | **Contact** |
| Project Manager | Menen Fantay | mfantay@student.umgc.edu |

**Table 2.** Overall Project Management

|  |  |  |
| --- | --- | --- |
| **Role** | **Team Member** | **Contact** |
| DevSecOps Engineer | Ivy Pham | ipham@student.umgc.edu |
| DevSecOps Engineer | Vincent Leung | vleung1@student.umgc.edu |

**Table 3.** DSO Team

|  |  |  |
| --- | --- | --- |
| **Role** | **Team Member** | **Contact** |
| Project Manager | Stefon Williams | swilliams538@student.umgc.edu |
| Lead Developer/DSO POC | Caleb Crickette | ccrickette@student.umgc.edu |
| Lead Developer | Jean-nae’ Ddrick | jdedrick@student.umgc.edu |
| Developer | Brian Malott | jmalott@student.umgc.edu |
| Developer | Thomas Barton | tbarton7@student.umgc.edu |
| Developer | Sohai Sobhani | ssobhani@student.umgc.edu |
| Developer | Jean-nae' Dedrick | jdedrick@student.umgc.edu |
| Business Analyst | Eugene Kim | ekim55@student.umgc.edu |
| Test Engineer | Nicholas Ballo | nballo@umgc.edu |

**Table 4.** Dialogflow Team

|  |  |  |
| --- | --- | --- |
| **Role** | **Team Member** | **Contact** |
| Project Manager | Sylvia Lopez-Willis | slopezwillis@student.umgc.edu |
| Lead Developer/DSO POC | Karim Mansour | kmansour@student.umgc.edu |
| Developer | Bertina Lee | blee99@student.umgc.edu |
| Developer | Brian Malott | jmalott@student.umgc.edu |
| Developer | Komi Noukafou | knoukafou@student.umgc.edu |
| Developer | Joselito Ocampo | jocamo426@gmail.com |
| Business Analyst | Ermias Seyoum | eseyoum@student.umgc.edu |
| Test Engineer | Sompon Boontho | sboontho@hotmail.com |

**Table 5.** Mobile Team

## 2.5 Cost Analysis

It is the mission of the DevSecOps team to provide all project teams and the Project-Sponsors a little to no cost solutions to on which to develop, deliver, and prototype software deliverables. Within these boundaries, the expected cost of all project-related expenditures is non-existence to negligible. Providing a no-cost solution limits the capability of the deliverable products to operate in the realm of open-source, freely available platforms. If accepted, the Project Sponsor may expand the capabilities of the specified deliverables at their discretion; however, they accept all cost increase from the expansions of said products.

While the project teams may exercise their independence and utilize any for-cost products in their internal development operations, the team cannot pass such expenses on to the client. Lastly, after the Capstone specified timeline, the teams must offer their deliverables in a non-expenditure acquired deployment solution.

# 3. Execution

Given the support nature of the DevSecOps team, it is logical that any proposed internal project schedule put forth is significantly influenced by given project teams' ability to meet and deliver artifacts on schedule. As such, the DevSecOps team reserves the right to make changes to the deliverable timeline within five (5) days of a set milestone, utilizing the above communication plan.

## 3.1 Methodology

The Capstone project is organized in a 12-week session starting January 13, 2021 with a four-phase approach identified by Milestones 1 through 4. With this specification, each phase has a definitive start/stop marker, with measurable criteria for success (identified by team-specified deliverables) that closes before the next phase. However, acknowledging the dynamic nature and scope of each software development initiative, changes or decisions from earlier phases may be revisited. Nevertheless, this is done within firm adherence to the specified Change Management Plan (CGMP). The included project schedule provides an outline of each of the major phases, with expected milestones.

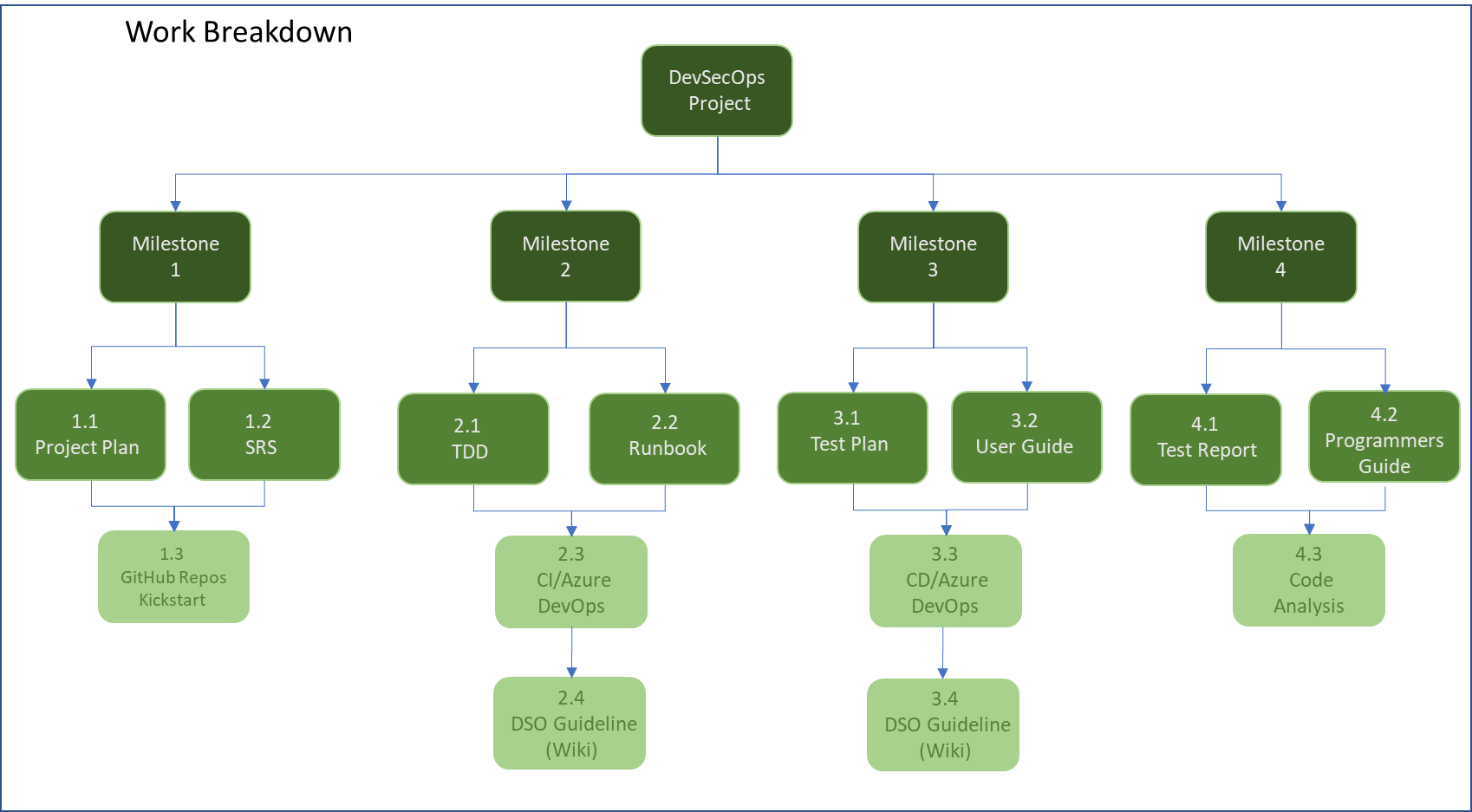
## 3.2 Key Deliverables and Milestones

The Form Scriber DevSecOps Project schedule shall take place over the course of eleven weeks and be delineated by four milestones to coincide with those of the development teams and overall project timeline. The milestone name, due date, and deliverables are detailed in the DevSecOps Milestones and Deliverables table below. Refer to descriptions below for details on contents of said deliverables.

|  |  |  |
| --- | --- | --- |
| **Milestone** | **Due Date** | **Deliverables** |
| One (1) | 02/02/2021 | Project Plan, SRS |
| Two (2) | 02/23/2021 | TDD, DSO Runbook |
| Three (3) | 03/16/2021 | Test Plan, User Guide |
| Four (4) | 03/30/2021 | Test Report, Programmers Guide |

**Table 7.** DevSecOps Milestones and Devliverables

* Project Plan – This document, detailing the overall plan, schedule, and deliverables of the Form Scriber DevSecOps Project. The document will be updated as required during the life of the project. However, it will be updated with a major version for each milestone.
* SRS - Software Requirements Specification document, describing the requirements for the CI/CD pipeline supporting the development and deployment of the Form Scriber application.
* TDD - Technical Design Document, describing the design of the entire CI/CD pipeline and its individual components.
* DSO Runbook - Deployment and Operations Guide (Runbook) document, an operation manual with steps on executing processes related to set up and configuration of the DSO pipeline.
* Test Plan - Document detailing the scope and strategy for testing the implementation of the DSO pipeline which will be added to the Project Plan at milestone three.
* User Guide - Document detailing the best practices, expectations, and required tools and environments to get started on integration, testing, and deployment of the project application.
* Test Report - Document detailing the test execution findings for testing session(s) on the DSO pipeline.
* Programmers Guide - Document detailing the implementation of the pipeline as reference for future DevSecOps teams for usage or maintenance.



**Figure 2.** Work Breakdown

*Note.* The documentation deliverables shall inform the implementation components for each milestone.

## 3.3 Change Management

Change Management, in this context, refers to the changes that deviate from the outline in this document. Whether changes are trivial or complicated, the project team must manage changes to prevent scope creep. The CGMP includes submitting, reviewing, and approving changes at all echelons of the development process. All requests and submissions follow the change control process described below to ensure changes are approved, tracked, and managed per this CGMP.

The proposed CGMP approach confirms that changes throughout the project lifetime are defined, reviewed, and agreed on and adequately implemented and communicated to all involved parties. Also, it ensures that only changes within the scope of the project are considered. By following these processes, it is the goal of the DevSecOps team to prevent unnecessary changes from occurring and help the team members to focus only on beneficial changes within the project scope.

* + Internal change(s) to the DevSecOps project shall be communicated from one DSO member to the other via Teams chat, email, or phone. If the change(s) affect the development teams, the information shall be passed on to development team DSO POCs and development team PMs via a link to the DSO Teams wiki page or a document within the DSO Teams file repository.
  + External change(s) to the Form Scriber project that require a new design or implementation of DSO shall require consultation with both DSO members, development team PMs, overall PM, and optionally, development team DSO POCs.
  + For communications outreach, the DSO team shall contact project team members in the order below, with each line following representing an additional level of escalation:
    - Development team(s) as a whole
    - Development team DSO POCs
    - Development team PMs
    - Overall PM
    - Project advisors
    - Project coordinator

## 3.4 Assumptions and Constraints

To enable each project team the freedom to operate as independent entities while also meeting the expectations levied by the Project Organizer, DevSecOps operates under the following assumptions and constraints:

•[Assumption] Timetables: As mentioned earlier, any provided timetable and schedules are heavily influenced by the ability of individual teams to formulate and deliver software.

•[Assumption] Production Value: All software architectures and implementation are designed based on providing proof-of-concept and product prototyping.

•[Constraint] Architecture Requirements: Unless prior coordinated with the Project Coordinator, the selection of specific technologies is limited to this expressed in earlier requirements documents.

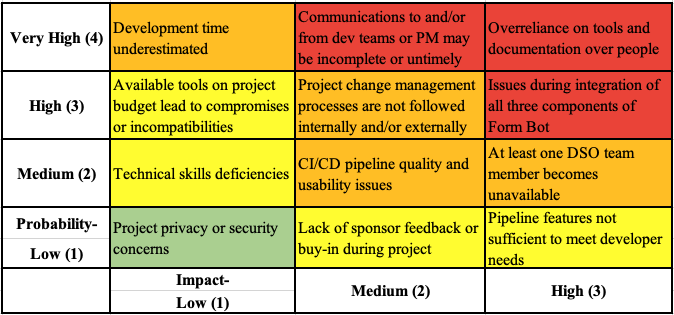
•[Constraint] Capstone Resources: Unless otherwise offered by the Project Sponsors, the Capstone provides no financial support for development efforts.

## 3.5 Risk Analysis

Identified risks run the gamut from internal, external, to project management related risks. Some of the major risks Form Scriber DevSecOps identified relate to potential shortcomings in inter- and intra-team communications because of the nature of the development environment, assumptions of familiarity with and reliance on tools and documentation, and possible difficulties during components integration. The below Figure 3 summarizes these and other risks, ranked by probability and impact.

**Figure 3**

*DevSecOps Risk Probability and Impact Matrix*



**Figure 3.** DevSecOps Risk Probability and Impact Matrix

*Note.* The DSO team shall need to ensure that development team members are informed about DevSecOps best practices and ensure lines of communication are highly accessible when needed, given the challenges of varied development schedules of teams.

# 4. Test Plan

## 4.1 Introduction

### 4.1.1 Purpose

The purpose of this section is to define the strategy and parameters to apply to testing of the Form Scriber CI/CD pipeline. The test plan will define the following:

* Coverage– Focus areas of the test plan to include definitions of system verification
* Software and hardware requirements – Detail the test environments and dependencies required to conduct the testing
* Test strategy - Testing approach and test case design criteria
* Test cases – The individual test cases that comprise the test plan
* Assumptions and constraints – The assumptions and limitations regarding the testing

### 4.1.2 Coverage

Functional testing of the CI/CD pipeline features shall be included. Due to the pre-defined nature of a DevSecOps pipeline, test coverage is limited to the core functionalities. Development operations processes run by the pipeline falls under the testing scope. The tester does not have access to nor needs knowledge of the internal code logic of the git implementation of GitHub, the build and deploy pipelines of Azure, and those of the container and container orchestration implementations as it is not these that are under test. Verification of the pipeline functionality is to ensure output matches expected results. In-scope components include actions such as building, testing, scanning, and deployment of the Form Scriber components. Out-of-scope components include, and are not limited to, testing activities such as security or penetration testing, localization testing, or usability testing.

Dialogflow and Mobile teams have developed the test plan in their Project Plan documentation which focuses on UI (User Interface) and end to end testing for the application. Most test cases are being done manually. The development teams have planned to write some unit testing into their solutions. DSO will work with development teams to ensure the pipeline runs and checks for unit tests with code coverage. However, DSO is not responsible to create unit test cases in the solution. DSO will rely on the test team from Dialogflow and Mobile to write the unit testing in the solution so that the pipeline can run directly from the solution.

## 4.2 Requirements

### 4.2.1 Software Requirements

The test environment shall consist of a desktop operating system with a modern web browser. The CI/CD pipeline components shall be accessed via aforementioned browser. Testing shall be done on macOS 10.15.7 and Chrome browser Version 88.0.4324.150.

### 4.2.2 Hardware Requirements

Testing shall be conducted on a MacBook Pro laptop computer, model A1707. Keyboard input shall be done with the built-in keyboard and other inputs shall be done with the built-in trackpad.

## 4.3 Testing

### 4.3.1 Test Strategy

Functional testing will be the main form of testing for this cycle. Due to the nature of the pipeline, which is dependent on development team inputs in the form of source code, and by which the source code is also dependent on said pipeline to build, test, and deploy it, testing is conducted when pipeline run(s) are initiated, either manually or automatically. Test cases will be designed to represent use-case scenarios as by DevSecOps personnel and other development project stakeholders. The test suite shall be validated to ensure that coverage is satisfactory for features and possible scenarios including positive and negative cases when possible, and permutations of selectable options. The actual results shall be compared to expected results to determine test case outcome in a later Test Report. At the time of this writing, development team is in the process of automating their testing requirements including unit and end-to-end tests. The automated testing may not be completed during this project which will be considered for the future works.

### 4.3.2 Proposed Test Cases

#### 4.3.2.1 Test Case 01:

|  |  |
| --- | --- |
| **Test Case Name** | **VersionControl-Merge\_develop\_branch-Developer** |
| **Description** | Verify merging to develop branch by a non-lead developer requires a pull request and code review/approval by lead developer. |
| **Requirements** | REQ-1.1 |
| **Prerequisites** | The development team’s repository is set up with the required branch protection rules and the developer has invited account to the organization and team. |
| **Steps** | 1. As developer, create code change and make an integration/feature branch. 2. Push the code to the remote repository. 3. Attempt merge to develop branch |
| **Expected output** | Merging to develop branch by non-lead developer is not possible. Developer must create a pull request for review, which notifies the developer lead. |

#### 4.3.2.2 Test Case 02:

|  |  |
| --- | --- |
| **Test Case Name** | **VersionControl-Merge\_develop\_branch-DeveloperLead** |
| **Description** | Verify merging to develop branch by a lead developer can be done without additional pull request or code review. |
| **Requirements** | REQ-1.2 |
| **Prerequisites** | The development team’s repository is set up with the required branch protection rules and the developer lead has invited account to the organization and team and elevated permissions as set by the DSO team. |
| **Steps** | 1. As developer lead, create code change and make an integration/feature branch. 2. Push the code to the remote repository. 3. Attempt merge to develop branch. |
| **Expected output** | Merging to develop branch by lead developer is successful. |

#### 4.3.2.3 Test Case 03:

|  |  |
| --- | --- |
| **Test Case Name** | **VersionControl-Merge\_master\_branch-DevTeam** |
| **Description** | Verify merging to master branch by the development team requires a pull request and code review/approval by the DSO team. |
| **Requirements** | REQ-1.1 |
| **Prerequisites** | The development team’s repository is set up with the required branch protection rules and the developer has invited account to the organization and team. |
| **Steps** | 1. As developer, attempt merge to master branch. |
| **Expected output** | Merging to master branch directly by development team is not possible. Developer must create a pull request for review, which notifies the DSO team. |

#### 4.3.2.4 Test Case 04:

|  |  |
| --- | --- |
| **Test Case Name** | **VersionControl-Merge\_master\_branch-DsoTeam** |
| **Description** | Verify merging to master branch by DSO team can be done without additional pull request or code review. |
| **Requirements** | REQ-1.2 |
| **Prerequisites** | The development team’s repository is set up with the required branch protection rules and the DSO team member has elevated permissions as set by the DSO team. |
| **Steps** | 1. As DSO team member, attempt merge to master branch. |
| **Expected output** | Merging to master branch by DSO team is successful. |

#### 4.3.2.5 Test Case 05:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationBuild-Manual\_execution** |
| **Description** | Verify the CI/CD pipeline application build tasks run when the build feature is manually executed. |
| **Requirements** | REQ-2.1 |
| **Prerequisites** | The CI pipeline is integrated with the remote code repository. |
| **Steps** | 1. As DSO team member, manually kick off the application build tasks. |
| **Expected output** | Application build tasks run successfully and notifications are sent. |

#### 4.3.2.6 Test Case 06:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationBuild-Automated\_execution** |
| **Description** | Verify the CI/CD pipeline application build tasks run when changes are made to the repository’s master branch. |
| **Requirements** | REQ-2.1 |
| **Prerequisites** | The CI pipeline is integrated with the remote code repository. |
| **Steps** | 1. As DSO team member, merge integration branch into master branch. |
| **Expected output** | Application build tasks run successfully and notifications are sent. |

#### 4.3.2.7 Test Case 07:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationTest-Manual\_execution** |
| **Description** | Verify the CI/CD pipeline application test tasks run when the application testing feature is manually executed. |
| **Requirements** | REQ-3.1 |
| **Prerequisites** | Test code is integrated with the CI pipeline. |
| **Steps** | 1. As DSO team member, manually kick off the application test tasks. |
| **Expected output** | Application testing tasks run successfully and notifications are sent. |

#### 4.3.2.8 Test Case 08:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationTest-Automated\_execution** |
| **Description** | Verify the CI/CD pipeline application test tasks run after build tasks are complete. |
| **Requirements** | REQ-3.1 |
| **Prerequisites** | Test code is integrated with the CI pipeline. |
| **Steps** | 1. Automated testing is triggered upon build tasks completing. |
| **Expected output** | Application testing tasks run successfully and notifications are sent. |

#### 4.3.2.9 Test Case 09:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationAnalysis-Manual\_execution** |
| **Description** | Verify the CI/CD pipeline application code analysis tasks run when the application code analysis feature is manually executed. |
| **Requirements** | REQ-4.1 |
| **Prerequisites** | Code analysis tools are integrated with the CI pipeline. |
| **Steps** | 1. As DSO team member, manually kick off the application code analysis tasks. |
| **Expected output** | Application code analysis tasks run successfully and notifications are sent. |

#### 4.3.2.10 Test Case 10:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationAnalysis-Automated\_execution** |
| **Description** | Verify the CI/CD pipeline application code analysis tasks run after testing tasks are complete. |
| **Requirements** | REQ-4.1 |
| **Prerequisites** | Code analysis tools are integrated with the CI pipeline. |
| **Steps** | 1. Automated code analysis is triggered upon testing tasks completing. |
| **Expected output** | Application code analysis tasks run successfully and notifications are sent. |

#### 4.3.2.11 Test Case 11:

|  |  |
| --- | --- |
| **Test Case Name** | **Triggers-Enable\_continuous\_integration** |
| **Description** | Verify when enabling continuous integration, the CI pipeline tasks run automatically when code changes are made to repository master branch. |
| **Requirements** | REQ-2.1, REQ-3.1, REQ-4.1 |
| **Prerequisites** | The CI pipeline is integrated with the remote code repository. |
| **Steps** | 1. As DSO team member, turn on the continuous integration flag. 2. Merge integration branch into master branch. |
| **Expected output** | Application build, test, and code analysis tasks run in sequence successfully and notifications are sent. |

#### 4.3.2.12 Test Case 12:

|  |  |
| --- | --- |
| **Test Case Name** | **Triggers-Disable\_continuous\_integration** |
| **Description** | Verify when disabling continuous integration, the CI pipeline tasks do not run automatically when code changes are made to repository master branch. |
| **Requirements** | REQ-2.1, REQ-3.1, REQ-4.1 |
| **Prerequisites** | The CI pipeline is integrated with the remote code repository. |
| **Steps** | 1. As DSO team member, turn off the continuous integration flag. 2. Merge integration branch into master branch. |
| **Expected output** | Application build, test, and code analysis tasks do not run automatically. |

#### 4.3.2.13 Test Case 13:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationDeployment-Manual\_execution** |
| **Description** | Verify the CI/CD pipeline application deployment tasks run when the application deployment feature is manually executed. |
| **Requirements** | REQ-5.1 |
| **Prerequisites** | Deployment environments are integrated with the CD pipeline. |
| **Steps** | 1. As DSO team member, manually kick off the application deployment tasks. |
| **Expected output** | Application deployment tasks run successfully and notifications are sent. |

#### 4.3.2.14 Test Case 14:

|  |  |
| --- | --- |
| **Test Case Name** | **ApplicationDeployment-Automated\_execution** |
| **Description** | Verify the CI/CD pipeline application deployment tasks run after code analysis tasks are complete. |
| **Requirements** | REQ-5.1 |
| **Prerequisites** | Deployment environments are integrated with the CD pipeline. |
| **Steps** | 1. Automated application deployment is triggered upon code analysis tasks completing. |
| **Expected output** | Application deployment tasks run successfully and notifications are sent. |

#### 4.3.2.15 Test Case 15:

|  |  |
| --- | --- |
| **Test Case Name** | **Triggers-Enable\_continuous\_deployment** |
| **Description** | Verify when enabling continuous deployment, the CD pipeline tasks run automatically after CI tasks are completed. |
| **Requirements** | REQ-5.1 |
| **Prerequisites** | Deployment environments are integrated with the CD pipeline. |
| **Steps** | 1. As DSO team member, turn on the continuous deployment flag. 2. Merge integration branch into master branch. |
| **Expected output** | Application deployment tasks run successfully after CI tasks and notifications are sent. |

#### 4.3.2.16 Test Case 16:

|  |  |
| --- | --- |
| **Test Case Name** | **Triggers-Disable\_continuous\_deployment** |
| **Description** | Verify when disabling continuous deployment, the CD pipeline tasks do not run automatically after CI tasks are completed. |
| **Requirements** | REQ-5.1 |
| **Prerequisites** | Deployment environments are integrated with the CD pipeline. |
| **Steps** | 1. As DSO team member, turn off the continuous deployment flag. 2. Merge integration branch into master branch. |
| **Expected output** | Application deployment tasks do not run automatically after CI tasks. |

## 4.4 Assumptions and Constraints

### 4.4.1 Assumptions

Testing assumes that the test environment is available at the time of execution. It is assumed that all the development teams will provide functioning source code for which the CI/CD pipeline is dependent upon. It is assumed that the providers of the chosen DevSecOps tools shall not change usage allowances to free and student tier accounts during the duration of the project. The requirements stipulated in the Form Scriber DevSecOps Software Requirements Specification document shall represent the requirements under which the pipeline shall be tested and not change for the duration of the test cycle.

4.4.2 Constraints

The test lifecycle shall take place within the latter half to quarter of the UMGC Capstone, coinciding with other development team activities, encompassing just a few weeks/sessions with uncontrolled variables and compatibility. There shall be no traditional software testing lifecycle or bug tracking/fixing processes because of the nature of DevSecOps projects as infrastructure services.

# 5. Future Work

The below items have been identified as possible additions or improvements to guide future development phases for the FormScriber DevSecOps project.

* + Advance Development Factory version for Dart/Flutter for the Mobile project
  + Further ADF integration into development projects
  + Code quality/compliance metrics and analytics
  + Switch over to Azure Container Registry from Docker Hub
  + SonarQube integration for Mobile component
  + Add the automated test tasks to the Azure DevOps pipeline when development has developed the automated testing.
  + AKS deployment directly from the release pipeline for FormScriber components
  + Add Android APK signing and Google Play release to the Azure pipeline
  + Add Security scanning for Docker containers

# 6. Project Schedule

