**Project Plan**

|  |  |
| --- | --- |
| UMGC Fall 2020 | Benjamin Fetterman, Benjamin Murray, Hanim Danur, James Cornelius, Robert Lee  SWEN 670 |

Project Plan Approvals

|  |  |  |
| --- | --- | --- |
| Name | Signature | Date |
| Approved by:  Dr. Mir Assadullah |  |  |
| Approved by:  "Stakeholder" |  |  |
| Approved by:  "Project Manager" | Robert Lee | 9/29/2020 |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Name | Description of Change(s) |
| 1.0 | 9/3/2020 | Robert Lee | Initial Release |
| 1.1 | 9/29/20 | Robert Lee | Corrections and Test Plan Integration |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.0 Introduction 6](#_Toc55319792)

[2.0 Project Description 6](#_Toc55319793)

[2.1 Customer/User 6](#_Toc55319794)

[2.2 Contracting 6](#_Toc55319795)

[2.3 Referenced Documents 6](#_Toc55319796)

[2.4 Re-approval Criteria 7](#_Toc55319797)

[2.5 Change Control 7](#_Toc55319798)

[3.0 Project Assignment 7](#_Toc55319799)

[3.1 Project Scope 7](#_Toc55319800)

[3.1.1 Project Assumptions 8](#_Toc55319801)

[3.1.2 Project Constraints 8](#_Toc55319802)

[3.2 Project Organization/Staffing 8](#_Toc55319803)

[3.2.1 Roles and Responsibilities 9](#_Toc55319804)

[3.3 Responsibility Assigned Matrix (RACI) 10](#_Toc55319805)

[3.4 Project Tools 10](#_Toc55319806)

[3.5 Project Facilities 11](#_Toc55319807)

[3.6 Project Documents Storage 11](#_Toc55319808)

[3.7 Distribution Statement and Security Classification Markings for Work Products 11](#_Toc55319809)

[3.8 Knowledge/Skills Needed 11](#_Toc55319810)

[3.9 Project Schedule 12](#_Toc55319811)

[3.9.1 Work Breakdown Structure 12](#_Toc55319812)

[3.9.2 High-Level Schedule 14](#_Toc55319813)

[3.9.1 Project Integrated Master Schedule (IMS) 14](#_Toc55319814)

[3.10 Estimates for Size, Effort, and Cost 14](#_Toc55319815)

[3.11 Project Budget 15](#_Toc55319816)

[3.12 Deliverables 15](#_Toc55319817)

[4.0 Description 16](#_Toc55319818)

[4.1 Communication 16](#_Toc55319819)

[4.2 Reviews and Meetings 16](#_Toc55319820)

[5.0 Risk Analysis 17](#_Toc55319821)

[5.1 Risk management Process 17](#_Toc55319822)

[5.2 Current program risks 19](#_Toc55319823)

[6.0 Organization of the project 20](#_Toc55319824)

[6.1 Overview 20](#_Toc55319825)

[6.2 Scrum Positions 21](#_Toc55319826)

[6.3 Scrum Components 21](#_Toc55319827)

[6.4 Scrum Artifacts 21](#_Toc55319828)

[6.5 Version Control and Build Process 21](#_Toc55319829)

[7.0 Test Plan Identifier 22](#_Toc55319830)

[7.1 Introduction 22](#_Toc55319831)

[7.1.1 Background 23](#_Toc55319832)

[7.1.2 Objectives of Test Plan 23](#_Toc55319833)

[7.1.3 Objectives of UAT 23](#_Toc55319834)

[7.2 Scope 23](#_Toc55319835)

[7.2.1 Test Items 23](#_Toc55319836)

[7.2.2 Features to Be Tested 23](#_Toc55319837)

[7.2.3 Features Not to Be Tested 24](#_Toc55319838)

[7.3 Test Tools and Resources 24](#_Toc55319839)

[7.3.1 Testing Tools 24](#_Toc55319840)

[7.3.2 Operating Systems and Browsers 24](#_Toc55319841)

[7.3.3 Hardware 24](#_Toc55319842)

[7.4 Testing Approach 24](#_Toc55319843)

[7.4.1 Develop Tests 24](#_Toc55319844)

[7.4.2 Activities of test development process includes: 25](#_Toc55319845)

[7.4.3 QA Reviews 25](#_Toc55319846)

[7.4.4 Test Preparation 25](#_Toc55319847)

[7.4.5 Run Test 25](#_Toc55319848)

[7.4.6 Review Test Results 25](#_Toc55319849)

[7.5 Testing Risk Issues 26](#_Toc55319850)

[7.5.1 Schedule Risk due to: 26](#_Toc55319851)

[7.5.2 Technical Risk 26](#_Toc55319852)

[7.5.3 Operational Risk 26](#_Toc55319853)

[7.5.4 Risk Matrix 26](#_Toc55319854)

[7.6 Item Pass/Fail Criteria 27](#_Toc55319855)

[7.6.1 Severity List 27](#_Toc55319856)

[7.7 Entry Criteria 28](#_Toc55319857)

[7.8 Suspension Criteria and Resumption Requirements 28](#_Toc55319858)

[7.9 Test Deliverables 28](#_Toc55319859)

[7.9.1 Reports and Content 28](#_Toc55319860)

[7.9.2 Format 29](#_Toc55319861)

[7.10 Test Tasks 29](#_Toc55319862)

[7.11 Responsibilities 29](#_Toc55319863)

[7.12 Summary Test Schedule 30](#_Toc55319864)

[7.13 Test Case and Matrix Preparation 30](#_Toc55319865)

[7.14 Test Case Execution and Reporting 30](#_Toc55319866)

[Appendix A: Acronyms and Abbreviations 32](#_Toc55319867)

[Appendix B: IMS/WBS 33](#_Toc55319868)

[APPENDIX C: References 41](#_Toc55319869)

List of Tables

[Table 1 Referenced Documents 6](#_Toc55319870)

[Table 2 Roles and Responsibilities 9](#_Toc55319871)

[Table 3 Project Tools 11](#_Toc55319872)

[Table 4 Estimate of Total Hours 14](#_Toc55319873)

[Table 5 Estimate of Total Cost 15](#_Toc55319874)

[Table 6 Deliverables 15](#_Toc55319875)

[Table 7 Reviews and Meetings 16](#_Toc55319876)

[Table 8 Risk Descriptions 19](#_Toc55319877)

[Table 9 Functional Test Summary 23](#_Toc55319878)

[Table 10 Risk Matrix 26](#_Toc55319879)

[Table 11 Severity List 27](#_Toc55319880)

[Table 12 Testing Responsibilities 29](#_Toc55319881)

[Table 13 Testing Assignments 29](#_Toc55319882)

[Table 14 Schedule for Test Tasks 30](#_Toc55319883)

List of Figures

[Figure 1 Scrum Project Structure 9](#_Toc55319884)

[Figure 2 Project Role/Staff 10](#_Toc55319885)

[Figure 3 Project Schedule 13](#_Toc55319886)

[Figure 4 Risk Management Process 18](#_Toc55319887)

[Figure 5 Risk Matrix 19](#_Toc55319888)

[Figure 6 Current Project Matrix 20](#_Toc55319889)

[Figure 7 Use Case Diagram of System 22](#_Toc55319890)

1. Introduction

This project plan is for the documentation of the City of Pasadena’s Zoning and Planning City Chatbot, and what is required to execute and control the software effectively. Within is contained the communication with stakeholders and the formation of the deliverables for this project. Additionally, details about the assumptions, constraints, decisions, facilities, organization, planning, scope, schedule, storage, and tools are documented within. The City Chatbot shall here onto be described as the system, application, or agent.

1. Project Description

This project’s goal is to begin developing a permit zoning assistance application for the city of Pasadena’s zoning and planning website. The application provides site visitors the ability to locate their permits and regulations that are relevant to their specific address by interacting with a chatbot, avoiding calling a customer service representative or searching the website. The application facilitates the collection of the residents’ addresses, the comparison of those addresses to a city provided map containing permitting zones, and then providing the applicable permitting information they are looking for based off of an established permit and regulations database. The application shall provide permit and zoning information and assistance through chatbot functionality on through the city of municipality website. The application assists users by directly providing permit and zoning information based on user input. The application displays links and referential information. The application is not intended to provide all details and information regarding permits and zoning.

Customer/User

The primary stakeholder for this project is UMGC. The end users shall be visitors to the local city website. Web site visitors shall access the system through a chatbot. User interactions are expected to be one off website visits looking for specific information. The term user will refer to any individual that is accessing the website and chatbot feature with the intention of finding permit or regulation information.

Contracting

The system is developed as a Capstone project finalizing University of Maryland Global Campus (UMGC) Master’s degree. This capstone project and document represents an incremental step towards completion of the system. There are two previous versions of the system. Each version was developed by a project team attending a previous SWEN 670 class. Each previous version has shown to not meet the expectations of the stakeholders in design and functionality. The application will require multiple iterations of development, where each UMGC SWEN 670 class will build upon the previous class’s iteration. This iteration will isolate a single feature of the project, the chatbot, and implement it in a way that allows for other teams to take it in its entirety and make it part of a larger system that the individual cities can maintain. In the process of isolating this functionality the project will also focus on creating a common development approach to the remaining sections and provide a base for them to start with.

Referenced Documents

Table 1 Referenced Documents

|  |  |
| --- | --- |
| Title | Reference Location |
| The Scrum Guide | <https://www.scrum.org/resources/scrum-guide> |
| CSPP-002 | **Software Development Coding Standards** |
| ISO- 31000:2018(E) | **Risk Management** |
| Course Material | **UMGC SWEN 670 Course Material** |

Re-approval Criteria

Reapproval criteria describes the change control process to functional and non-functional requirements. The primary stakeholder is the sole individual able to change functional and non-functional requirements. This could occur at any moment, without notice.

Change Control

The change control process describes the procedural steps and authorizations to formalize changes to project management and system requirement derivatives. The primary stakeholder, UMGC, is the final authorizer for all changes. Any changes to project or requirement process shall be documented in their respective document.

Changes to project and requirements shall be communicated to the primary stakeholder. The primary stakeholder shall then approve or disapprove of the proposed change. Although larger system requires a structured and detailed change control definition. The size and scope of this system does not warrant that level of detail. Simple email, verbal, and group communication through common applications is sufficient for this process.

To ensure control, project changes shall be documented in the Project Plan. Requirement changes shall be documented in the Software Requirements Specification. System architecture and physical architecture changes shall be documented in the Technical Design Document. The Programmers Guide, Test Plan and Deployment Installation is out of scope of the project change control process.

1. Project Assignment

The following section outlines how the project and all project resources will be managed throughout the life of the project to include all milestones, all roles and responsibilities, and all software and locations used.

Project Scope

This project will enter the initial documentation phase on the date specified in the Work Breakdown section below and will reach completion upon the submission of the fourth and final milestone. After which time, the development team will transition the project to the project stakeholders over a weeklong transition period. This project will be conducted using Scrum project management methodologies and will subscribe to the Scrum sprint development framework. In addition to the documentation milestones and deliverables identified in this section, the team will work towards obtaining the following functionality objectives:

* Establish an automated Chatbot that will assist users with identifying the correct permit and regulation codes that are associated to specific addresses.
* Establish a backend handler to translate address inputs to zones and identify the applicable permit and regulation codes.
* Create dynamic links throughout the application that allow for changes in codes and permit applications without the city management employees from having to adjust code or database references.
* Deliver a finished project that is modular and capable of being deployed with little modification to multiple municipalities throughout the country.
  + 1. Project Assumptions

The following project assumptions will be made before the start of work on the project:

* The project sponsor and stakeholders will be responsible for providing all human and material resources in order to complete the project.
* All staff and resources related to the cost of this project will maintain the same expense rate throughout the life of the project unless otherwise approved by all stakeholders.
* The scope and timeline of this project will not be adjusted throughout the life of the project unless otherwise approved by all stakeholders.
* All documentation development for the project and code generated throughout the project life cycle will be considered open source and useable by future teams.
* All deadlines submitted in this project plan and the referenced course syllabus are fixed dates for submission.
* The information called by the chatbot is not directly from the city of Pasadena’s database.
* The information called by the chatbot resides in a custom database used to show functionality.
* The system is not configured to execute on the city of Pasadena’s IT infrastructure.
  + 1. Project Constraints

The following constraints are placed on the project and will be conformed to throughout the project lifecycle:

* All milestone submissions for this project will be submitted by official project channels to include UMGC online classroom and emails to the established stakeholder.
* All milestones will have stakeholder approval prior to the project team advancing to the next development stage and working on follow on milestones.
* All software requirements will be identified and approved prior to the software development team continuing the project.

Project Organization/Staffing

The project organization is structured in a way that will follow typical Scrum project architecture to allow for easier communication between the development team and the project sponsor. Due to the small team size and short project timeline, the team will be arranged so that the Scrum Master also performs the duties of a typical project manager in order to facilitate the project. The remaining team members will take on various roles throughout the project lifecycle and will be collectively referred to as the development team. The following graphic depicts the organizational structure that will be used throughout the project.

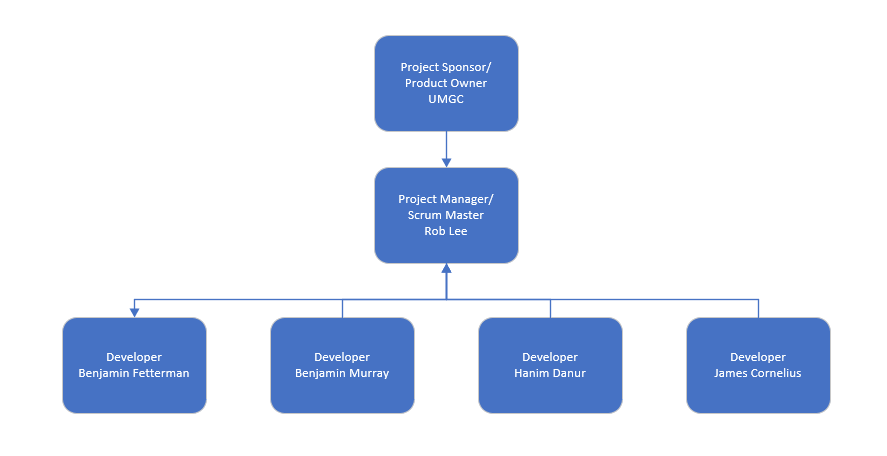


Figure 1 Scrum Project Structure

* + 1. Roles and Responsibilities

The following table describes the various roles throughout the project lifecycle and the responsibilities that are assigned to those roles for the duration of the project. The developer team will act in multiple capacities including doing the development of the project and all testing in the project.

Table 2 Roles and Responsibilities

| Role | Assignee | Responsibility |
| --- | --- | --- |
| Project Owner/Sponsor | Professor Assadullah | The project sponsor is responsible for dictating the views and requirements of the end product to ensure the end result of the project meets the expectations of all involved and provides a working product that adds value. |
| Project Manager/Scrum Master | Robert Lee | The project manager and project sponsor role are responsible for managing the project direction and ensuring the project conforms to the Scrum framework throughout the life of the project. The project manager will also coordinate meetings with the development team and the project sponsor. |
| Development Team | Benjamin Fetterman,  Benjamin Murray,  Hanim Danur,  James Cornelius | The development team is responsible for executing the project and building the deliverables. The development team will also act as the testing team for this project and perform all unit test for the project. |

Responsibility Assigned Matrix (RACI)

This section identifies the responsibilities of individuals across project task life cycle. Although all individuals will be contributing to all components of the project. The chart below shows a loose depiction of owners to project tasks. Responsible staff perform the work. Accountable staff authorizes and guides the task. Consulted staff act as advisors and a source of information. Informed staff is aware of milestone updates and does not enter into detailed decisions of that task.

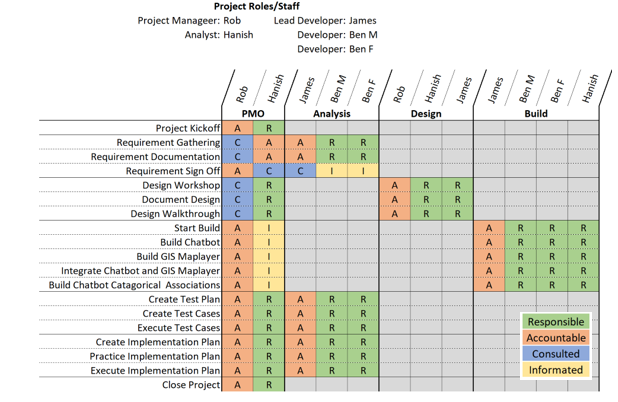


Figure 2 Project Role/Staff

Project Tools

The following table details the tools and services used for developing and testing application development for this project.

Table 3 Project Tools

| Tool Name | Tool Function |
| --- | --- |
| UMGC SWEN 670 Course Material | Course material provided by the UMGS portal will be the primary location for requirements gathering and will facilitate as the delivery mechanism for all project submissions throughout the life of the project. |
| Microsoft Office 365 | The Microsoft Office 365 environment will be used to establish document repositories, a group communication platform, and facilitate group meetings. Additionally, the productivity tools and applications offered on Office 365, including Word, Excel, Visio, and Project will be used to create and update artifacts throughout the life of the project. |
| Eclipse IDE | The Eclipse integrated Development environment (IDE) will be used to write, edit, and compile code into executable content that can be run from both a Mac and Windows operating platform. Eclipse will be used as the primary tool for debugging and testing all code generated for this project. |
| IBM Watson | The IBM Watson API and interface will be used to construct the data flow and responses of the chatbot interface. |
| Github | Github repositories will be used to control code versioning and to aid in the development of the dispersed development team over the course of the project. |

Project Facilities

This project is organized in a way that facilitates the collaboration of remote workers throughout the project lifecycle and will not utilize any specific physical location as its designated working location. All work will be performed using cloud services and remote tools that all developers have access to.

Project Documents Storage

All code and project documents for this project will be stored in the cloud using Microsoft Teams and OneDrive for documents and Github for all code generated. All of the code generated for this project will be stored and remotely accessed using a Github repository specifically created for this project in an effort to both increase collaborative working and focus on quality code. The use of the cloud in this manner helps facilitate a dispersed work force and provides them the ability to collaborate in real time on various project deliverables without the need to be collocated.

Distribution Statement and Security Classification Markings for Work Products

All code, project documents, and milestone submissions for this project are intended to be open source and are freely available to future project teams, and all, to take the work that has been done here and build upon it.

Knowledge/Skills Needed

All project team members are versed in applying Scrum project management methodology principles throughout the software development lifecycle (SDLC). In addition to applying these principles, all team members have the requisite experience necessary to design, develop, test, and deploy applications created in the Java, JavaScript, and HTML Programing languages. Finally, all project team members will be capable of developing Unit Test and functional test to test the developed code against specified requirements.

Project Schedule

* + 1. Work Breakdown Structure

The following Work Breakdown Structure (WBS) will be used as the initial guiding schedule for project development. This WBS in its simplest state specifies the individual milestones and their delivery date. In addition to the milestones, the WBS provides an intended format for the sprints that will be used throughout this project to deliver specific functionality to the stakeholders at the earliest opportunity. This work breakdown structure is detailed in overview below and in a more in-depth manner in Appendix B. The WBS will be maintained and updated throughout the life of this project.

Start  
Wed 8/19/20

Finish  
Tue 11/10/20

Aug 23, '20

Sep 6, '20

Sep 20, '20

Oct 4, '20

Oct 18, '20

Nov 1, '20

**Sprint 1**  
Wed 9/2/20 - Thu 9/10/20

**Sprint 2**  
Fri 9/11/20 - Mon 9/21/20

**Sprint 3**  
Tue 9/22/20 - Wed 9/30/20

**Sprint 4**  
Thu 10/1/20 - Fri 10/9/20

**Sprint 5**  
Mon 10/12/20 - Tue 10/20/20

**Sprint 6**  
Wed 10/21/20 - Wed 10/28/20

**Milestone 1**  
Wed 9/9/20 - Mon 9/14/20

**Milestone 2**  
Wed 9/30/20 - Thu 10/8/20

**Milestone 3**  
Wed 10/21/20 - Fri 10/30/20

**Milestone 4**  
Wed 11/4/20 - Mon 11/9/20

**Software Requirement Specification (SRS)**  
Thu 8/27/20 - Tue 9/8/20

**Software Test Plan (STP)**  
Wed 9/9/20 - Wed 9/23/20

**Technical Design Document (TDD)**  
Fri 9/4/20 - Tue 9/29/20

**Programmer Guide (PG)**  
Tue 9/29/20 - Tue 10/20/20

**User Guide (UG)**  
Wed 10/21/20 - Tue 11/3/20

**Test Report**  
Wed 10/21/20 - Tue 11/3/20

**Deploy and Operations Book (OPS)**  
Wed 9/23/20 - Fri 10/16/20

**Launch Chatbot**  
Mon 11/9/20 - Tue 11/10/20

**Finalize Technical Data Package**  
Wed 11/4/20 - Fri 11/6/20

**Engineer Release 2**  
Mon 10/12/20

**Engineering Release 3**  
Thu 10/29/20

**Engineering Release 1**  
Tue 9/22/20

**Conduct Technical Review 1**  
Fri 9/4/20

Conduct Technical Review 4  
Fri 10/30/20

Conduct Technical Review 3  
Fri 10/16/20

Conduct Technical Review 2  
Mon 9/28/20

**Today**

Figure 3 Project Schedule

* + 1. High-Level Schedule

This project is taking place over an eleven-week development period followed by a week of turnover. The project is broken down into four major milestones that deliverables must meet. The delivery cycle is in three week increments with the final stage being a two-week increment. At the conclusion of each milestone the deliverables are submitted for review and approval is provided prior to working on the subsequent deliverables. At the completion of the final deliverable the project will enter the transition phase where the project is in completed form and the product owner begins to take responsibility for it. While each milestone documentation is being worked, the development team will simultaneously be evaluating and creating the code in six one-week sprints. Each weeklong sprint will focus on the delivery of a specified number of requirements that will be determined at the start of the individual sprint. Various review cycles and test intervals are implemented in the project cycle to ensure progress and code quality are in line with the expected outcome.

* + 1. Project Integrated Master Schedule (IMS)

The project master schedule, specified in Appendix B and the above figure, represents the intended development pace throughout the eleven-week SDLC of this project. All milestones in the schedule are fixed dates that cannot be moved in order to meet the specified requirements. The schedule in this document is the initial schedule and will be updated and maintained outside of this project plan.

Estimates for Size, Effort, and Cost

The total effort for this project is not to exceed the part time work of the five team members over the course of the eleven-week development lifecycle. The amount of work put into each sprint will be the functional equivalent to twelve working hours per a developer and the scrum master. This will account for roughly half of the estimated time with the remaining time being allocated to document development and testing. The hours as broken out in the table below represent the time allocated for over the life of the project and can be directly associated with the amount of work planed for in the master schedule.

Table 4 Estimate of Total Hours

| Functional Role | Total Hours |
| --- | --- |
| Scrum Master | **132** |
| Development team | **528** |
| Business Analyst | 165 |
| Developer | 200 |
| Tester | 110 |
| QA | 53 |
| Total Project Hours | **660** |

Those hour estimates have been used to estimate the total expense of the project from a monetary standpoint. The salaries for the roles identified has been pulled from industry sources, including salary.com, and represents the average expense for a mid-career individual in the specified fields. That salary is then divided by the total number of working hours in the year, 2080, to arrive at an estimated cost per hour per an individual. Those estimates are then multiplied by the expected total hours of work to arrive at the estimated total project cost estimate. All details for the calculation can be associated with the work breakdown structure and are listed in the below table.

Table 5 Estimate of Total Cost

| Functional Role | Salary | Cost/HR | Total Hours | Total Cost |
| --- | --- | --- | --- | --- |
| Scrum Master | $103,000 | $49.52 | 132 | $6,536.54 |
| Development team | $109,400 | $52.60 | 528 | $27,770.77 |
|  | **Total Estimate** | | **660** | **$34,307.31** |

Project Budget

Initial funding for this project will be provided upfront for the initial development of requirements and supporting documentation composing the first milestone of the project. Following the first milestone, the budget and scheduled delivery of funds at each millstone for this project will be managed on a weekly basis through the submission of earned value reports that account for the total level of effort put forth each week. These reports will act as supporting documentation for the milestone payments that will compose the funding for the work composing the subsequent milestone after all stakeholders have agreed to the completion of the milestone.

Deliverables

The deliverables listed below identify the deliverable that is due, a description of the deliverable, the milestone of the deliverable, and the due date for that milestone. All deliverables are integral parts of the SDLC and require approval from all stakeholders prior to progressing to the next deliverable.

Table 6 Deliverables

| Milestone | Due Date | Deliverable | Description |
| --- | --- | --- | --- |
| 1 | 9/8/2020 | Project Plan | Provides the individuals involved in the project, their responsibilities, the requirements for the project, the schedule for the project, and how the project will be managed. |
| 1 | 9/8/2020 | Software Requirement Specification (SRSS) | Provides an outline for the expected performance of the software being developed and the requirements it must meet. |
| 2 | 9/29/20 | Technical Design Document | Provides details on the technical attributes to the project including the interaction with external entities and how they are handled as well as internal technical nuances. |
| 2 | 9/29/20 | Software Test Plan (STP) | Provides the approach that will be taken to test the software and outlines the official testing and validation approach that will be used. |
| 3 | 10/20/20 | Programmer Guide (PG) | Provides the coding standards, code repository schema and information necessary to begin developing on the project. |
| 3 | 10/20/20 | Deployment and Operations Book | Provides an overview of the software architecture and provides system administrators with the steps for deploying the application. |
| 4 | 11/3/20 | User Guide (UG) | Provides information on how the users will interact with the application and the steps they can take within the application. |
| 4 | 11/3/20 | Test Report (TR) | Provides the results from test completed in accordance with the STP |

1. Description

Communication

Communication will use several different methods. Internally the team shall communicate with the Microsoft Teams application, email, and google documents. The project team will communicate with the goal of gathering and disseminating vital information. This will aid the team in resolving any issues and keeping every team member up to date about the progress of the project.

In communication with the advisor and stakeholders the Microsoft Teams application and email will be used. This will allow the team to manage the expectations of the stakeholders and facilitate a platform for the stakeholders to bring up any issues and wants that they have.

These communications can and will be increased from the purposed schedule in order to aid the project’s progress. By having this flexibility, the stakeholders will know that if they have an issue or need information that they can bring it to the team and the team will respond quickly and efficiently.

Reviews and Meetings

Table 7 Reviews and Meetings

| Communication | Objective | Medium | Frequency | Attending |
| --- | --- | --- | --- | --- |
| Sprint Planning | To decide what will be accomplished this sprint | Microsoft Teams conference call | Beginning of each sprint | Project Team |
| Sprint Review | To review the previous sprint, to understand if the team accomplished what it set out to do and how to improve. | Microsoft Teams conference call | End of each sprint | Project Team |
| Milestone Analysis | To meet with the advisor, review the milestone and the work done for said milestone. | Microsoft Teams conference call | Thursday before each milestone is due | Advisor  Project Team |
| Milestone Analysis Review | To review and assign work so the milestone deliverables meet the suggested edits and changes of the advisor | Microsoft Teams conference call | Sunday before each milestone is due | Project Team |
| Milestone Presentation | To present the milestone to the stakeholders and gather their feedback | Microsoft Teams conference call | End of each milestone | Project Team  Stakeholder |
| Status Updates | To update the stakeholders and advisors to advancement, difficulties, and expenditures of the team | Email | Weekly | Advisor  Project Team  Stakeholders |
| Informal | To update team members of work done, or request help and insight on work being done | Microsoft Teams channel,  Email | As needed | Project Team |

1. Risk Analysis

Risk management Process

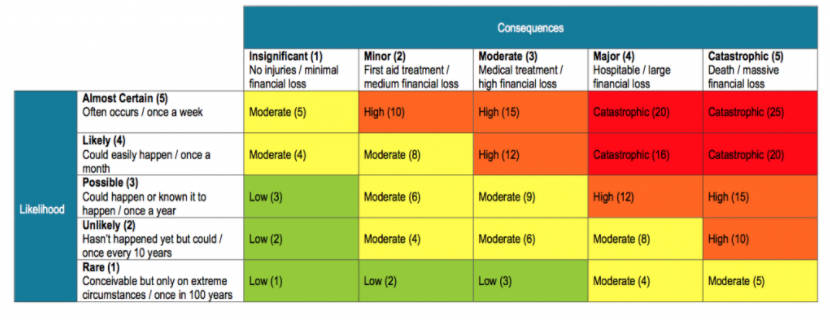
Five steps need to be taken to manage risk; these steps are referred to as the risk management process. The project team begins with identifying risks, go on to analyze risks, then prioritize the risk, implement the solution, and finally, the risks are monitored. The below diagram depicts the five steps the project team will take to manage risk. These project risk management processes are not resource-intensive or difficult for the team to undertake. To first identify the risks, each team member will be requested to contribute a risk in writing during an online scrum meeting. This will have an advantage to the team member to think individually and separately about the risk. During this process, risk events and their relationships will be defined. During analyzing the risk/measure risk, the team will determine the probability and consequence of each risk identified in the first step. Then the team will make an effort to eliminate or contain the risk. As needed the team will connect with the experts in the field to which the risk belongs. During the monitoring result/review risk, the team will identify risks that need to be monitored always.



**(Horvath, 2020)**

Figure 4 Risk Management Process

A risk assessment matrix mechanism will be implemented to increase the visibility of risks and assist management decisions. The process includes numerical values on the hazard and risk identification process. The team will utilize this diagram to define the level of risk by considering the category of probability or likelihood against the category of consequence severity.



**(SafeworkPro, 2019)**

Figure 5 Risk Matrix

Current program risks

Below we have detailed general risk descriptions with possible mitigation tasks that will be observed throughout the life cycle of the project. Identifying risks is an iterative process because new risks may become known as the project progresses through its project life cycle.

Table 8 Risk Descriptions

| **Risk Number** | **Description** | **Mitigation** |
| --- | --- | --- |
| RI.1 | Efficient but not effective: - The chatbot is lacking in “intelligence”, the outcomes could take too long, be over-engineered, have unnecessary restriction, be only partially complete, or simply inaccurate | * Carefully consider the types of tasks that the chatbots can comfortably handle. |
| RI .2 | Insufficient time and staff - project may face insufficiency with time and staffing. Inadequate staff available from external stakeholders until very late in cycle. | * Hiring of new employees. * Hire quality engineers * Engage in change management while collaborating with crucial Stakeholders regarding rescheduling or reprioritizing work. |
| RI.3 | Not detailed out requirements. If the overall system performance is not fully defined, then there lies the possibility that the overall requirements are not fully defined. | * Win-win agreement between parties concerned; business-case analysis; prototyping; application description in early phases. |
| RI.4 | Team communication: - Team communication to address issues and roadblocks may result in blocking of development and progress on the project. Lack of communication can affect overall team performance and productivity, resulting in a slip in the schedule. | * Team needs to openly communicate and sync * Ensure to identity roadblocks and prioritize tasks require team communication * All team participate on creating tasks and required story points. |
| RI.5 | Constant alteration of requirements and poor software quality | * Requirements should be defied well, clear, complete and detailed. * Established requirement review boards with the stakeholders. |

Critical or high priority risks require prompt responses and quick action. They are designated high priority and should be addressed swiftly. Below diagram depicts current project Risk matrix and impact from minor to critical and from 0 to 100% likelihood.

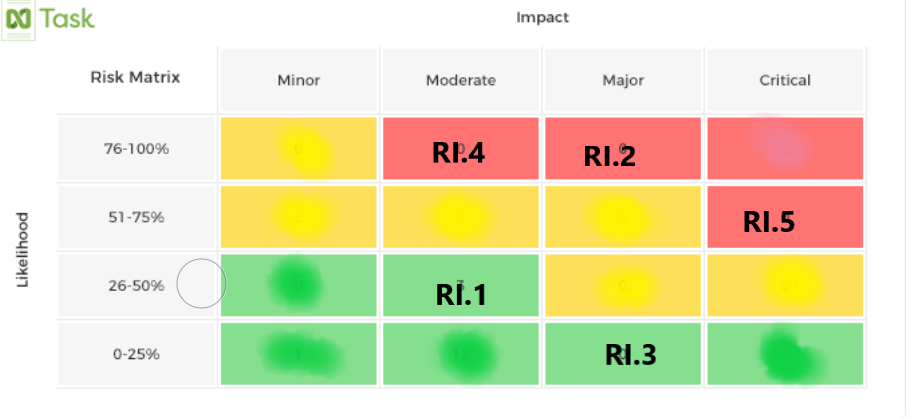


Figure 6 Current Project Matrix

1. Organization of the project

Overview

This project will utilize the Scrum methodology for organizing development activities. The Scrum methodology is a type of Agile methodology that organizes development into a series of iterations called sprints. It was chosen because of its track record of success with projects that are complicated and because of its flexibility in managing development. Several different positions, components, and artifacts make up the Scrum methodology and are used by this project team. Each of these parts of the Scrum methodology are explained in their own subsections below. All version control and build processes are conducted by the UMGC SWEN 670 Fall DevOps team and are explained in their own subsection below.

Scrum Positions

The three positions in the Scrum methodology are the product owner, the development team, and the scrum master. These three positions comprise what is known as the scrum team. The product owner is the individual who is accountable for organizing the items in the team’s product backlog. A product backlog is an important component of the Scrum methodology and is explained in the Scrum Components section. The development team consists of the project members who are responsible for creating the software. The Scrum master is an individual who works with the product owner and the development team in order to help them both achieve their goals.

Scrum Components

Sprints, sprint planning, daily standups, sprint reviews, and sprint retrospectives are the components that make up the Scrum methodology used by this project team. A sprint is a period of time where the sprint team will try to accomplish a set of tasks that fulfill the requirements of the project. Sprints usually have a duration between two to four weeks, but for this project sprint duration is one week due to the short amount of time for development. A sprint planning session is conducted before each sprint in order to plan the tasks that are to be completed in the sprint. Each sprint has a review after it is over where the team goes over what was completed and what was not completed during the sprint and how this affects the product backlog. Besides the review, a separate retrospective is usually conducted where the team discusses how they can work to be better. Because of the availability of team members, the review and retrospective of a sprint are conducted consecutively followed by the planning for the next sprint. These meetings are held virtually using Microsoft Teams. Daily standups are short meetings for monitoring the progress of each member tasks in the sprint.

Scrum Artifacts

The product backlog and the sprint backlog are the Scrum artifacts produced by this project team. A product backlog contains all the work that still needs to be completed by the project team. Work on items in the product backlog can only begin once they have enough information and accurate effort estimates. The product owner and development team work together to provide more information and effort estimates for items that need them. A product owner has the responsibility of organizing the items that are in the product backlog. A sprint backlog contains the items that the development team proposes to accomplish within a sprint. It serves to track the tasks that have been completed and the tasks that still need to be completed. Both a product backlog and a sprint backlog are utilized in this project to organize and prioritize the tasks for each sprint. The product backlog is initially developed from the requirements of the project and updated as the project goes through development. The team uses this product backlog to create a sprint backlog for each of the planned sprints.

Version Control and Build Process

Version control and a build process are both important components when developing software. The UMGC SWEN 670 Fall DevOps team will assist this project with managing its version control system and its build process. A GitHub repository has been set up for the team to host the code for this project. Before committing any code to the main branch in the repository, a pull request needs to be opened for the code changes. At least one member from the development team and one member from the DevOps team must approve the code changes. Project documents are worked on in a team Google Drive and final submissions are submitted to the team’s Microsoft OneDrive account that was set up by the DevOps team. The DevOps team is responsible for creating the build process and assisting the project team with it.

1. Test Plan Identifier

Municipal Permit Chabot System Test Plan

Introduction

The application assists the visitors of the City of Pasadena’s Zoning and Planning website, with finding zone specific permit and regulation information. Visitors are able to question the chatbot for permit and regulation information. Text and speech options are available for residents to communicate with the chatbot. This current version does not integrate speech to text. The system has access to a database containing references to permits and regulations on the city of Pasadena’s website and can provide the appropriate references to residents in response to their questions. Before providing any reference information to residents, the system prompts them for their addresses. This information is needed in order to provide references applicable to the zone that residents live in. The address that a resident enters is verified to be within the jurisdiction of the city of Pasadena and is remembered throughout the duration of the conversation. Below is the use case diagram of the system which contains the expected interactions between residents and the system. This diagram shows the interactions that residents will have with the application and the services that the application will use to fulfil residents’ questions.

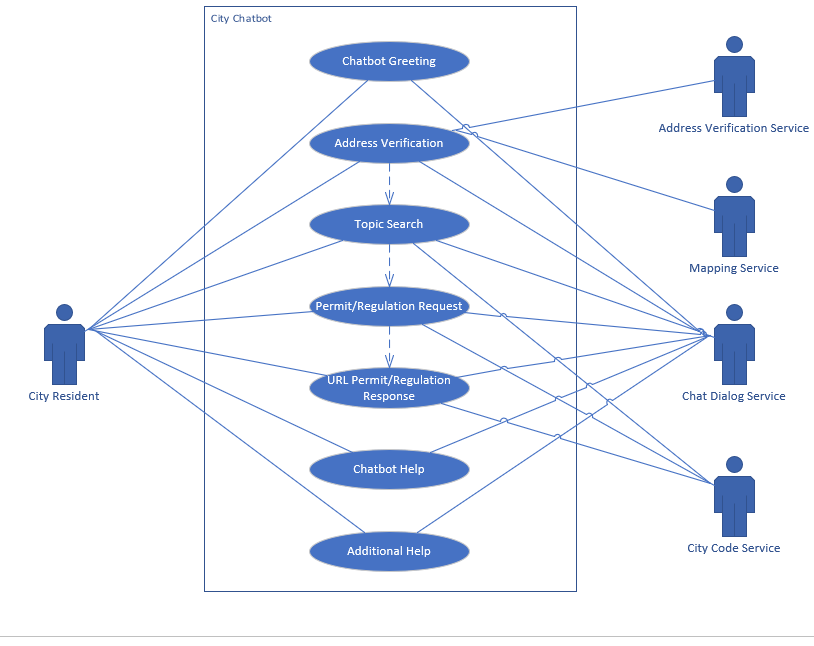


Figure 7 Use Case Diagram of System

The rest of this document contains sections on the features that are being tested, the approach to testing, pass/fail criteria, test deliverables, testing tasks, environmental needs, and responsibilities.

* + 1. Background

The testing plan contains the strategy that the project team will use to test the system that it develops. Testing is required in order to verify that the chatbot system meets the requirements specified in the Software Requirements Specification (SRS) document. Additionally, testing will assist with finding bugs with the system. Testers of the system will use the procedures specified in this document throughout all testing activities.

* + 1. Objectives of Test Plan

The objectives of the testing plan document are listed below.

* Provide information on the purpose and need of the test plan.
* List the features that will and will not be tested.
* Specify the project team’s approach to developing, running, and reviewing tests.
* Describe the environment that tests will run in.
* Describe testing tasks and deliverables.
* List who is responsible for testing tasks.
  + 1. Objectives of UAT

The objectives of the user acceptance testing plan are listed below.

* Verify that the system meets the requirements specified in the SRS document.
* Verify that the system does not contain bugs that would impede users from using the system.

Scope

* + 1. Test Items

The specific items to be tested and that will not be tested are specified in the following sections. The items to be tested have been identified as falling exclusively within the scope of the overall project and do not include features or components that reside outside of the scope of the individual implementation of the system.

* + 1. Features to Be Tested

The following features represent the functional test that will be tested during test of this project:

Table 9 Functional Test Summary

| Test ID | Test Name | Functional Requirement ID |
| --- | --- | --- |
| Test-1.1 | Chatbot Greeting | CH-2 |
| Test-2.1 | Chatbot Zoning/Permit Links | CH-3 |
| Test-2.2 | Chatbot Application Links | CH-4 |
| Test-2.3 | Chatbot Retrieve Regs | CH-5 |
| Test-2.4 | Chatbot Retrieve Standards | CH-6 |
| Test-3.1 | Chatbot Retrieve Permits Applicable to Zoning Statues | CH-7 |
| Test-3.2 | Chatbot GIS Map Layer Integration | CH-8 |
| Test-4.1 | Chatbot Address Requirements | CH-9 |
| Test-4.2 | Address Validation | CH-10 |
| Test-5.1 | Chatbot Other Information | CH-11 |
| Test-5.2 | Chatbot Help | CH-13 |

* + 1. Features Not to Be Tested

Features that are outside the scope of this project and testing include the integration of the application with a backend platform allowing the update of map locations and permit data as it pertains to individual cities and their specific implementation.

Test Tools and Resources

Testing tools and resources provide an array of solutions. Combined with the testing approach, testing tools and resources are designed to expedite, organize, and simplify the testing process. Tests are designed to discover unknown functional and system bugs and conflicts.

* + 1. Testing Tools
* JIRA – A project management software tool. This tool follows an Agile/Scrum process. Utilizing sprints, backlogs, Kanban boards, etc.
* Selenium 4.0.0 Alpha – An automated testing tool. Selenium is a browser based black box testing tool.
* Postman v7.33.0 - Application for testing API. It involves the collection of APIs and checking if they meet expectations for functionality, reliability, performance and security and returns the correct response.
  + 1. Operating Systems and Browsers
* Operating Systems: Windows 10(v2004), Linux, MacOS(vCatalina), iOS(v14), Android(v10), etc.
* Browsers: Chrome(v85), Firefox(v81), Safari(v14), Edge(v85), Opera(v71)
  + 1. Hardware
* Laptop: Lenovo Thinkpad, Microsoft Surface, HP Envy, Dell Inspiron, etc.
* Tablets: Microsoft Surface Go, Samsung Galaxy Tab, iPad, etc.

Testing Approach

A test approach is the test procedure implementation of a project and describes how testing would be carried out.

* + 1. Develop Tests
* Acceptance testing: Acceptance testing is testing the application against business requirements. Acceptance testing is a functional test. This development cycle will not include user acceptance testing, where real end users will test the application against their business requirements.
* Unit testing: Unit testing are component level tests within the application architecture. This testing shall be performed during normal development (coding phase) of chatbot application by developers.
* Functional testing: Functional tests vet functional and non-functional tests. This includes communication between the various application layers.
* Regression testing: Testing components and operability by tracing workflow and system components, by verifying changes not impacting existing functionality.
  + 1. Activities of test development process includes:

The test development process are a series of steps and procedures used to create the test plan with testing details.

* Requirement Analysis: Analyze requirement form specification document and use cases.
* Scenario Development: Create and list down high-level test scenarios.
* Define acceptance criteria: Fulfill the defined criteria of the chatbot and its acceptance by user
* Test Case Development: Based on expected input against expected output.
* Writing Test Scripts: The creation of an executable form of test to identify any deviance in the actual behavior of the program.
* Test Documentation Review: Revaluate testing documentation.

* + 1. QA Reviews

Teams will conduct the following QA reviews: review test cases, test progress & milestones, test deliverables such as test results, reports, defect report, test procedure guidelines.

* + 1. Test Preparation

Test preparation ensures that the various components of the application are in functioning order. Test preparation consists of the following tasks:

* Test Environment: Stand up of the java Spring.io framework and the AI Watson Chatbot; as well as internet connectivity and required local and webservices accounts.
* Test Data: Stand up and connectivity of the database with test data.
* Testing documentation:Finalization of testing documentation and procedures and steps.
  + 1. Run Test

Teams will run the test using the created test scripts and processes from the test case, create a bug tracking system/use a bug tracking tool to trace specific failed test cases, trace the bug status, and record test results.

* + 1. Review Test Results

Reviewing test results is an after-action review to identify the findings of each tests, and creating a plan to remediate deviations from requirements, technical bugs, and other impediments to application functioning in a operable manner. The following process will be conducted when testing is completed:

* Testing against business requirements
* Testing against non-functional requirements
* Application layer communication
* Overall Operability
* Defects and Bugs shall be identified, categorized as per severity, and prioritized against the development schedule and system needs

Testing Risk Issues

Software testing is a risk management tool. Testing can be hampered by multiple primary, secondary, and tertiary factors. These factors fall within three categories: schedule risk, technical risk, and operation risk. Below is a more detailed description of these risk types and examples.

* + 1. Schedule Risk due to:
* Scope Creep – Continuous change to requirements expands the project scope.
* Incorrect Distribution of Tasks – Testing resources are not allocated to the needs of the project schedule.
* Violation of specifications – Requirements are overloaded, or the requirements are not exact and precise.
  + 1. Technical Risk
* Development Environment Configuration Conflicts – Testing and development is conducted on separate local or virtual environments.
* Scope Creep – Continuous change to requirements requires continuous rework to update component designs.
* Lack of technical resources – Due to the limited budget, free or open source technical solutions are required.
  + 1. Operational Risk
* Failure to Establish Testing Priorities – Failure to identify dependencies and map relationships can result in false negatives for passed test cases and system modules.
* Conflicting Test Priorities – Having multiple high priority errors to resolve at one time.
* Improper Training – The software tester does not have the skill set, or was not trained, in the application or test tool.
  + 1. Risk Matrix

Table 10 Risk Matrix

| **Item** | **Risk** | **Risk Type** | **Impact** | **Mitigation Plan** |
| --- | --- | --- | --- | --- |
| 1 | Scope Creep | Schedule Risk | Low | The project requirements have been stable, with a few technical and functional changes over the entirety of the City Chatbot development project. Changes to requirements are maintained within a change control process. |
| 2 | Incorrect Distribution of Tasks | Schedule Risk | Low | The distribution of project tasks are discussed as a group and resources volunteer for tasks. |
| 3 | Violation of Specifications | Schedule Risk | Medium | Specifications are finalized through a review process. Draft à Peer Review à Advisor/Mentor Review à Primary Stakeholder |
| 4 | Development Environment Configuration Conflicts | Technical | Low | All of the technical solutions are common and in wide use. These solutions are compatible with leading operating systems such as Windows 10, MacOS, etc. |
| 5 | Scope Creep | Technical | Low | The project requirements have been stable, with a few technical and functional changes over the entirety of the City Chatbot development project. Changes to requirements are maintained within a change control process. |
| 6 | Lack of Technical Resources | Technical | Medium | Budgetary constraints limit usable solutions. The market for free and open source technical solutions is broad enough to find a resource. |
| 7 | Failure to Establish Testing Priorities | Operational | Medium | Technical and functional design documentation provides modular level of detail. It does not contain script level descriptions such as pseudo code. |
| 8 | Conflicting Test Priorities | Operational | Medium | Determine severity, complexity, and estimated time to completion. |
| 9 | Improper Training | Operational | Low | Utilize online resources, existing documentation, and subject matter experts. |

Item Pass/Fail Criteria

Each requirement, process and system feature established will be evaluated for testing. The test performed will be a set of established test cases for each feature. Each test will have an established stimulus and expected response the test will be looking for. In each test the stimulus will be performed, the results evaluated, and the observed response will be documented and compared to the expected response. A “passed” test will result when the observed result is equal to the expected result. Anything other than an equal result will be designated as a “failed” test.

* + 1. Severity List

The following defect severity levels will be applied to failed test:

Table 11 Severity List

| **ID** | **Severity Level** | **Description** |
| --- | --- | --- |
| Sev1 | Critical | Impacts business objectives and completion of critical task. Testing should be halted, and defect addressed. |
| Sev2 | Hight | Impact is detrimental to system. Testing should potentially be halted, and defect needs to be addressed before next build. |
| Sev3 | Medium | Defect provides inaccurate response to stimulus. |
| Sev4 | Low | Defect is aesthetic and functionality is not impacted. |
|  | Info | Defect observed at testing that requires further information or work. This can relate to a gap in requirements or an unrelated item observed that needs attention. |

Entry Criteria

Testing will commence when the project meets entry criteria. The two primary criteria for commencement encompass the hardware and software environment, and project documentation. Additionally, all test cases are required to be written, reviewed, and approved prior to entry.

For software, all necessary software is required to be installed and configured to local or virtual environments. As well as all hardware and software require to be in functioning order in which the software resides.

The Software Requirements Specifications (SRS), Software Test Plan (SRP), and Technical Design Document (TDD) must be thoroughly reviewed and approved by the project team and primary stakeholder. The SRS describes the functional requirements. The Software Test Plan contains the test cases, matrix, etc. The TDD describes the technical details of the system’s environment and components.

Suspension Criteria and Resumption Requirements

The testing will be suspended when all test cases have been executed and the results are documented for the Test Summary Report. Individual test will be deemed complete when the testing lead has reviewed test results and assigned the necessary defect level for each test.

In the event a critical defect is identified in testing, the testing lead will halt testing and address the defect with the development team. After the defect has been addressed and a new build is delivered to the testing team, the testing lead will resume the testing.

In the event a high defect is identified in testing, the testing lead will use their discretion as to if testing should be halted and will address the defect with the development team. If testing is halted, testing will only resume after a new build has been delivered. If the testing lead determines other components can still be tested, the testing team will continue to test the applicable test.

Test Deliverables

The deliverables identified below are to be delivered at the completion of all test:

* Test Acceptance Plan - What should be accomplished/tested on the UAT complying with acceptance criteria.
* Test Case – Values of stimulus and expected responses for each test.
* Test Log – The results from running the test.
* Incident Report – Observations from test and any documented results that were unexpected.
* Incident Summary – Summary of all incident reported and their potential impact on the project.
* Test Summary Report – Summary of the test and all testing data.
  + 1. Reports and Content

The reports submitted with the completion of each test will include a summary of the events that occurred during testing, the total number of test under taken during the specified testing round, the total number of test completed for the project, the pass fail ratio for each of those intervals, and a list of the defects found and their status.

* + 1. Format

Test deliverables will be delivered in word format with imbedded tables that list the pass-fail criteria for each test undertaken during the testing period. Supplemental excel tables with the list of all test performed during the project will accompany the deliverable.

Test Tasks

Each test performed will be done in the following order:

* Test plan generated
* Testing specifications documented and delivered to testing team.
* Testing environment set up for specific test.
* Complete specified test.
* Defect Analysis, defect severity assigned, summarization of test conducted, and Test Summary Report generated.
* Test Report delivered.

Responsibilities

The project manager will take on the role of Test Lead and will be responsible for overseeing and facilitating the testing of the system and all components of the project. The lead will work in conjunction with the development team, acting as the testing team, to outline the testing requirements for the UAT. An individual development team member will be identified as the Quality Lead to act as checks and balance system in the testing environment. Together the Quality Lead and Testing Lead will be responsible for signing off on a test report as being complete.

Table 12 Testing Responsibilities

|  |  |
| --- | --- |
| Task | Assigned To |
| Test Plan Generated | Testing Lead, Test Team |
| Testing Specifications | Testing Lead, Test Team |
| Environment Setup | Test Team |
| Testing | Test Team |
| Defect Analysis | Testing Lead, Quality Lead |
| Test Report delivered. | Testing Lead |

Table 13 Testing Assignments

|  |  |  |
| --- | --- | --- |
| Position | Team Member | Responsibilities |
| Testing Lead | Rob Lee | Writing and executing functional test. Signs off on pass-fail and severity criteria. Submits reports. |
| Quality Lead | Ben Murray | Witnessing and validating testing. Confirming pass-fail criteria. Determines status of testing and resumption criteria. |
| Testing Team | Benjamin Fetterman  Hanim Danur  James Cornelius  Ben Murray  Robert Lee | Responsible for conducting testing. |

Summary Test Schedule

The matrix below outlines the schedule for completing the items outlined in this document. Due dates were chosen in order to align with the project schedule in the project plan.

Table 14 Schedule for Test Tasks

| Due Date | Task | Comment |
| --- | --- | --- |
| 09/29/2020 | Test plan generated | Submit to project stakeholder for approval. |
| 09/29/2020 | Testing specifications documented and delivered to testing team | Included in this document |
| 10/09/2020 | Testing environment set up | Start with unit testing for developers and then system integration tests. |
| 10/20/2020 | Milestone 3 Deliverable | Test plan updates and interim test reports delivered for Milestone 3 |
| 10/28/2020 | Complete specified tests | Complete tests by the end of the final development sprint. |
| 11/03/2020 | Defect analysis and test summary report generated | Use severity list in section 7 of this document rating impact of defects |
| 11/03/2020 | Test report delivered | Submit to project stakeholder for approval. |

Test Case and Matrix Preparation

Test cases are prepared by reviewing the functional and nonfunctional requirements specified in the project’s SRS. These requirements are used to determine what the application is expected to perform and thus what needs to be tested. Pass and fail criteria are determined from these expected results. The test team and test lead are responsible for preparing the test cases and pass/fail criteria matrix contained in this document. The test lead and project stakeholder will review these items once they are completed and make recommendations as needed. The project stakeholder has final approval for these items.

Test Case Execution and Reporting

The test team is responsible for writing all test cases which include providing all data necessary to execute the test cases. Once the test cases have been executed, the test team is responsible for including the results in the test report. Any defects found during testing will be included in these results for developers to see and fix. When updates and fixes are made to the applications, any test case that involves those areas of the application shall be retested and shall have the results reported in the test report. The test lead is responsible for guiding the test team with completing these tasks and that they are completed on time.

1. Acronyms and Abbreviations

|  |  |
| --- | --- |
| Acronym/Abbreviation | Definition/References |
| AI | Artificial Intelligence – An application that aims to mimic human intelligence. |
| API | Application Programming Interface |
| CH | Chatbot |
| CU | Conditional Use Permit |
| DevSecOps | Development, Security and Operations – Group of developers responsible for the deployment and security of an application. |
| ECUP | Expressive Use Permit |
| EPSP | East Pasadena Specific Plan |
| ETL | Extract, transfer, and load |
| FGSP | Fair Oaks/Orange Grove Specific Plan |
| GIS | Geographical Information System – System for working with geographical data. |
| HTTP | Hyper Text Transfer Protocol – A network protocol for specifying how servers and clients communicate with each other. |
| KML | Keyhole Markup Language – A markup language for visualizing geographical data. |
| LASP | Lincoln Avenue Specific Plan |
| MCUP | Minor Conditional Use Permit |
| MVC | Model-View-Controller |
| REST API | Representation State Transfer – An API for interacting with data. |
| SRS | Software Requirements Specification |
| UI | User Interface – The part of the application that users use to interact with the application. |
| URL | Uniform Resource Locator |

1. IMS/WBS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| WBS | Task Name | Duration | Start | Finish | Predecessors | Successors |
| 1 | **Chatbot System** | **61 days** | **Wed 8/19/20** | **Tue 11/10/20** |  |  |
| 1.1 | **Milestones** | **56 days** | **Wed 8/19/20** | **Tue 11/3/20** |  |  |
| 1.1.1 | Milestone 1 | 4 days | Wed 9/9/20 | Mon 9/14/20 | 8,9 |  |
| 1.1.2 | Milestone 2 | 7 days | Wed 9/30/20 | Thu 10/8/20 | 10,11 |  |
| 1.1.3 | Milestone 3 | 9 days | Wed 10/21/20 | Fri 10/30/20 | 12,13 |  |
| 1.1.4 | Milestone 4 | 4 days | Wed 11/4/20 | Mon 11/9/20 | 14,15 |  |
| 1.2 | **Deliverables** | **41 days** | **Tue 9/8/20** | **Tue 11/3/20** |  |  |
| 1.2.1 | Project Plan | 0 days | Tue 9/8/20 | Tue 9/8/20 | 30 | 3,118 |
| 1.2.2 | SRS Document | 0 days | Tue 9/8/20 | Tue 9/8/20 | 85 | 3,118 |
| 1.2.3 | Software Test Plan | 0 days | Tue 9/29/20 | Tue 9/29/20 | 91 | 4,118 |
| 1.2.4 | Technical Design Document | 0 days | Tue 9/29/20 | Tue 9/29/20 | 96 | 4,118 |
| 1.2.5 | Programmer Guide | 0 days | Tue 10/20/20 | Tue 10/20/20 | 101 | 5,118 |
| 1.2.6 | Deployment and Operation Book | 0 days | Tue 10/20/20 | Tue 10/20/20 | 106 | 5,118 |
| 1.2.7 | User Guide | 0 days | Tue 11/3/20 | Tue 11/3/20 | 111 | 6,118 |
| 1.2.8 | Test Reports | 0 days | Tue 11/3/20 | Tue 11/3/20 | 116 | 6,118 |
| 1.2.9 | **Software Releases** | **29 days** | **Tue 9/22/20** | **Thu 10/29/20** |  | **118** |
| 1.2.9.1 | Engineering Release 1 | 1 day | Tue 9/22/20 | Tue 9/22/20 | 63 |  |
| 1.2.9.2 | Engineer Release 2 | 1 day | Mon 10/12/20 | Mon 10/12/20 | 71 |  |
| 1.2.9.3 | Engineering Release 3 | 1 day | Thu 10/29/20 | Thu 10/29/20 | 79 |  |
| 1.3 | **Initiation Phase** | 4 day | Thu 10/27/20 | Thu 10/29/20 |  |  |
| 1.3.1 | Stakeholder Meeting | 1 day | Wed 8/19/20 | Wed 8/19/20 |  | 22 |
| 1.3.2 | Team Introduction | 3 days | Mon 8/24/20 | Wed 8/26/20 | 21 | 23 |
| 1.3.3 | Initial Team Meeting | 1 day | Thu 8/27/20 | Thu 8/27/20 | 22 |  |
| 1.3.4 | Project Kick-Off Meeting | 0 days | Thu 8/27/20 | Thu 8/27/20 |  | 27,54,56,57 |
| 1.4 | **Project Planning** | **15 days** | **Wed 8/19/20** | **Tue 9/8/20** |  |  |
| 1.4.1 | **Project Plan** | **9 days** | **Thu 8/27/20** | **Tue 9/8/20** |  |  |
| 1.4.1.1 | Create Project Plan | 4 days | Thu 8/27/20 | Tue 9/1/20 | 24 | 28 |
| 1.4.1.2 | Conduct Internal Review | 4 days | Wed 9/2/20 | Mon 9/7/20 | 27 | 29 |
| 1.4.1.3 | Closeout Internal Review | 1 day | Tue 9/8/20 | Tue 9/8/20 | 28 | 30 |
| 1.4.1.4 | Submit Project Plan Deliverable | 0 days | Tue 9/8/20 | Tue 9/8/20 | 29 | 8 |
| 1.5 | **Monitoring and Controlling** | **56 days** | **Wed 8/19/20** | **Tue 11/3/20** |  |  |
| 1.5.1 | **Stakeholder Meetings** | **56 days** | **Wed 8/19/20** | **Tue 11/3/20** |  |  |
| 1.5.1.1 | **Technical Review #1** | **15 days** | **Wed 8/19/20** | **Tue 9/8/20** |  |  |
| 1.5.1.1.1 | Create Draft Presentation | 5 days | Tue 8/25/20 | Mon 8/31/20 |  |  |
| 1.5.1.1.2 | Submit Slides to Stakeholder | 1 day | Thu 9/3/20 | Thu 9/3/20 |  | 36 |
| 1.5.1.1.3 | Conduct Technical Review 1 | 1 day | Fri 9/4/20 | Fri 9/4/20 | 35 | 39,37 |
| 1.5.1.1.4 | Obtain Stakeholder Approval to Progress | 2 days | Mon 9/7/20 | Tue 9/8/20 | 36 |  |
| 1.5.1.2 | **Technical Review #2** | **15 days** | **Wed 9/9/20** | **Tue 9/29/20** |  |  |
| 1.5.1.2.1 | Create Draft Presentation | 5 days | Wed 9/9/20 | Tue 9/15/20 | 36 |  |
| 1.5.1.2.2 | Submit Slides to Stakeholder | 1 day | Mon 9/28/20 | Mon 9/28/20 |  | 41 |
| 1.5.1.2.3 | Conduct Technical Review 2 | 0 days | Mon 9/28/20 | Mon 9/28/20 | 40 | 44,42 |
| 1.5.1.2.4 | Obtain Stakeholder Approval to Progress | 1 day | Tue 9/29/20 | Tue 9/29/20 | 41 |  |
| 1.5.1.3 | **Technical Review #3** | **15 days** | **Wed 9/30/20** | **Tue 10/20/20** |  |  |
| 1.5.1.3.1 | Create Draft Presentation | 5 days | Wed 9/30/20 | Tue 10/6/20 | 41 | 45 |
| 1.5.1.3.2 | Submit Slides to Stakeholder | 1 day | Fri 10/16/20 | Fri 10/16/20 | 44 | 46 |
| 1.5.1.3.3 | Conduct Technical Review 3 | 0 days | Fri 10/16/20 | Fri 10/16/20 | 45 | 49,47 |
| 1.5.1.3.4 | Obtain Stakeholder Approval to Progress | 1 day | Tue 10/20/20 | Tue 10/20/20 | 46 |  |
| 1.5.1.4 | **Technical Review #4** | **11 days** | **Wed 10/21/20** | **Tue 11/3/20** |  |  |
| 1.5.1.4.1 | Create Draft Presentation | 3 days | Wed 10/21/20 | Fri 10/23/20 | 46 |  |
| 1.5.1.4.2 | Submit Slides to Stakeholder | 1 day | Fri 10/30/20 | Fri 10/30/20 |  | 51 |
| 1.5.1.4.3 | Conduct Technical Review 4 | 0 days | Fri 10/30/20 | Fri 10/30/20 | 50 | 52 |
| 1.5.1.4.4 | Obtain Stakeholder Approval to Progress | 1 day | Mon 11/2/20 | Mon 11/2/20 | 51 |  |
| 1.6 | **Execution Phase** | **52 days** | **Tue 8/25/20** | **Tue 11/3/20** |  |  |
| 1.6.1 | Create Product Backlog | 3 days | Fri 8/28/20 | Tue 9/1/20 | 24 | 57 |
| 1.6.2 | **Sprints (weekly)** | **42 days** | **Wed 9/2/20** | **Wed 10/28/20** |  |  |
| 1.6.2.1 | **Sprint 1** | **7 days** | **Wed 9/2/20** | **Thu 9/10/20** | **24** |  |
| 1.6.2.1.1 | Sprint Planning | 1 day | Wed 9/2/20 | Wed 9/2/20 | 24,54 | 58,93 |
| 1.6.2.1.2 | Sprint Execution | 6 days | Thu 9/3/20 | Thu 9/10/20 | 57 | 59 |
| 1.6.2.1.3 | Sprint End/Demo | 0 days | Thu 9/10/20 | Thu 9/10/20 | 58 | 61 |
| 1.6.2.2 | **Sprint 2** | **7 days** | **Fri 9/11/20** | **Mon 9/21/20** |  |  |
| 1.6.2.2.1 | Sprint Planning | 1 day | Fri 9/11/20 | Fri 9/11/20 | 59 | 62,87 |
| 1.6.2.2.2 | Sprint Execution | 6 days | Mon 9/14/20 | Mon 9/21/20 | 61 | 63 |
| 1.6.2.2.3 | Sprint End/Demo | 0 days | Mon 9/21/20 | Mon 9/21/20 | 62 | 65,17 |
| 1.6.2.3 | **Sprint 3** | **7 days** | **Tue 9/22/20** | **Wed 9/30/20** |  |  |
| 1.6.2.3.1 | Sprint Planning | 1 day | Tue 9/22/20 | Tue 9/22/20 | 63 | 66,98,103 |
| 1.6.2.3.2 | Sprint Execution | 6 days | Wed 9/23/20 | Wed 9/30/20 | 65 | 67 |
| 1.6.2.3.3 | Sprint End/Demo | 0 days | Wed 9/30/20 | Wed 9/30/20 | 66 | 69 |
| 1.6.2.4 | **Sprint 4** | **7 days** | **Thu 10/1/20** | **Fri 10/9/20** |  |  |
| 1.6.2.4.1 | Sprint Planning | 1 day | Thu 10/1/20 | Thu 10/1/20 | 67 | 70 |
| 1.6.2.4.2 | Sprint Execution | 6 days | Fri 10/2/20 | Fri 10/9/20 | 69 | 71 |
| 1.6.2.4.3 | Sprint End/Demo | 0 days | Fri 10/9/20 | Fri 10/9/20 | 70 | 73,18 |
| 1.6.2.5 | **Sprint 5** | **7 days** | **Mon 10/12/20** | **Tue 10/20/20** |  |  |
| 1.6.2.5.1 | Sprint Planning | 1 day | Mon 10/12/20 | Mon 10/12/20 | 71 | 74,108 |
| 1.6.2.5.2 | Sprint Execution | 6 days | Tue 10/13/20 | Tue 10/20/20 | 73 | 75 |
| 1.6.2.5.3 | Sprint End/Demo | 0 days | Tue 10/20/20 | Tue 10/20/20 | 74 | 77 |
| 1.6.2.6 | **Sprint 6** | **7 days** | **Wed 10/21/20** | **Wed 10/28/20** |  |  |
| 1.6.2.6.1 | Sprint Planning | 1 day | Wed 10/21/20 | Wed 10/21/20 | 75 | 78 |
| 1.6.2.6.2 | Sprint Execution | 6 days | Thu 10/22/20 | Wed 10/28/20 | 77 | 79 |
| 1.6.2.6.3 | Sprint End/Demo | 0 days | Wed 10/28/20 | Wed 10/28/20 | 78 | 19 |
| 1.6.3 | **Documentation Execution** | **52 days** | **Tue 8/25/20** | **Tue 11/3/20** |  |  |
| 1.6.3.1 | **Software Requirement Specification (SRS)** | **9 days** | **Thu 8/27/20** | **Tue 9/8/20** |  |  |
| 1.6.3.1.1 | Create Draft SRS Document | 5 days | Tue 8/25/20 | Mon 8/31/20 |  | 83 |
| 1.6.3.1.2 | Conduct Internal Review - SRS | 2 days | Wed 9/2/20 | Thu 9/3/20 | 82 | 84 |
| 1.6.3.1.3 | Closeout Internal Review- SRS | 1 day | Mon 9/7/20 | Mon 9/7/20 | 83 | 85 |
| 1.6.3.1.4 | Submit SRS Document to Stakeholder | 0 days | Tue 9/8/20 | Tue 9/8/20 | 84 | 9,87 |
| 1.6.3.2 | **Software Test Plan (STP)** | **11 days** | **Wed 9/9/20** | **Wed 9/23/20** |  |  |
| 1.6.3.2.1 | Create Initial Draft Test Plan | 2 days | Tue 9/15/20 | Wed 9/16/20 | 61,85 | 88 |
| 1.6.3.2.2 | Conduct Internal Review - TP | 3 days | Thu 9/17/20 | Mon 9/21/20 | 87 | 89 |
| 1.6.3.2.3 | Closeout Internal Review- TP | 1 day | Tue 9/22/20 | Tue 9/22/20 | 88 | 90 |
| 1.6.3.2.4 | Incorporate Test Plan into Project Plan | 1 day | Wed 9/23/20 | Wed 9/23/20 | 89 | 91 |
| 1.6.3.2.5 | Submit Test Plan Material to Stakeholder | 0 days | Wed 9/23/20 | Wed 9/23/20 | 90 | 10 |
| 1.6.3.3 | **Technical Design Document (TDD)** | **18 days** | **Fri 9/4/20** | **Tue 9/29/20** |  |  |
| 1.6.3.3.1 | Create Draft TDD Document | 10 days | Thu 9/3/20 | Wed 9/16/20 | 57 | 94 |
| 1.6.3.3.2 | Conduct Internal Review - TDD | 4 days | Thu 9/17/20 | Tue 9/22/20 | 93 | 95 |
| 1.6.3.3.3 | Closeout Internal Review- TDD | 1 day | Tue 9/29/20 | Tue 9/29/20 | 94 | 96 |
| 1.6.3.3.4 | Submit TDD Document to Stakeholder | 0 days | Tue 9/29/20 | Tue 9/29/20 | 95 | 11 |
| 1.6.3.4 | **Programmer Guide (PG)** | **16 days** | **Tue 9/29/20** | **Tue 10/20/20** |  |  |
| 1.6.3.4.1 | Create Draft PG | 11 days | Tue 9/29/20 | Tue 10/13/20 | 65 | 99 |
| 1.6.3.4.2 | Conduct Internal Review - PG | 4 days | Wed 10/14/20 | Mon 10/19/20 | 98 | 100 |
| 1.6.3.4.3 | Closeout Internal Review- PG | 1 day | Tue 10/20/20 | Tue 10/20/20 | 99 | 101 |
| 1.6.3.4.4 | Submit PG Document to Stakeholder | 0 days | Tue 10/20/20 | Tue 10/20/20 | 100 | 12 |
| 1.6.3.5 | **Deploy and Operations Book (OPS)** | **18 days** | **Wed 9/23/20** | **Fri 10/16/20** |  |  |
| 1.6.3.5.1 | Create Draft OPS Document | 13 days | Wed 9/23/20 | Fri 10/9/20 | 65 | 104 |
| 1.6.3.5.2 | Conduct Internal Review - OPS | 4 days | Mon 10/12/20 | Thu 10/15/20 | 103 | 105 |
| 1.6.3.5.3 | Closeout Internal Review- OPS | 1 day | Fri 10/16/20 | Fri 10/16/20 | 104 | 106 |
| 1.6.3.5.4 | Submit OPS Document to Stakeholder | 0 days | Fri 10/16/20 | Fri 10/16/20 | 105 | 13 |
| 1.6.3.6 | **User Guide (UG)** | **11 days** | **Wed 10/21/20** | **Tue 11/3/20** |  |  |
| 1.6.3.6.1 | Create Draft UG Document | 5 days | Wed 10/21/20 | Mon 10/26/20 | 73 | 109 |
| 1.6.3.6.2 | Conduct Internal Review - UG | 4 days | Tue 10/27/20 | Fri 10/30/20 | 108 | 110 |
| 1.6.3.6.3 | Closeout Internal Review- UG | 1 day | Mon 11/2/20 | Mon 11/2/20 | 109 | 111 |
| 1.6.3.6.4 | Submit UG Document to Stakeholder | 0 days | Mon 11/2/20 | Mon 11/2/20 | 110 | 14 |
| 1.6.3.7 | **Test Report** | **11 days** | **Wed 10/21/20** | **Tue 11/3/20** |  |  |
| 1.6.3.7.1 | Create Draft TR Document | 3 days | Thu 10/22/20 | Sat 10/24/20 |  | 114 |
| 1.6.3.7.2 | Conduct Internal Review - TR | 1 day | Mon 10/26/20 | Mon 10/26/20 | 113 | 115 |
| 1.6.3.7.3 | Closeout Internal Review- TR | 1 day | Tue 10/27/20 | Tue 10/27/20 | 114 | 116 |
| 1.6.3.7.4 | Submit TR Document to Stakeholder | 0 days | Tue 10/27/20 | Tue 10/27/20 | 115 | 15 |
| 1.7 | **Project Closeout** | **5 days** | **Wed 11/4/20** | **Tue 11/10/20** |  |  |
| 1.7.1 | Finalize Technical Data Package | 3 days | Wed 11/4/20 | Fri 11/6/20 | 8,9,10,11,12,13,14,15,16 | 119 |
| 1.7.2 | Launch Chatbot | 2 days | Mon 11/9/20 | Tue 11/10/20 | 118 |  |

APPENDIX C: References

Horvath, I. (2020, July 5) Five steps of risk management process Retrieved from <https://www.invensislearning.com/blog/risk-management-process-steps/>

Safeworkpro. (2019, Jan 4) An example of risk assessment matrix Retrieved from <https://www.safeworkpro.com.au/an-example-of-risk-assessment-matrix/>

Wilson, F. (2020, July 22) How to use the risk assessment matrix in project management? Retrieved from <https://www.ntaskmanager.com/blog/risk-assessment-matrix/>