Advanced Development Factory: Technical Design Document

Jeroen Soeurt, Michelle Monfort, and Robert Wilson
University of Maryland Global Campus
Summer 2021

UMGC Capstone Project
Advanced Development Factory
Technical Design Document
Version 1.0

Template Based in part on: Technical Design Document Template.docx - Google Drive

Name: Jeroen Soeurt, Michelle Monfort, and Robert Wilson
SWEN670 9040
Date: 06/20/2021

Revision History

Date	Version	Description	Author
06/20/2021	1.0	Initial document	Robert Wilson
07/18/2021	1.1	Revise Test Plan	Robert Wilson

Table of Contents

1 INTRODUCTION

1.1 Business Requirement

The purpose of this document is to provide the software engineer team a guide that establishes the technical aspects of what the products must do and how it does it. Specifically, this document aims to cover how the consideration of the Advanced Development Factory (ADF) and how the engineers on the team should go about solving the problems.

- Functional
  - ADF must run within a docker support container
  - ADF must support Remote Desktop Protocol
  - ADF must be configurable to support a variety of program languages
  - ADF must support local debug operations
  - ADF must have connectivity to GitHub
  - ADF must support bi-direction sound

- Non-Functional
  - ADF is an Ubuntu Linux container
  - ADF is ephemeral
  - ADF supports JAVA, .Net, and Flutter out of the box
  - ADF supports IDE configuration via the associated shell script

1.1.1.1 *Timings & Frequency*

- The dockerfile containers version of software and should be updated quarterly
- Verification of the docker container should be checked after each version update to ensure combability before the main branch is updated. This should happen quarterly

1.1.1.2 *Security*

- Version of software should be evaluated monthly for vulnerabilities accessed to be high risk by Mitre: CVE - CVE (mitre.org)

1.1.1.3 *Training*

- Assumes some level of prior develop skill
- Training to be delivered in two parts
    - Video training series accessible via YouTube
    - User Guide

2 Technical Design

2.1 Technical Approach Scope

- In Scope

The ADF concept aims to provide students of the University of Maryland Global Campus (UMGC) required to complete the development of web, desktop, or mobile applications a common way to accomplish their tasks without having to account for individual environmental requirements. The scope of the project includes all configuration files to manage the CI/CD pipeline for the ADF project and the configuration files of the project itself. The project itself will be one dockerfile that is used within the deployment pipeline to create Docker Images within a container hosting service such as AKS. Other files that are within scope are script files that either manage the hosting service or run as part of the logon and logoff scripts of the hosted container.

- Out of Scope

Although a dependency of the project, the installation, and management of Docker on hosting machines is out of scope for this project. Specific interactions with associated

applications are also out of scope for this document but can be referenced as part of the user guide.

2.2 Other Affected Applications

- GIT
- VS Code
- Pulse Audio
- XRDP
- XFCE4

2.3 Assumptions

- Docker supported hyper visor is running on either the customer's computer or available from a cloud provider
- Develops are familiar with basic concept of version control

2.4 Issues

- RDP experience varies if running the docker from a remote source and is largely based upon geographical location of the cloud hosting the container and the user's available bandwidth.
- Clipboard copy and paste between the client the RDP session does not always work as intended and should not be relied upon
- This concept relies upon VS Code which is open source and maintained by the community. The extensions available for VS Code to meet the demand of the class are also maintained by the community and may not specifically support the technology requested by the education program.

2.5 Risks

- Time zone difference
  - The time zone difference of 9 hours cannot reasonably be reduced, as we all have other commitments in our regions (employment). We will work around this by scheduling meetings at night in Europe / in the morning on the west coast.
- Changing requirements
  - Looking at the previous GitHub repositories, it is clear that the programming language used over the past year has varied a lot. The

solution for the ADF must be adjustable to future changes. We plan to use CLI tools, so that they can easily be replaced, and an IDE with support for more than one language.

- Short project timeline
  - It is a challenge to complete all this work in the duration of a single class. Furthermore, the pipeline is essential to maintaining the development teams code repositories. We will address this by first implementing the pipeline, and then working on the ADF.

- Limited experience
  - Two out of three members have limited experience with CI/CD. The risk is that the wrong decisions can be made early in the project, based on limited experience on knowledge. This issue is addressed by assigning the project manager position to the team member with the most experience and knowledge in this area.

3 Design Elements - ADF

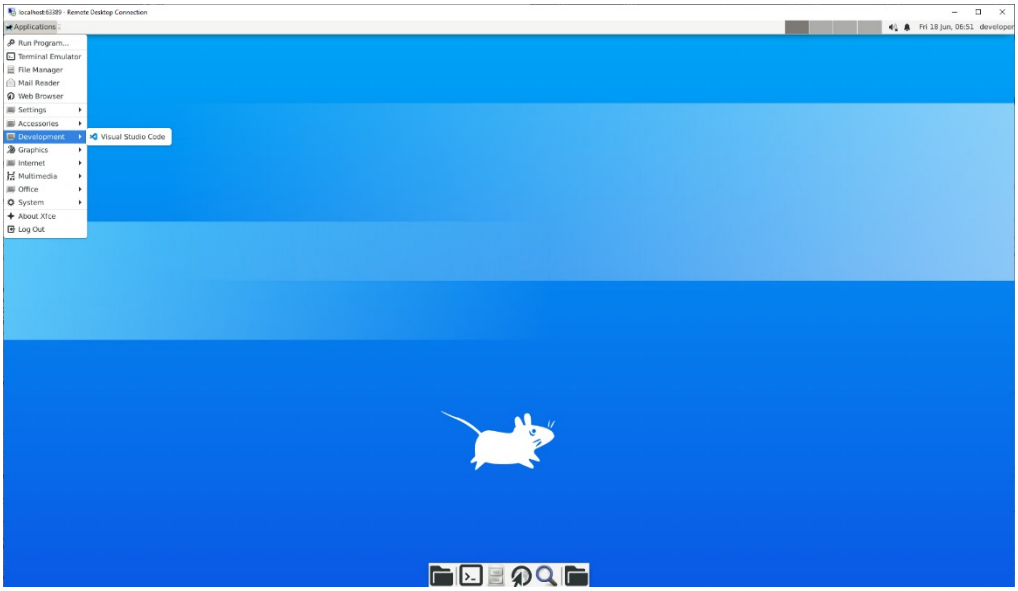
3.1 Online Objects

- Project Details
  - ADF

- Components
  - GIT
  - VS Code
  - Android Emulator
  - Android SDK
  - Flutter Command Line Utilities
  - Dart SDK
  - AKS
  - Docker
  - .Net Runtime
  - .Net SDK
  - Pulse Audio

- XRDP
- XFCE4

- Pages
  The aim of this project is to provide a development environment to the end-users. This is down via configuration only. The expected outcome provides the user an environment to which they can connect via RDP. Upon Connecting the below should be seen:



3.2 Batch Programs

- Batch Program: startup.sh which enables boot time workflows of the docker image providing the development environment to the user.
  - Deliver runtime environment variables
  - Configure and load Pulse Audio
  - Enable RDP session
  - Install VS Code extensions
  - Stop at entry point: /bin/bash

4 Design Elements – Project Proposal Site

The Project Proposal Website was started as a product of the 2020 Fall Semester class. The project has been revived in this semester and renamed to the Project Proposal Site. The overall purpose remains the same and to that extent the same markup will be used to develop the site.

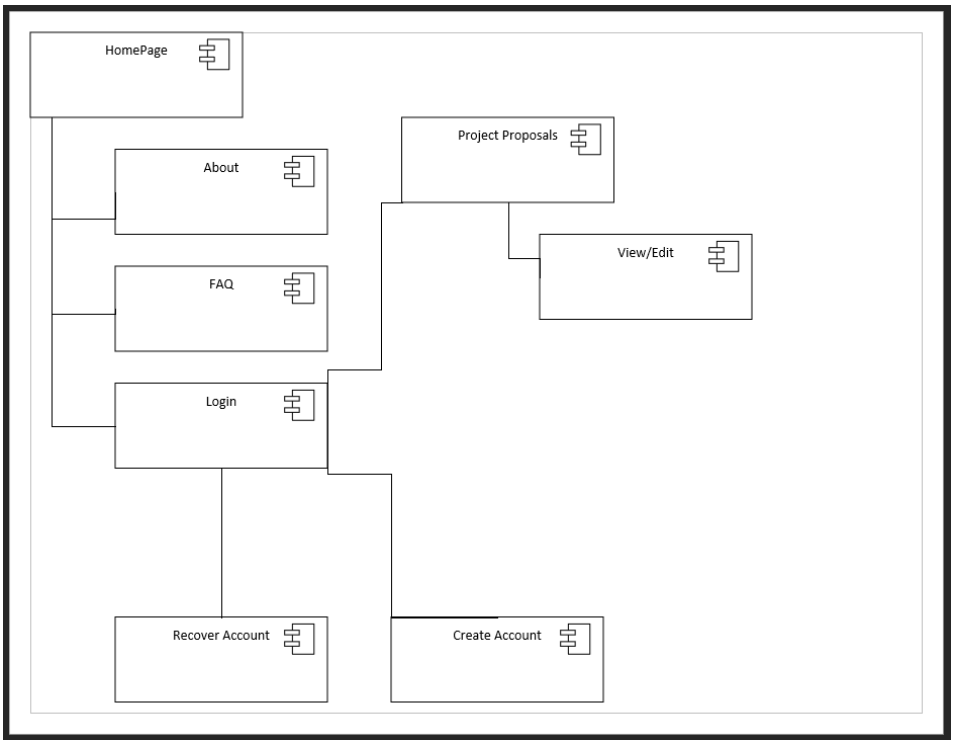
The difference? In this version of the project, complexity is reduced from mixing Angular and Spring Root (Java) based frameworks for just the Blazor feature of the ASP.NET framework. In addition to reducing complexity, in most cases the .Net framework out performs Java based project which reduces TCO and provides a better experience for the customer.

4.1 Online Objects

- Project Details
  Blazor is a feature of ASP.NET, it is open source, cross-platform, and runs easily from a Docker container. A Blazor project handles 90% of code in C#, and the rest is a mix of HTML and CSS. There is no longer a need for JavaScript along with some language running on the backend server. Because the Blazor project is C#, the development time can be greatly reduced and the tooling is well supported by a growing number of Open-Source contributors and Microsoft.

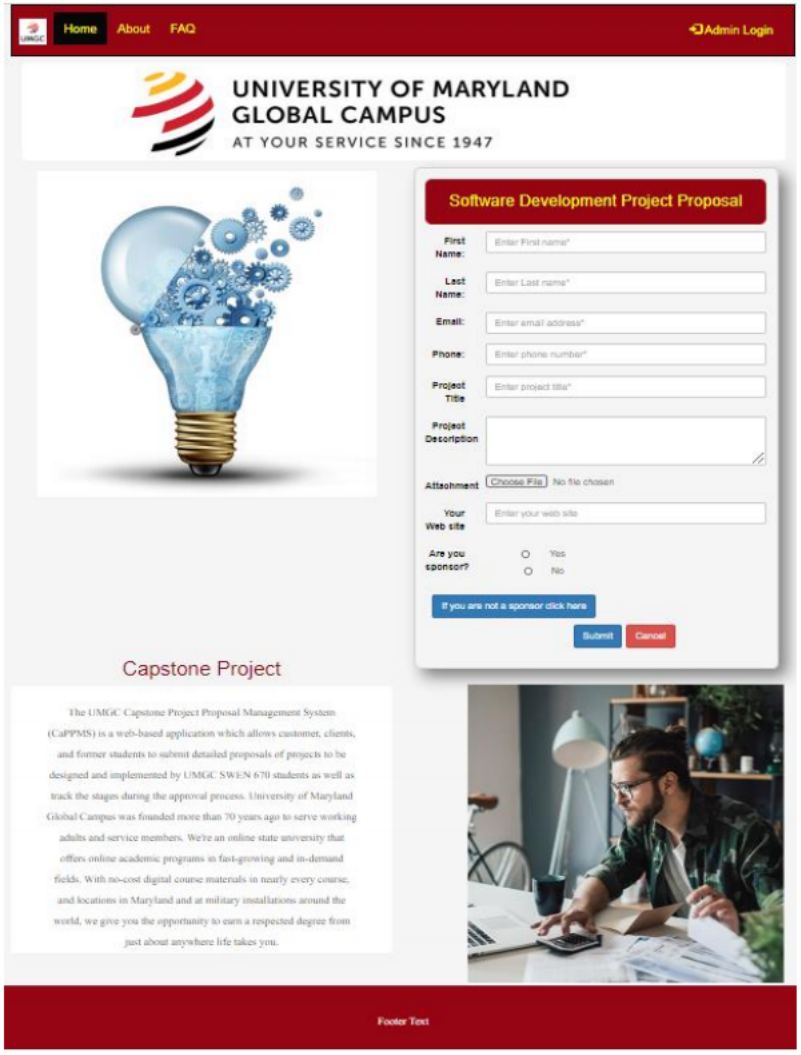
- Menus
  A Common header can be found across all the pages. Three buttons on the left: Home, About, and FAQ. One button along the right side which is a toggle state of either Admin Login or Logout.
  Below the button row is UMGC global branding header image.

- Components
  Considering the web application, a component will be identified as a reusable element that can be rendered independent of a specific page given the correct conditions.
  - DisplayRepos
  - Idea
  - LoginDisplay
  - MainLayout
  - NavMenu
  - SubmitIdea
  - UMGCFooter
  - UMGCHeader



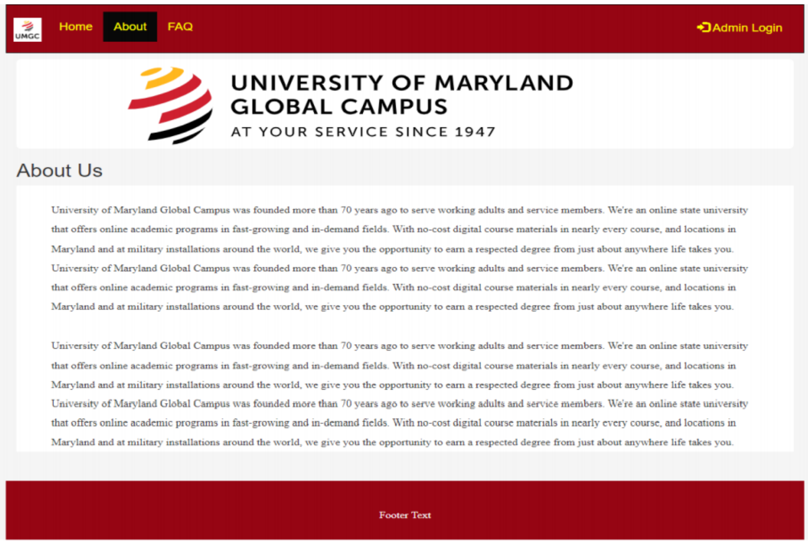
- Pages
  There are four pages to this application. They are Home, About, Proposed Project List, and FAQ.

    - Home:



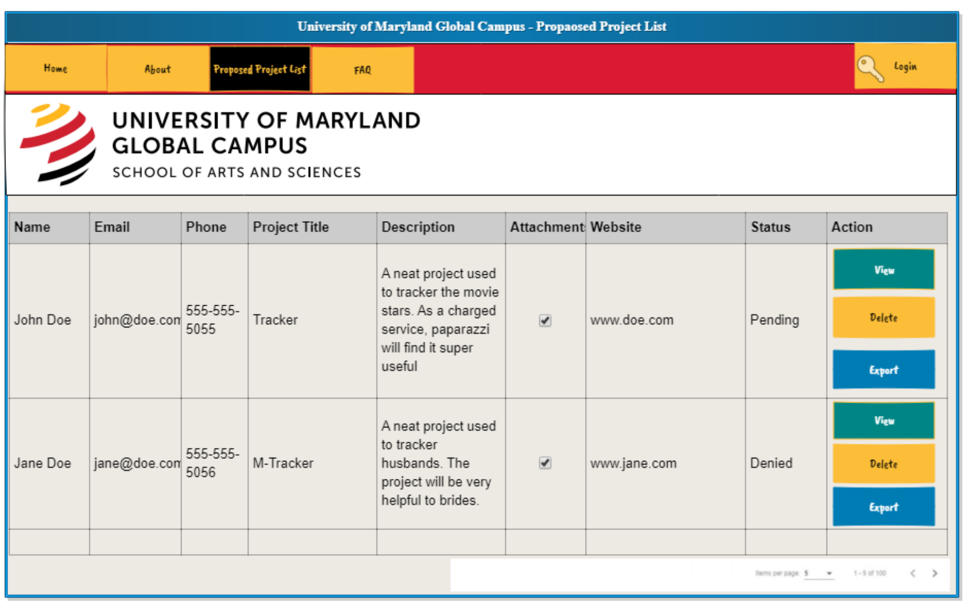
Purpose:	The Home page is where would be dreamers come to submit an idea. This is the default page as it is the sole reason for this project to exist.
Navigation & User Interaction	The user can login for admin pages, or explore the about page to find out why we do this, or maybe check out a recently asked question.

- About:



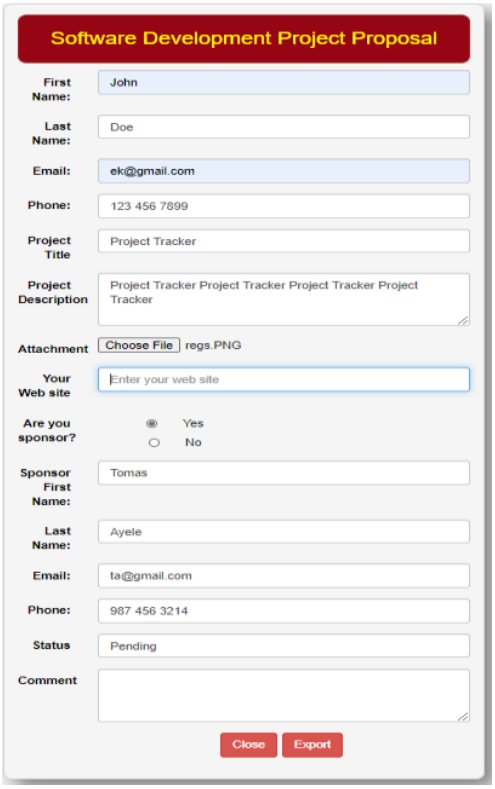
Purpose:	The about page is to help users understand who we are and maybe a little bit about why we do this.
Navigation & User Interaction	The user can login for admin pages, or head back to the home page and submit an idea, or maybe check out a recently asked question.

- Proposed Project List:



Purpose:	This page gives authorized users a nice view of submitted projects
Navigation & User Interaction	The user can login for admin pages, or head back to the home page and submit an idea, or maybe check out a recently asked question.

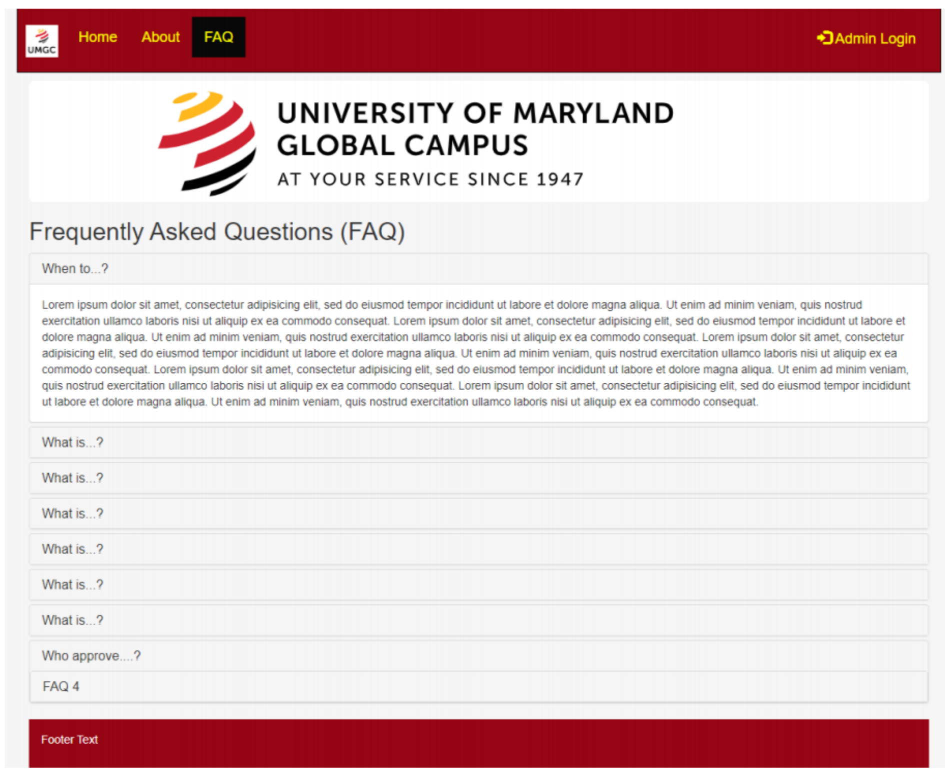
- Project View



Purpose:	The project view is a nice popup on top of the project list when the user decides to view a project. The user can edit straight form the view if they wish, open attached files, attach more files, and export the important data of the project to future classes if approved.

Navigation & User Interaction	The close button closes the view. After editing, a Save button will appear allowing the user to save the view and the close button will become a cancel button. The export button will export all fields except status and comments into a text file, then zip the text file and attachments into a download for the user.

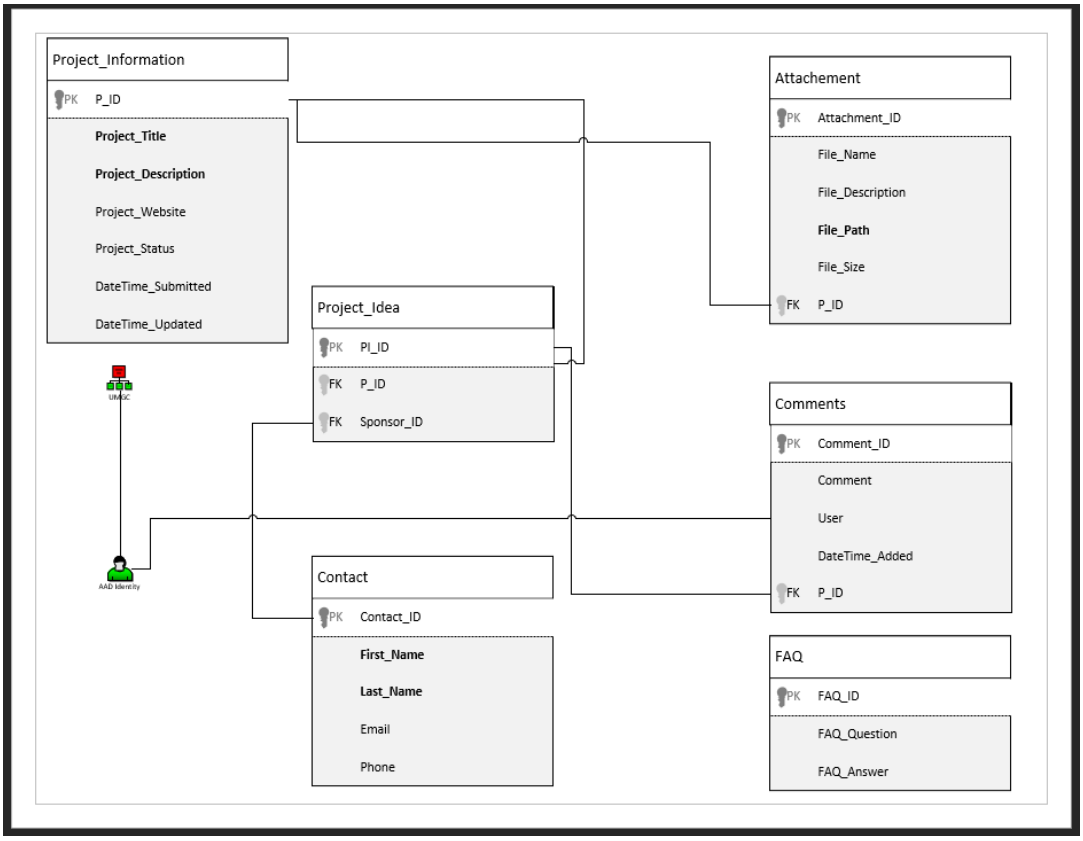
- FAQ:



Purpose:	The FAQ page is all about posting answers to some commonly asked questions. Like when do these ideas

	come to life, and is there an approval process. Will my supporting documents get eaten my werewolves? All the things curious minds want to know.
Navigation & User Interaction	The user can login for admin pages, or head back to the home page and submit an idea, or maybe checkout the about page to find out who we are and what we are about.

- Database



Purpose:	This entity diagram describes the relation of object that will be stored in a persistent Azure SQL Database. UMGC Represent the AAD Organization object. AAD Identity represents a single authenticated user.

- FAQ

```
CREATE TABLE [dbo].[FAQ]
(
    [FAQ_ID] INT NOT NULL PRIMARY KEY,
    [FAQ_Question] NVARCHAR(50) NOT NULL,
    [FAQ_Answer] NVARCHAR(MAX) NULL
)
```

- Contact

```
CREATE TABLE [dbo].[Contact]
(
    [Contact_ID] BIGINT NOT NULL PRIMARY KEY,
    [First_Name] NVARCHAR(50) NOT NULL,
    [Last_Name] NVARCHAR(50) NOT NULL,
    [Email] NVARCHAR(50) NULL,
    [Phone] NCHAR(10) NULL
)
```

- Project_Information

```
CREATE TABLE [dbo].[Project_Information]
(
    [P_ID] BIGINT NOT NULL PRIMARY KEY,
    [Project_Title] NVARCHAR(50) NOT NULL,
    [Project_Description] NVARCHAR(MAX) NULL,
    [Project_Website] NCHAR(10) NULL,
    [Project_Status] NCHAR(10) NULL
)

GO

CREATE INDEX [Index_Project_Title] ON [dbo].[Project_Information] (
[Project_Title])
```

- Comments

```
CREATE TABLE [dbo].[Comments]
(
    [Comment_ID] BIGINT NOT NULL PRIMARY KEY,
    [Comment] NVARCHAR(MAX) NOT NULL,
    [User] NVARCHAR(50) NOT NULL,
    [DateTime_Added] DATETIME NOT NULL,
    [P_ID] BIGINT NOT NULL,
    CONSTRAINT [FK_Comments_Project_Information] FOREIGN KEY ([P_ID
]) REFERENCES [Project_Information]([P_ID])
```

```
)
```

- Project_Idea

```
CREATE TABLE [dbo].[Project_Idea]
(
    [PI_ID] BIGINT NOT NULL PRIMARY KEY,
    [P_ID] BIGINT NOT NULL,
    [Submitted_ID] BIGINT NOT NULL,
    [Sponsor_ID] BIGINT NOT NULL,
    CONSTRAINT [FK_Project_Idea_Project_Information] FOREIGN KEY ([
P_ID]) REFERENCES [Project_Information]([P_ID]),
    CONSTRAINT [FK_Project_Idea_Contact] FOREIGN KEY ([Sponsor_ID])
 REFERENCES [Contact]([Contact_ID])
)
```

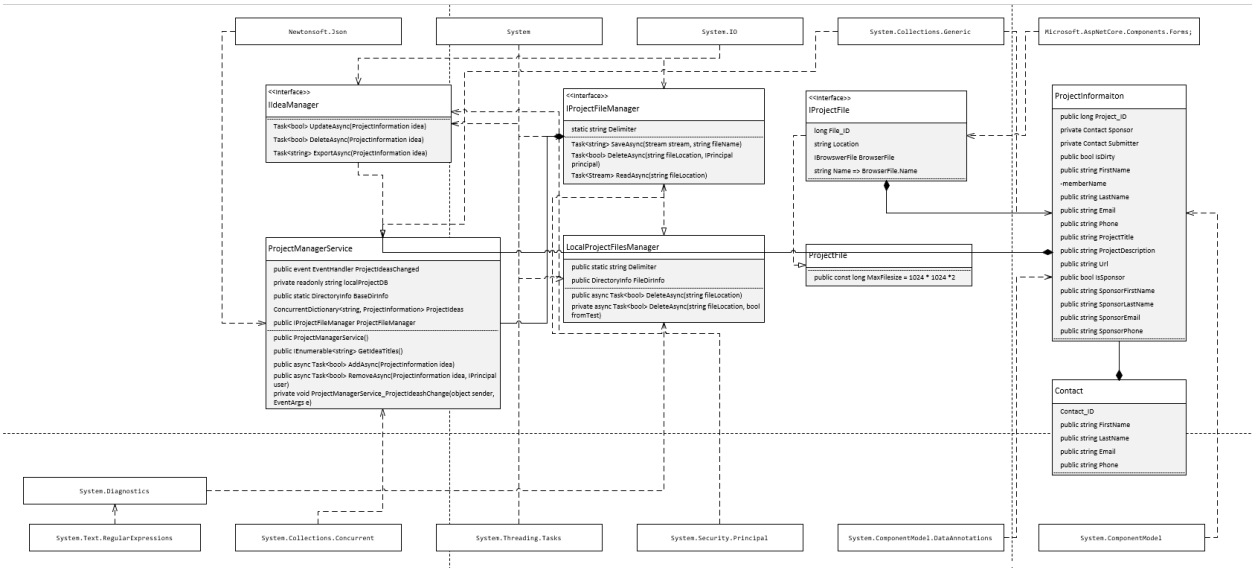
- Code
  Code will be developed utilizing ASP.NET utilizing dotnet 5.0. This version of dotnet is platform agnostic which allows the developer to use their operating system of choice.

  Dotnet use C# as the language which has extensive documentation at https://docs.microsoft.com/en-us/dotnet/. The user experience is developed with a mixture of basic html and C#.

  Components are designed by using a Razor type file and usually located in the Shared folder, while pages are also usually Razor type with the @page attribute at the top of the file and usually located in the Pages folder.

  Services usually provide back-end communication from the application to other applications. They are developed in C# without HTML and can be found in the Data folder. The compile into DLL files which can be loaded by the backing ASP.NET application. Services are injected into the page using the @inject attribute and identified in the Program.cs file as a service.

- Class Diagram



- Message Catalog
  CaPPMS uses SignalR for the communication channel between user and server. The protocol between the client and server are secure via https protocol with TLS 1.2.

- Process Definition

- Security Requirements
  - Only authorized users can delete proposed projects

  - Files should be scanned for malicious content before put to final location

4.2 Batch Programs

- Batch Program: None

5 Testing

The engineering team should be intimately familiar with the overall structure of ADF, but the actual usage of ADF is what needs to be tested. The premise of the tests will be conducted as functional tests that cover the following axes:

- Functionality

- External Interfaces

- Input Space
- Output Space
- Configuration elements
- Performance

5.1 Test Plan

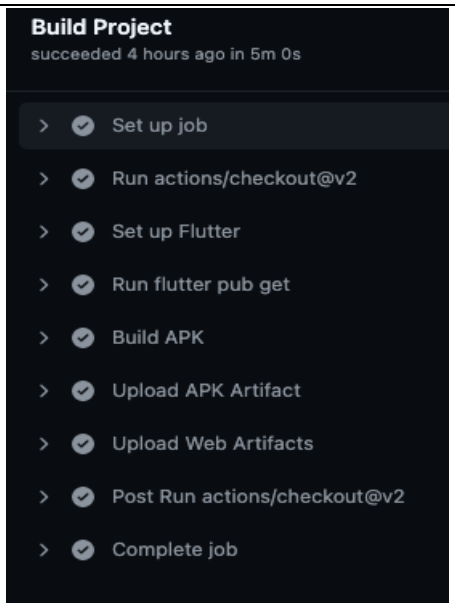
Humans are fallible, as it the code that is written to provide instruction to a computer system. With this in mind we have to assume that any change to current code will break something as the current code is fallible. It is unlikely that enough test can be written that will simply ensure a product that does not faulter. This is an un-realistic approach to testing. It also needs to be considered that the DevSecOps team is responsible for multiple projects within their own team and ensuring the build pipeline success for the use of development teams' collaboration.

- Build Pipeline:

DevSecOps team is responsible for the develop teams' pipelines. As such, any pipeline configuration change must be monitored for completion. The process of the pipeline must be considered to be faulty until completely validated against the contract with the development team. The basic elements of the pipeline must be able to analyze, build, test, and post artifacts from the build.

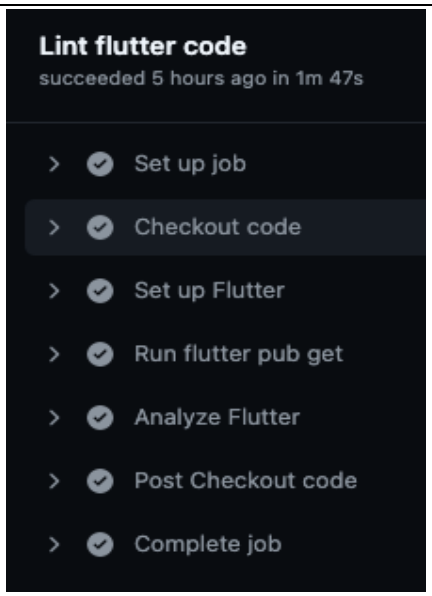
A configuration change to the pipeline could result in failures in any of these mention categories. A validation of each category is the only way a change in configuration can be validated. It is equally important that the pipeline be validated occasionally after branch merge operations. A change in the development team's branch may affect how the pipeline conducts its own processes. For an example, if the development team chooses to change the base folder of their project, the pipeline may no longer build the project correctly.

- ***(Charlie) Pipeline builds successfully***

Item	**Description**
Title	Pipeline builds successfully
Description	Tests if the Pipeline builds successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click build
Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run flutter pub get => Build APK => Upload APK Artifact => Upload Web Artifacts => Post Run actions/checkout@v2 => Complete job
Image	

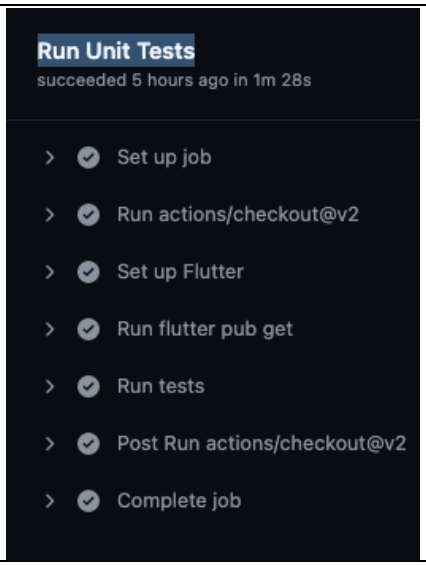
- ***(Charlie) Pipeline Lint flutter code successfully***

Item	**Description**

Title	Pipeline Lint flutter code successfully
Description	Tests if the Pipeline Lints flutter code successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click Lint flutter code
Expected Result	Artifacts Produced during runtime: => Set up job => Checkout code => Set up Flutter => Run flutter pub get => Analyze Flutter => Post Checkout code => Complete job
Image	

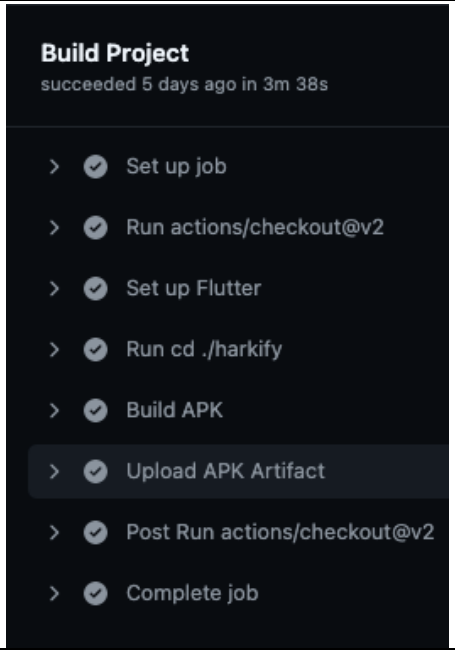
- ***(Charlie) Pipeline Runs Unit Tests successfully***

Item	**Description**
Title	Pipeline Runs Unit Tests Successfully
Description	Tests if the Pipeline Runs Unit Tests successfully.

Input Data	See GitHub Actions Double click most recent workflow Double click Run Unit Tests
Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run flutter pub get => Run tests => Post Run actions/checkout@v2 => Complete job
Image	

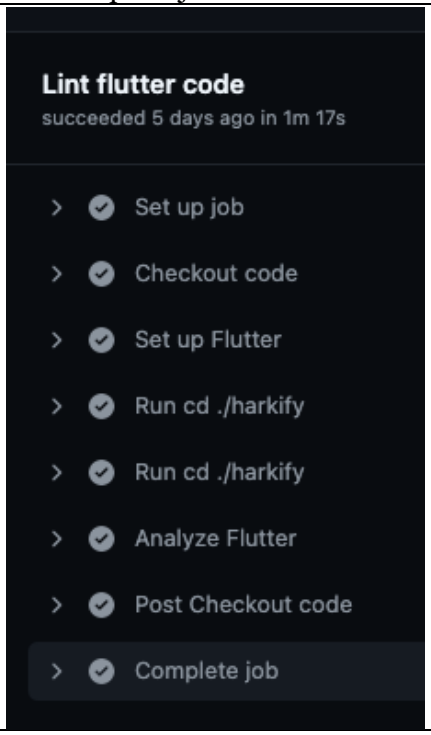
- ***(Bravo) Pipeline builds successfully***

Item	**Description**
Title	Pipeline builds successfully
Description	Tests if the Pipeline builds successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click build

Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run cd ./harikfy => Build APK => Upload APK Artifact => Upload Web Artifacts => Post Run actions/checkout@v2 => Complete job
Image	

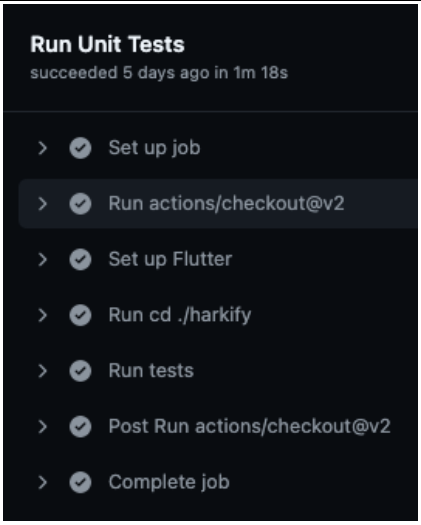
- ***(Bravo) Pipeline Lint flutter code successfully***

Item	**Description**
Title	Pipeline Lint flutter code successfully
Description	Tests if the Pipeline Lints flutter code successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click Lint flutter code

Expected Result	Artifacts Produced during runtime: => Set up job => Checkout code => Set up Flutter => Run cd ./harikfy => Run cd ./harikfy => Analyze Flutter => Post Checkout code => Complete job
Image	

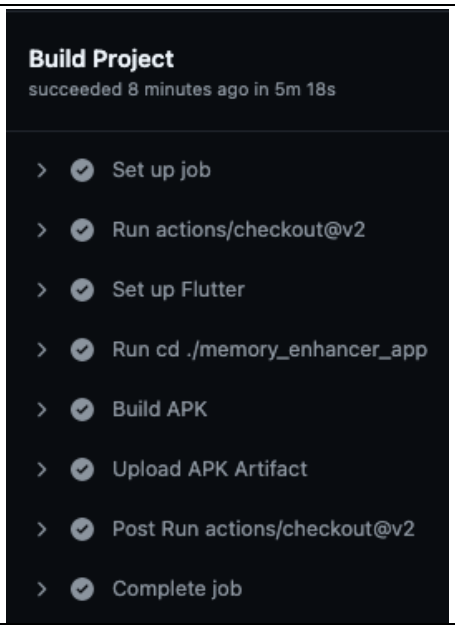
- ***(Bravo) Pipeline Runs Unit Tests successfully***

Item	**Description**
Title	Pipeline Runs Unit Tests Successfully
Description	Tests if the Pipeline Runs Unit Tests successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click Run Unit Tests

Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run cd ./harikfy => Run tests => Post Run actions/checkout@v2 => Complete job
Image	

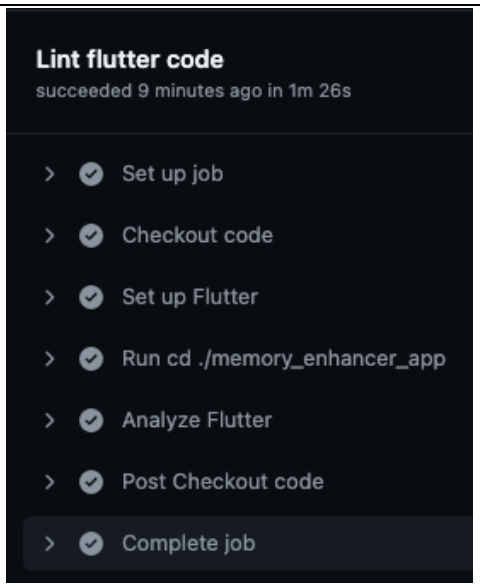
- ***(Amazing) Pipeline builds successfully***

Item	**Description**
Title	Pipeline builds successfully
Description	Tests if the Pipeline builds successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click build

Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run cd ./memory_enhancer_app => Build APK => Upload APK Artifact => Post Run actions/checkout@v2 => Complete job
Image	

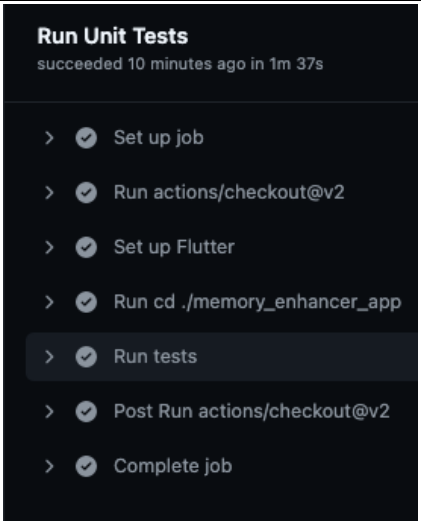
- ***(Amazing) Pipeline Lint flutter code successfully***

Item	**Description**
Title	Pipeline Lint flutter code successfully
Description	Tests if the Pipeline Lints flutter code successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click Lint flutter code

Expected Result	Artifacts Produced during runtime: => Set up job => Checkout code => Set up Flutter => Run cd ./memory_enhancer_app => Analyze Flutter => Post Checkout code => Complete job
Image	

- ***(Amazing) Pipeline Runs Unit Tests successfully***

Item	**Description**
Title	Pipeline Runs Unit Tests Successfully
Description	Tests if the Pipeline Runs Unit Tests successfully.
Input Data	See GitHub Actions Double click most recent workflow Double click Run Unit Tests

Expected Result	Artifacts Produced during runtime: => Set up job => Run actions/checkout@v2 => Set up Flutter => Run cd ./memory_enhancer_app => Run tests => Post Run actions/checkout@v2 => Complete job
Image	

- ADF:

The ADF project takes two files: Dockerfile and startup.sh file. Given that these two files are static but functional it is important to test the resulting combination of these two files. The startup.sh file is particularly touchy to change. Given that ADF is a Linux driven environment, if the startup.sh file is edited on a Windows system, the file may switch to using CRLF line endings. These line endings are not fully compatible with a Linux system and must LF line endings.

Because of this, the ADF system must be validated and verified after each configuration change. Validation is to ensure the ADF system meets the current demand of the courseware. The verification is to ensure ADF can execute the expected tasks of the current courseware. Some axis specifically tested are: Performance of the emulator on supported platforms, Access to data

from remote sources, Platform agnostic capability (known issue: Docker doesn't support nested virtualization on Windows 10), and can ADF build a basic project in the desired language.

- ***ADF builds successfully***

Item	**Description**
Title	ADF builds successfully
Description	Tests if the ADF image builds successfully.
Input Data	Run the following commands: git clone https://github.com/umgc/ADFSummer2021.git cd .\ADFSummer2021\ docker build --pull --rm -f "ADF/dockerfile" -t adfsummer2021:latest "ADF"
Expected Result	Docker image build, and completes with: => exporting to image => => exporting layers => => writing image sha256:XXXXXXXXXXXXXXXXXXX => => naming to docker.io/library/adfsummer2021:latest

- ***ADF can be accessed over RDP***

Item	**Description**
Title	ADF can be accessed over RDP
Description	Tests if the ADF is accesible over RDP.
Input Data	Run the following command, and try to connect to localhost:63389 using RDP.

	docker run -dit -p 63389:3389 --rm --privileged adfsummer2021:latest
Expected Result	RDP client should connect and a login screen should show up.

- ***User can log in***

Item	**Description**
Title	User can log in
Description	Tests if the user can log in using the developer account.
Input Data	RDP is connected to the container, and shows the login screen. User enters: Username: developer Password: password
Expected Result	User should log in and desktop should be shown.

- ***Network connection from within container works***

Item	**Description**
Title	Network connection from within container works.
Description	Tests if there is an active internet connection from within the container, given that there is an active internet connection on the host that runs the container.
Input Data	User open a new terminal window and issues the following commands: sudo apt-get update

	sudo apt-get install iputils-ping ping google.com
Expected Result	Google.com should respond, Ie: developer@3bcdd837d177:~$ ping google.com PING google.com (142.250.184.238) 56(84) bytes of data. {returned ip address = 142.250.184.238 but ttl and response time may vary from tests to test} --- google.com ping statistics --- 2 packets transmitted, 2 received, 0% packet loss, time 1001ms rtt min/avg/max/mdev = 5.308/5.776/6.244/0.468 ms

- ***Visual Studio Code starts***

Item	**Description**
Title	Visual Studio Code starts
Description	Tests if VS Code starts
Input Data	User clicks on VS Code icon on desktop.
Expected Result	VS Code screen appears.

- ***Git is available***

Item	**Description**
Title	Git is available
Description	Tests if Git is available from the command line.
Input Data	User opens a new terminal and issues the following command: git --version
Expected Result	Git version appears, for example: git version 2.25.1

- ***Flutter/Dart is available***

Item	**Description**
Title	Flutter/Dart is available
Description	Tests if Flutter and Dart CLI tools are available from the command line.
Input Data	User opens a new terminal and issues the following command: flutter --version
Expected Result	Flutter and Dart versions appears, for example: Flutter 2.2.1 • channel stable • https://github.com/flutter/flutter.git Framework • revision 02c026b03c (8 weeks ago) • 2021-05-27 12:24:44 -0700 Engine • revision 0fdb562ac8 Tools • Dart 2.13.1

- ***Emulator runs***

Item	**Description**
Title	Android emulator runs
Description	Tests if the Android emulator runs
Input Data	User opens a new terminal and issues the following command: flutter emulators --launch flutter_emulator
Expected Result	Emulator appears and starts Android.

- ***Flutter/Dart app can run in emulator***

Item	**Description**
Title	Flutter/Dart app can run in emulator.
Description	Tests if a Flutter/Dart app can run in emulator.
Input Data	User opens a new terminal and issues the following command while the emulator is running: flutter run
Expected Result	App builds and runs in the emulator

- CaPPMS:

The CaPPMS website is the most interesting to test. This application will use dynamic functional unit tests in order to ensure as much functionality remains operational as possible. The goal would be to have at least 50% test coverage. The tests are expected to be boundary tests and ALAC tests. ALAC test are a natural selection as this is a customer facing website. ALAC tests are to be covered by validation tests by purposefully entering the wrong data which invokes validation to report the errors against fields. It is expected the system for humans will have

problems before the system for human is used by humans. Customers are expected to make mistakes as well and much of the website has been developed with this in mind.

- ***Export Attribute Default is false***

Item	**Description**
Title	Export Attribute Default is false
Description	Test ensures the default value of the ExportAttribute class is set to false when placed on a property of an object.
Input Data	Any test class that uses the ExportAttribute without a parameter on a property within the object.
Expected Result	Attribute is set to false.

- ***Column Header Attribute is true***

Item	**Description**
Title	Column Header Attribute is true
Description	Asserts that decoration of a property with the attribute ColumnHeaderAttribute returns true for the attribute
Input Data	Any test class that decorate a property with the ColumnHeaderAttribute.
Expected Result	ColumnHeader = true

- ***Number Of Columns***

Item	**Description**
Title	Number Of Columns

Description	Test verifies that n number of columns are added to the table based on an enumerable object.
Input Data	An enumerable object that implements ColumnHeaderAttribute on at least two properties of the enumerated object.
Expected Result	N = table.HeaderRow.Count

- ***Column Names***

Item	**Description**
Title	Column Names
Description	The names of expected n columns equals the names of n columns in the test table.
Input Data	An enumerable object that implements ColumnHeaderAttribute on at least two properties of the enumerated object.
Expected Result	Column[n].Name = table.HeaderRow[n].Value.ToString()

- ***Verify Single Row***

Item	**Description**
Title	Verify Single Row
Description	At least the first row of the table matches an enumerable object with one item added to the collection.
Input Data	An enumerable object with at least one item in the collection with different values assigned to at least two properties.
Expected Result	N property values = table.rows[n][c].Value

- ***Delete File from file system***

Item	Description
Title	Delete File from file system
Method	LocalProjectFileManager.DeleteAsync(string location)
Description	The system can delete a generated file.
Input Data	The system creates a file to be deleted.
Expected Result	The system receives an empty string after calling LocalProjectFileManager.DeleteAsync()

- ***Save file to file system***

Item	Description
Title	Save file to file system
Method	LocalProjectFileManager.SaveAsync(stream, fileId, filename)
Description	Save Test generated file to file system
Input Data	System generated file can be saved with LocalFileProjectFileManager
Expected Result	File.Exists(location) = true

- ***Read stored file contents***

Item	Description
Title	Read stored file contents
Method	LocalProjectFileManager.ReadAsync(string location)
Description	Reads the contents of a test generated file.
Input Data	Test generated file

Expected Result	File contents returned by ReadAsync matches generated data saved before read.

- ***Add Idea***

Item	**Description**
Title	Add Idea
Method	ProjectManagerService.AddAsync(ProjectInformation idea)
Description	Add a new instance of a ProjectInformation object
Input Data	A test generated instance of a ProjectInformation object
Expected Result	Return response from AddAsync is true and one title exists when checked using method ProjectManagerService.GetIdeaTitles()

- ***Remove idea***

Item	**Description**
Title	Remove idea
Method	ProjectManagerServiceRemoveAsync(ProjectInformation idea, IPrincipal principal)
Description	Test generates an idea, adds the idea to the system, verifies that the test has been added, removes the idea, and verifies that the test has been removed.
Input Data	Test generated idea
Expected Result	Verifies that added n items before verifying m items are removed

- ***Verify ProjectManagerService saved backing file***

Item	**Description**
Title	Verify ProjectManagerService saved backing file
Description	The project manager saves the file upon receiving a change event. A test needs to be able to add an idea the verify that the change event fired and saved the file by manually loading the expected save file.
Input Data	N number of items added to the ProjectManagerService
Expected Result	N number of items retrieved manually

- ***Update Idea***

Item	**Description**
Title	DidUpdateProjectManager
Description	After an idea is submitted to the manager can it be updated.
Input Data	An idea that has been added to the ProjectManagerService has an edit to the projoct title that includes a unique identifier.
Expected Result	N = $"New Test Project Title-{Guid.NewGuid()}";

- ***Export to PDF***

Item	**Description**
Title	DoesExportHashMatchExpected
Description	An exported pdf containing the a baseline project should match pre-computed hash.

Input Data	Base line Idea created from the CreateIdea method that can be used in all idea tests for creation.
Expected Result	Hash using SHA512 against known good Export

- ***Add Comment***

Item	**Description**
Title	DoesAddComment
Description	Adds a comment and checks to see if comment has been added
Input Data	Adds $“Test comment-{Guid.NewGuid()}”;
Expected Result	N = $“Test comment-{<guid created>}”;

- ***Contact Initialization***

Item	**Description**
Title	Contact Initialization
Description	Zero Properties are null and ContactID is not empty
Input Data	None
Expected Result	All Properties initialize to not null values and Guid is not empty

- ***Project List Does Not Load without Authentication***

Item	**Description**
Name	ProjectListAuthenticationVerification
Description	The project list page does not display the project table
Requirements	The application is running in a hosted environment.

Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Navigate to /projectlist
Expected Output	You must be authorized to access this page.
Assumptions	Test harness is not authenticated with AAD.

- ***List of users does not load***

Item	**Description**
Name	ProjectListAuthenticationVerification
Description	The project list page does not display the project table
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Navigate to /users
Expected Output	You must be authorized to access this page.
Assumptions	Test harness is not authenticated with AAD.

- ***A FAQ can be added***

Item	**Description**
Name	Add Faq
Description	A FAQ can be added to the system
Requirements	The system saves the fact and can be checked indepenantly of the FaqService. Can be verified via FaqService.

Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Faq Page initialized and a faq is added.
Expected Output	The added faq can be verified via the faq service and independently verified in the backing file
Assumptions	

- ***A Faq Reply can be added***

Item	**Description**
Name	Add Reply to faq question
Description	A question can be replied to
Requirements	The system has a faq without a question. The question gets replied to. The question gets updated in backing service.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Faq Page initialized and a faq is added. The previously added faq can be answered by authorized account holders
Expected Output	The updated faq can be verified via the faq service and independently verified in the backing file
Assumptions	Uses authenticated user with user role

- ***The project idea page loads without authentication***

Item	**Description**
Name	Load Index page
Description	When the Index.razor component page is loaded the markup has load the submit idea form.
Requirements	The idea form contains at least the labels for first name, last name, email, phone, project title, project description, attachments, and website.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Navigate to /
Expected Output	The page contains the specified requirements as elements return by the system
Assumptions	No authentication is required

- ***FailedBlankSubmission***

Item	**Description**
Name	FailedBlankSubmission
Description	Without inputting any data into the SubmitIdea component, a validation error is received upon clicking the submit button
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.

Steps	Leaving all inputs blank, click submit
Expected Output	Validation error displays with the following: • The ProjectTitle field is required. • The ProjectDescription field is required. • The FirstName field is required. • The LastName field is required. • The Email field is not a valid e-mail address. • The Phone field is not a valid phone number. • The SponsorFirstName field is required. • The SponsorLastName field is required. • The SponsorEmail field is not a valid e-mail address. • The SponsorPhone field is not a valid phone number.
Assumptions	

- ***FileSizeUploadValidation***

Item	**Description**
Name	FileSizeUploadValidation
Description	The file size will be validated upon upload.
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Filling out First Name, Last Name, Email, Phone, Project Title, Project Description, and ensuring the Self Sponsored checkbox is check, attempt to upload a file larger than the maximum sized.
Expected Output	Validation error displays with the following:

	• Max file size (10) exceeded on: testfile
Assumptions	The maximum size is set in the appsetting.json file.

- ***FileCountUploadValidation***

Item	**Description**
Name	FileCountUploadValidation
Description	The number of files will be validated upon upload.
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Filling out First Name, Last Name, Email, Phone, Project Title, Project Description, and ensuring the Self Sponsored checkbox is check, attempt to upload a more files than allowed by the system.
Expected Output	Validation error displays with the following: • Exceeded max number of files. Max:10.
Assumptions	The maximum number of files is set in the appsetting.json file.

- ***Email Field Does not Validate***

Item	**Description**
Name	EmailFieldDoesNotValidate

Description	Users may not enter a qualified email address. This should be validated before input is accepted.
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Enter someemailaddress in the Email address input field
Expected Output	Validation error displays with the following: • The Email field is not a valid e-mail address.
Assumptions	

- ***PhoneFieldDoesNotValidate***

Item	**Description**
Name	PhoneFieldDoesNotValidate
Description	Users may enter text for a phone number. We expect all numbers
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Enter “Some Text” into the Phone input field.
Expected Output	Validation error displays with the following: • The Phone field is not a valid phone number.
Assumptions	

- ***FristNameFieldDoesValidate49Characters***

Item	**Description**
Name	FristNameFieldDoesValidate49Characters
Description	User is limited to 50 characters for their first name
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Enter 49 character into the First Name Field
Expected Output	There should be no First Name validation errors
Assumptions	Any other validation errors are ignored.

- ***FristNameFieldDoesNotValidate51Characters***

Item	**Description**
Name	FristNameFieldDoesNotValidate51Characters
Description	User is limited to 50 characters for their first name
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Enter 51 character into the First Name Field
Expected Output	There should be a First Name validation error: • First Name is too long.
Assumptions	Any other validation errors are ignored.

- ***LastNameFieldDoesValidate49Characters***

Item	**Description**
Name	LastNameFieldDoesValidate49Characters
Description	User is limited to 50 characters for their first name
Requirements	The application is running in a hosted environment.
Prerequisites	The system has initialized bUnit for tests and has created a mock project manager service.
Steps	Enter 49 character into the Last Name Field
Expected Output	There should be no First Name validation errors
Assumptions	Any other validation errors are ignored.

- ***LastNameFieldDoesNotValidate51Characters***

Item	**Description**
Name	LastNameFieldDoesNotValidate51Characters
Description	User is limited to 50 characters for their first name
Requirements	The system has initialized bUnit for tests and has created a mock project manager service.
Prerequisites	The test system is capable of automated selenium tests.
Steps	Enter 51 character into the Last Name Field
Expected Output	There should be a First Name validation error: • Last Name is too long.
Assumptions	Any other validation errors are ignored.

5.2 Test Data

- To be completed during the Testing Phase (Milestone 4)

5.3 Third-Party Requirements

- Mentor feedback during the Listening Phase (Milestone 4)
- Sponsor feedback during the Listening Phase (Milestone 4)

6 Issues

- To be identified during the Testing Phase

Appendix

Definitions and Acronyms

ADF– Advance Development Factory

AKS – Azure Kubernetes Service which offers serverless Kubernetes and integrated CI/CD

CI/CD – Continuous Integration and Continuous Deployment

cluster – a group of computer servers working together towards a common goal

Docker container – when the image is started, and the container environment is created

Docker image – the actual application package, an artifact that can be moved around, such as in a Docker repo

DSO – DevSecOps

kubectl – CLI tool for Kubernetes clusters

namespaces – clusters inside a cluster, groups Kubernetes resources together for organizational purposes

node – the smallest package of Kubernetes to manage, usually has one container inside

pod – the smallest deployable unit in Kubernetes, a layer of abstraction over containers

TCO – Total Cost of Ownership

VM – Virtual Machine

YAML – data serialization language used for configuration files

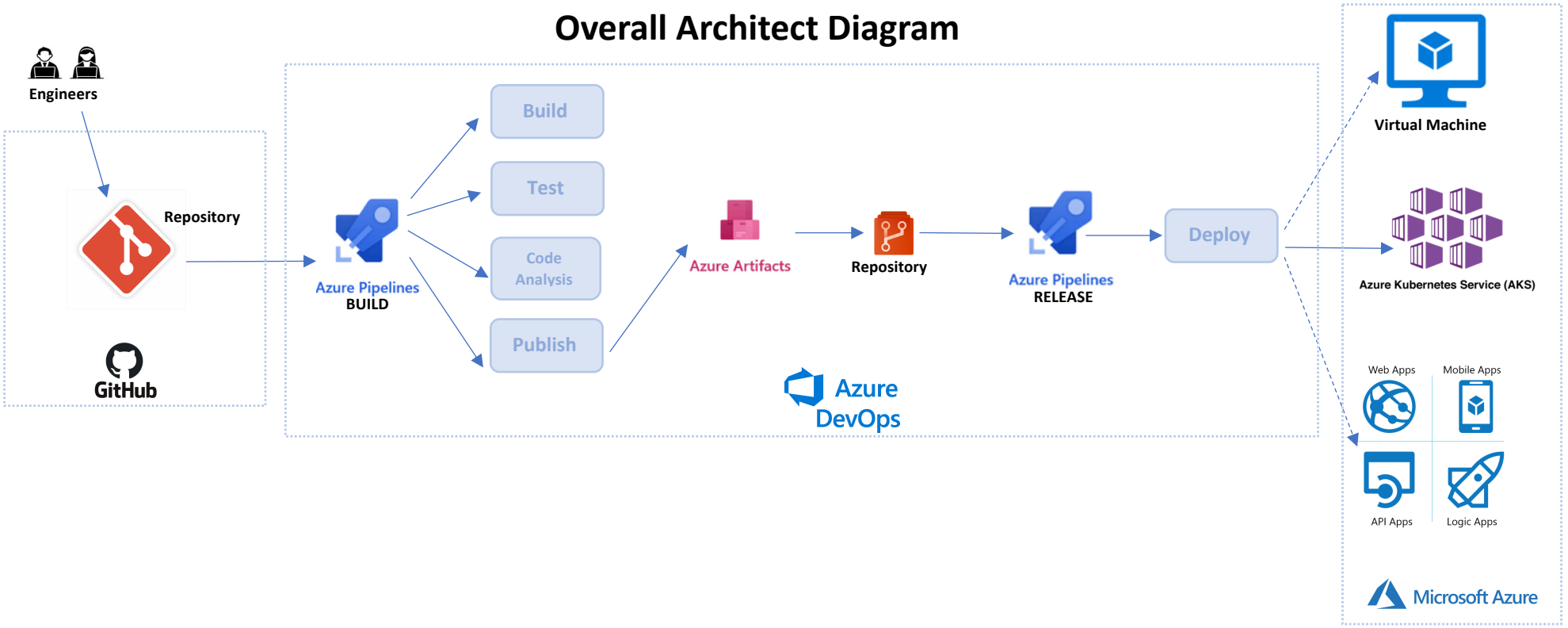
SYSTEM ARCHITECTURE

Overall Architectural Design

The Form Scriber DevSecOps project requires the use of free GitHub and Azure services. Therefore, the solution is designed to be deployed on these platforms. GitHub will provide the code repository for collaboration via distributed version control. The Form Scriber project will utilize two main services from Azure. **Azure DevOps** provides a set of tools to improve collaboration and productivity for development and IT operations. **Microsoft Azure** provides cloud services related to infrastructure such as virtual computing, storage, networking, containers, and other services. Please see Figure 1 for the overall architecture.

Figure 1

Overall Architect Diagram



Code Repository

All development teams (DialogFlow and Mobile) are required to utilize GitHub repository. The DSO team is responsible for the creation of the project and appropriate repositories. Each team will have access to the appropriate repository with a set of policies created by the DSO team. Please see the structure of the repository below.

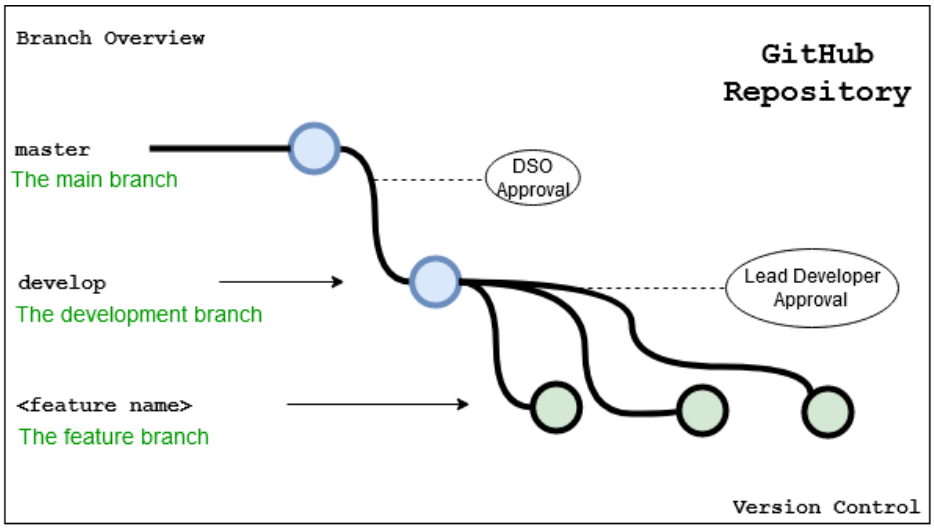


Figure 2. Branch Overview
Note. Read from bottom to top

The master branch is protected, which requires the approval of the DSO team for the merge from the development branch in order to check for code quality and security. Direct merge to master branch from the feature branch is not prohibited.
The development branch is restricted to the lead developer or one required review from the team member.
The feature branch allows developers to create a new branch for each feature being worked on.
Azure DevOps

GitHub repository for each development team will be integrated with Azure DevOps pipeline. Azure DevOps pipeline will monitor changes in the repository and trigger processes as configured. There are two types of pipeline in Azure DevOps. The **Build** pipeline provides steps for the build (compile), test, code analysis, package/publish artifacts, the process known as CI (Continuous Integration). The **Release** pipeline provides the means to deploy artifacts to a specific environment such as a Virtual Machine (VM), Docker image, or Kubernetes in Microsoft Azure or other supported cloud providers. The Release pipeline is known as CD (Continuous Deployment), which is a part of the CI/CD process.

Build Pipeline

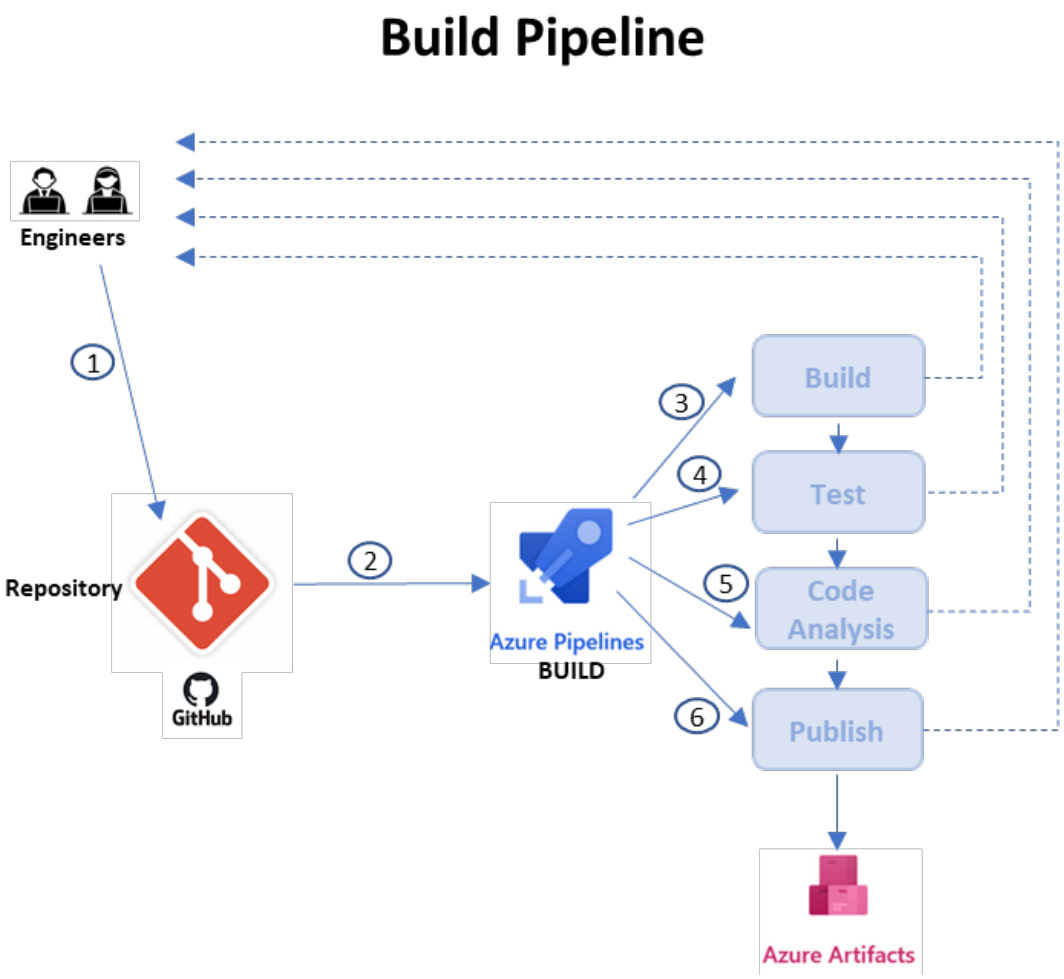


Figure 3. Build Pipeline

For the development branch, Azure pipeline will be configured only to run Build and Test to ensure the code is compiled and passes unit tests. The developer will get notification from the pipeline for the status of each merge to the development branch. For the master branch, the merge from development is controlled by the DSO team as previously mentioned in section 2.2. Therefore, DSO team will do a manual code review and compliance checks before approving the merge to the master branch. Azure DevOps will run the full pipeline which includes Build, Test, Code Analysis and Publish. DSO team will be notified with the status for every merge to the master branch. The gated check-in for Azure pipeline will be considered for the master branch.

The Form Scriber application utilizes 3 (3) main programming languages, which includes JavaScript (Vue), Dart (Flutter) and Go. Therefore, one pipeline (Dialogflow) will cover Go and JS and the other (Flutter/Dart) covers Go. Below are the high-level steps of the Build pipeline.

Merge (Pull) is requested to a branch of the GitHub repository.
Azure pipeline automatically detects changes and triggers the Build task.
The Build task is configured to build (compile) code and can automatically trigger the next task in the pipeline if the build is successful.

The Test task is configured to run unit tests and can automatically trigger the subsequent tests if the unit test is passed.
The Code Analysis task is configured to check for code quality and security. The plan is to review the report and remediate according to policies agreed among the teams.
The Publish task is configured to package and store the artifacts such as executables, zip, or application package to a binary repository which can be Azure Artifacts or a staged location used by the Build pipeline.

Release Pipeline

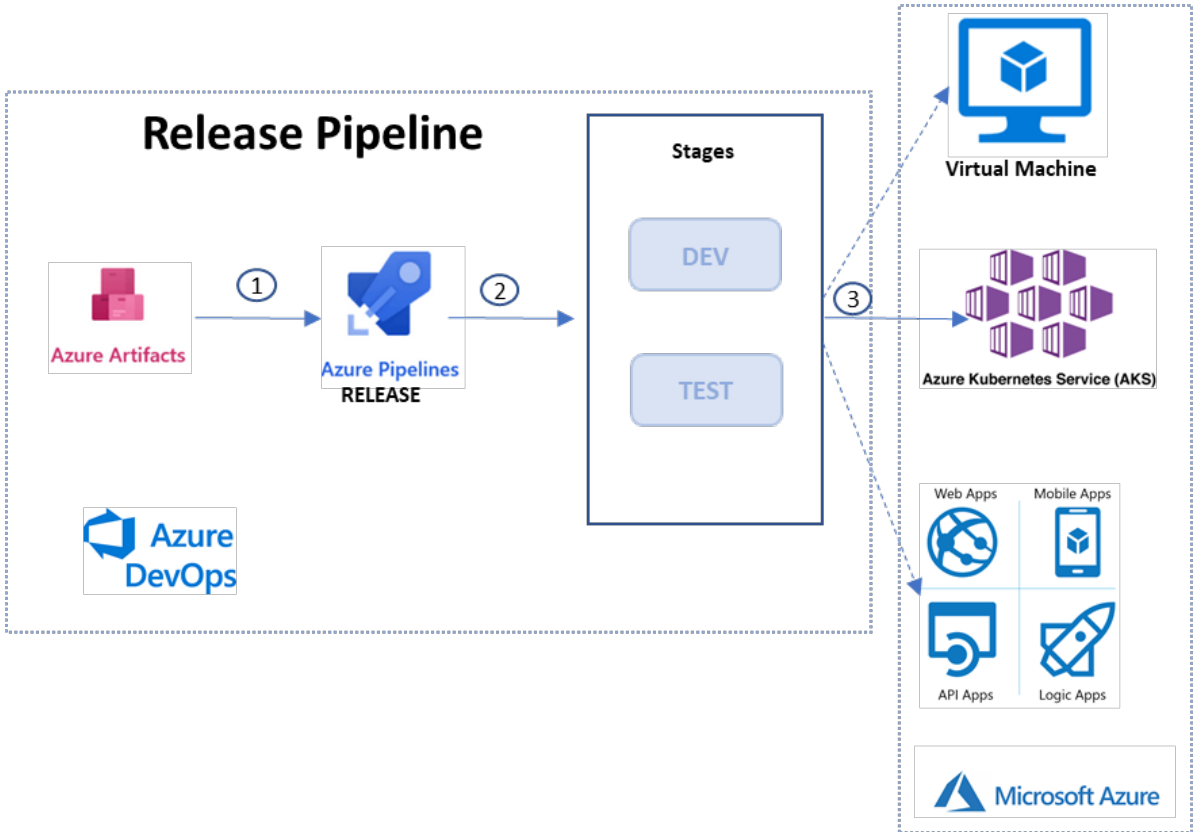


Figure 4. Release Pipeline

The release pipeline can be configured to monitor an artifact repository for changes in versions to deploy the application to a specified environment such as development or test. In Figure 4, the release pipeline monitors Azure Artifacts, which can be the staged or temporary repository for the build pipeline to publish application binaries or packages.
The release pipeline detects new version of the artifact that is published by the Azure Build pipeline and triggers the deployment.
The release is configured with stages such as development and test.
The release pipeline deploys the artifact to a specific environment.
Microsoft Azure

As described previously, **Microsoft Azure** provides cloud services relating to infrastructure such as virtual computing, storage, networking, containers, and other services. The pipeline shall

deploy a Kubernetes cluster in Azure utilizing AKS (Azure Kubernetes Service). The fallback plan is to leverage traditional Azure virtual machines and app services where it is not feasible or possible to deploy Form Scriber components on AKS.

Azure Kubernetes Service (AKS)

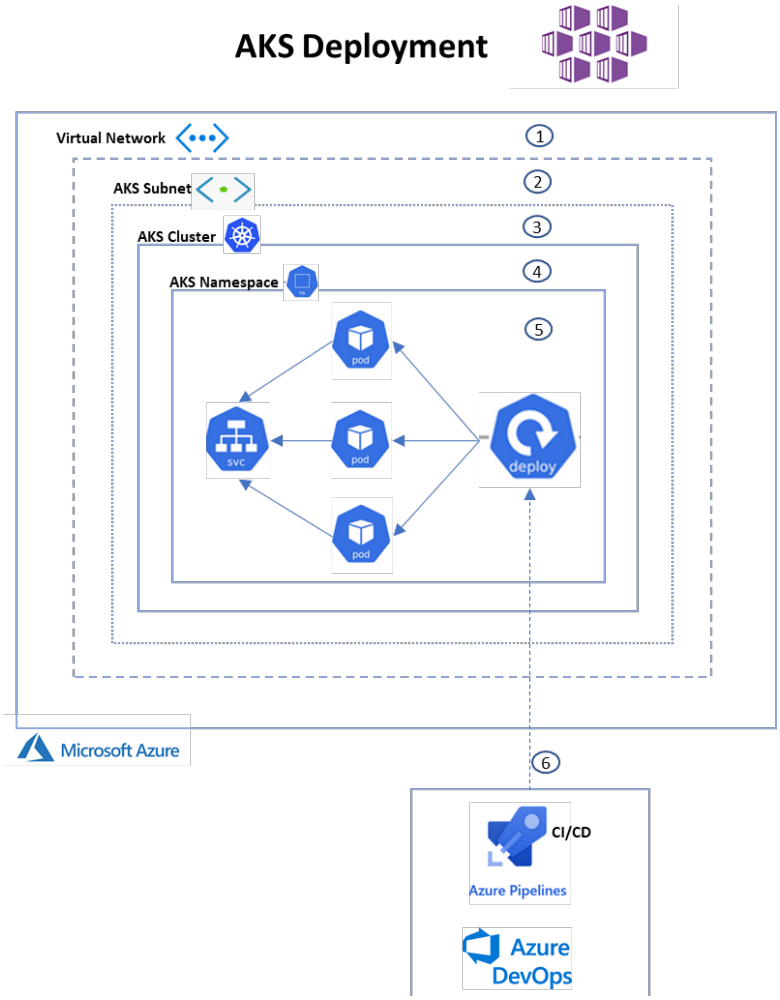


Figure 5. AKS Deployment

Azure provides several methods to deploy an AKS cluster. Most components can be deployed at one time. However, each component can be deployed separately for more customization. Figure 5 shows each AKS component.

Virtual Network can be created separately to provide an internal virtual network within an organization.
AKS Subnet provides the separation and boundary between networks within an organization.
AKS Cluster provides a group of all related components and services to manage, deploy and scale containers.
AKS Namespace groups logical resources such as pods and deployments, which can be designated to separate each project with access restrictions.

AKS resources include services, deployments, and pods (containers) for applications, which can be scaled rapidly.
CI/CD pipeline automates the deployment of container applications to AKS cluster.
Azure Virtual Machine and App Services

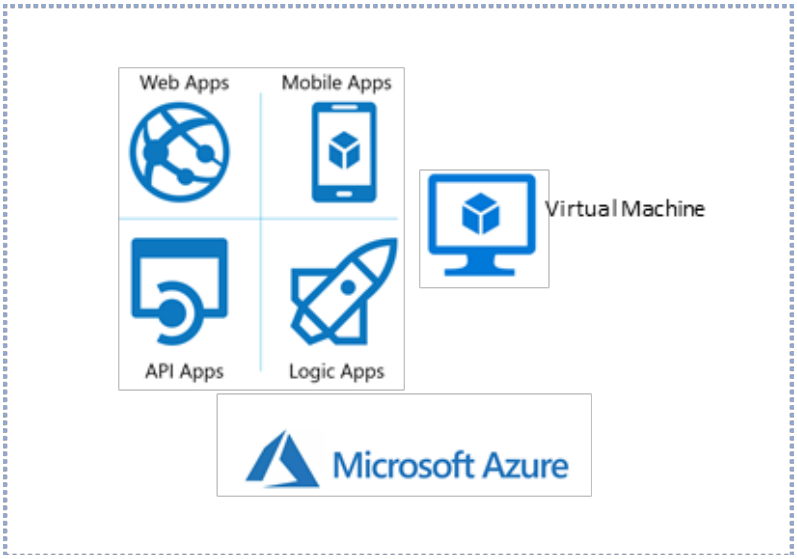


Figure 6. Microsoft Azure Services

The traditional virtual machines and app services offered by Microsoft Azure will be considered where it is not feasible to deploy Form Scriber components on AKS. Additional information will be provided as needed.

Advanced Deployment Factory (ADF)

To satisfy a requirement of using ADF framework that was created from the previous semester, an ADF solution for Go will be written with the Make file template. ADF will be leveraged to provide developers tools and the method to deploy to AKS.
COMPONENT DESIGN

Technologies

GitHub – Distributed Code Repository
Microsoft Azure – Infrastructure Cloud Services
Azure DevOps – CI/CD tools
Docker.OI
Build Pipelines

Flutter and JavaScript

For the Dart (Flutter) language which will be utilized by the development team, an Azure pipeline will be created using the YAML method. JavaScript has small footprint in the Mobile

application. Therefore, the Go pipeline will be installed NPM (Native Package Manager) to do some checks on JavaScript codes. Please see the example below:

Go

For the Go language which will be utilized by the development team, an Azure pipeline will be created using the YAML method. Please see the example below:

azure-pipelines.yml

C: > temp > azure-pipelines.yml

```yaml
#Trigger from GitHub repository branch
    Trigger:
      - master

#Build Environment
    pool:
      vmImage: 'ubuntu-latest'

#Build Code
    steps:
      - task: GoTool@0
        inputs:
          version: '1.13.5'
      - task: Go@0
        inputs:
          command: 'get'
          arguments: '-d'
          workingDirectory: '$(System.DefaultWorkingDirectory)'
      - task: Go@0
        inputs:
          command: 'build'
          workingDirectory: '$(System.DefaultWorkingDirectory)'

#Test
      - task: Go@0
          inputs:
            command: 'test'
            arguments: '-v'
            workingDirectory: '$(modulePath)'

#Publish code coverage results
    - task: PublishCodeCoverageResults@1
    inputs:
        codeCoverageTool: Cobertura # or JaCoCo
        summaryFileLocation: '$(System.DefaultWorkingDirectory)/**/*coverage.xml'
        reportDirectory: '$(System.DefaultWorkingDirectory)/**/coverage'

#Publish
    - task: CopyFiles@2
      inputs:
        TargetFolder: '$(Build.ArtifactStagingDirectory)'
      - task: PublishBuildArtifacts@1
        inputs:
          artifactName: drop
```

Release Pipelines

An AKS environment will be added to the Azure DevOps pipeline to deploy Docker image with Form Scriber components. Please see the example in Figure 7.

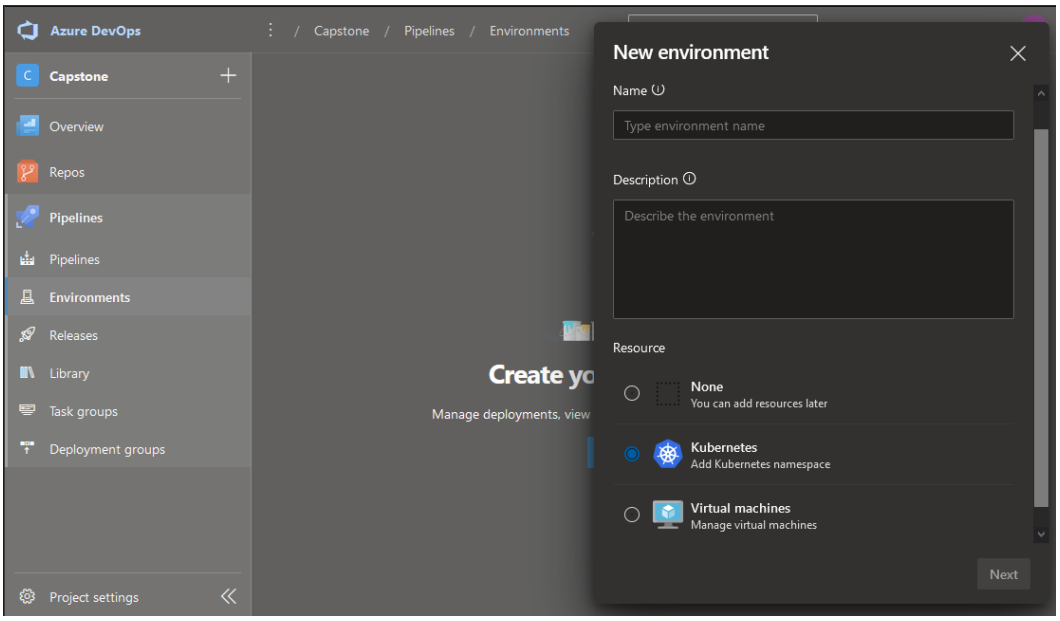


Figure 7. Azure DevOps and AKS

A release pipeline will be created to contain one stage called “Test Environment” to deploy Docker image with Form Scriber components. The release pipeline will also deploy the Docker image to the AKS cluster on Azure. Please see the example in Figure 8.

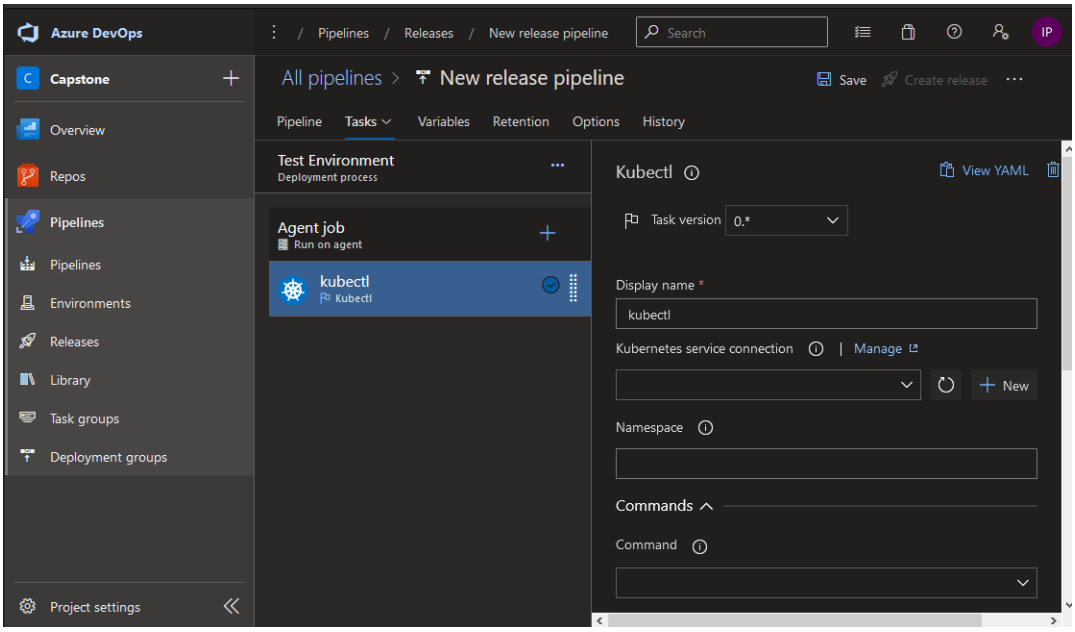


Figure 8. Release Pipeline Stage

Azure Kubernetes Service

An AKS cluster will be created on Microsoft Azure. The cluster will be named as “Form ScriberAKS”. Creating the AKS cluster can be completed and managed using command line (kubectl) or the wizard provided by Azure. For customization, each layer can be deployed

separately. For this project, AKS will be deployed with recommended defaults. Please see the example in Figure 9. The AKS will be added to the student free account with $200 credit for the first month.

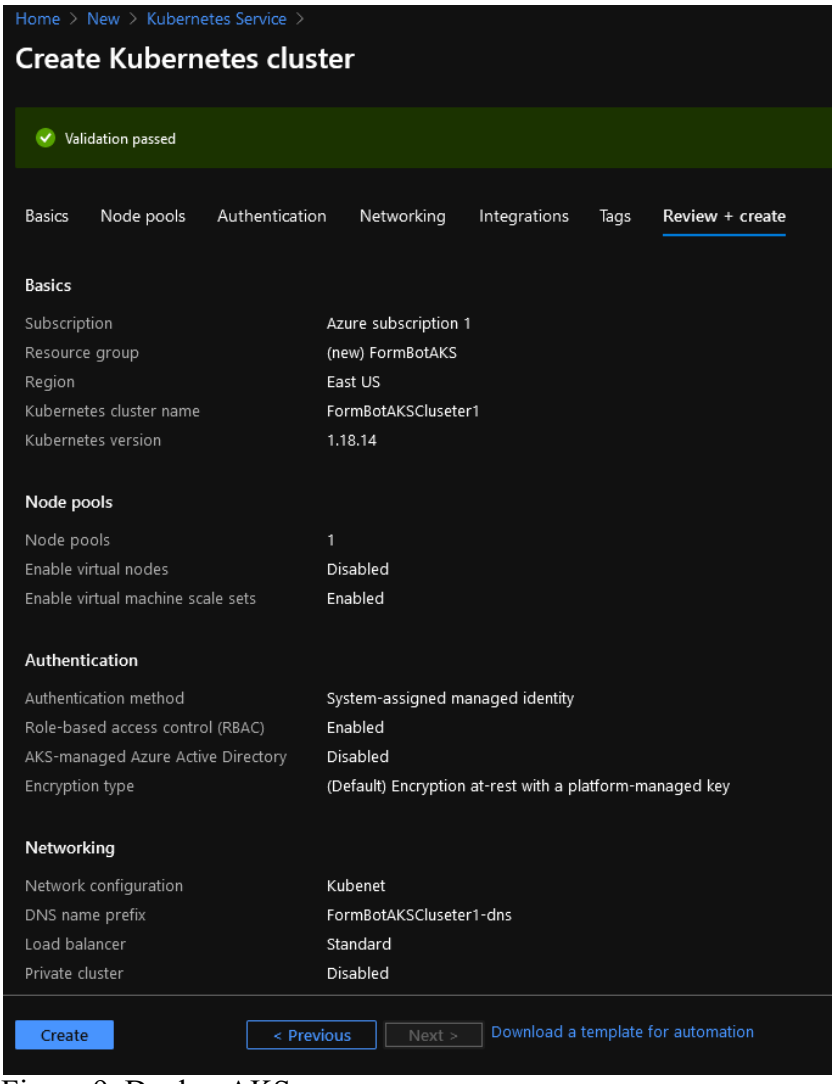


Figure 9. Deploy AKS

Advance Development Factory (ADF)

DSO will create the ADF for Go to cover all development tools. The docker image will include all necessary GO tools, Azure CLI, Helm and KuberCTL for developers to use. Developers will only need to install GIT and Docker to able to able to leverage GO ADF.