Deliverable I

Melanie Brady
Project 4: Messaging Web Application
COP4331 Fall 2020

TABLE OF CONTENTS

Concept of Operations

Melanie Brady

Project 4: Messaging Web Application

COP4331 Fall 2020

Contents of this Document

The Current System

The Messaging Web Application will allow for two users to register for accounts, search for users they may know on the application, send an instant message to any other user, all stored within a secure database. Software is all about streamlining daily inconveniences. A messaging web application allows for the user to message any other user from any location if they both users have access to the internet. There a lot of different options of messaging apps some have self-destructing messages, can only send videos, only send text, and chat only with friends. Some messaging apps are integrated as a part of other social media such as Discord can partner with your Steam activity, Messenger is a part of Facebook, Google Chat is linked with your Google Account, and so on.

The Proposed System: Motivation

There are thousands of instant messaging applications with countless bells and whistles, but they often over complicate humans most essential need – and that's connection. Sometimes people just need someone to talk to, connect with, and ease the burden of loneliness. That is what this messaging application will try to tackle. It will not have as many bells and whistles as a trillion-dollar company's messaging client will, but it will try it is best to help the users connect with deep conversation prompts, scheduled calls with anyone in the world, and it's focused design.

This app will be minimalistic and only focus on the essentials which makes it streamlined and easier to use. Unlike the trillion-dollar company's, it will not collect your data beyond your user credentials and temporarily your messages. There will be more privacy due to not having all your data stored and sold to advertisers. If more funding is needed, then there will be advertisements, but not personalized to maintain it's free to use status. It will allow users to focus on the most important part of a messaging application and that is the connection with the other user.

The Proposed System: Users and Modes of Operation

- User – Instant message, call, video call, send links, add, and search users of the platform.
- Admin – Same exact features as a free user, but beta mode.
- Not registered – will only be able to access a login page or register as an user page.

For the project, this will be an unpaid version.

The Proposed System: Operational Scenarios

- **Register to become a user:** The person will register a unique login name, email, and password. The database will search if the login name is already taken, if not taken, then the user will register with that account name.
- **Username already exists:** When creating a new account, the database will check first if the username exists, if it already exists, it will inform the user that the username is taken and to try a different username.
- **Incorrect input of username and/or password:** The database will tell them that the combination of username and password does not exist. Then prompt an option to change their password after inputting a valid username an email will be sent to the associated email so that the user can change their password.
- **User searches for another user and add another user as a friend:** The user will type a username into the search bar, then the database will look up that string, if a user is associated with that string, they can add each other as friends otherwise it will tell the user that username is invalid.
- **Send instant messages to other users:** Users can send each other instant messages and send links that will be temporarily stored in the server then pushed to the receiving user.
- **Call users:** Users that are friends with each other can call each other over just audio or audio and voice.
- **User Settings:** User profile will allow for customizations such as their name, interests, languages they speak, location, time zone, profile picture, and so on.
- **User schedule calls with other users:** Schedule a call with other users around the world, mindful of different time zone, the users can choose a topic of interest and/or just be paired up for the time they'd like to speak.
- **Loss of connection**: As soon as the user no longer has internet connection, the application will default to an error messaging informing the user.
- **Crash**: The other category for errors. Output a registered error code and notify the user that "An error has happened, please try again shortly. Thanks for your patience!".

The Proposed System: Operational Features

Must Have:

- Users can register an account with: unique email, unique username, and password. Verifies if the username or email account already exist. Offers reset password prompt.
- User profile settings where they can add first and last name to the account.
- Users can search for other user accounts. Add friends.
- Friends can send each other instant messages.
- Error messages and loss of connection updates.
- Cross platform mobile and web accessibility.
- Encrypted passwords stored in the database.

Would Like to Have:

- Friends can have calls just audio or audio and video.
- Further user profile customization: Description, pictures, languages spoken, languages learning, location, time zone, interests, so on.
- Schedule calls with other users – paired for call time, call type, interests, languages, so on
- Further security measures beyond passwords being encrypted.

The Proposed System: Analysis

The mobile web application will be developed in JavaScript, Java, and Python. Accessible through both web and mobile platforms. Employed React UI, Spring and Express microservices, OAuth authentication, MongoDB credential storage, Socket.io client notification, Agora.io video and audio, Git source control, and AWS EC2 hosted. Generated unit tests using Jest for JavaScript and JUnit and Mockito for Java. Automated deployment to AWS with scripts. At this current stage, there will be four services offered: Authentication, Messaging, Search, and Call. I am comfortable with JavaScript and Java. I would love to learn more Python as well. I am very excited to work with these frameworks as well.

This should be a smooth development environment. It will also have a clean user interface regardless of the platform. There will be a period of learning how to work with those specific APIs and freshening up any JavaScript and Python knowledge. The system is mobile and depends on internet connection. The application can be used anywhere the user has stable internet connection.

Some disadvantages are: Debugging in JavaScript can be tricky sometimes, some browsers interpret JavaScript differently, and that some of the code could be visible client side. The main hurdle with any messaging application is the security. I could use MySQL instead of Mongo, but MongoDB has a lot more flexibility. There will be a bit of a learning curve with ReactJS due to not having the easiest to understand documentation. Beyond that, React has an extremely clean UI that is hard to beat. I could program in python, but JavaScript has much better web support. A common problem with messaging applications is how to have the user receive the message. Some platforms have each user check to see if they have a new message, but by using Socket.io the server will ping the user once they have an update. There are a lot of support forums regarding these frameworks working together so any troubleshooting hurdles have more than likely been addressed before.

Project Management Plan

Melanie Brady

Project 4: Messaging Web Application

COP4331 Fall 2020

Contents of this Document

Project Overview:

Develop a mobile app where users can register with their email and password and send messages to each other. The registered user will have to create a unique username in the messaging app, but they can also add their first and last name to the account, which can be seen by other users. A user can search for another user by their username and send a text message to the receiver. The website should

also be mobile compatible. As long as the user knows the username of the receiver, they can send any message to that person. The passwords should be stored encrypted in the database.

Applicable Standards

- Coding Standard:
  - camelCase
  - Meaningful variable names
  - Handle all errors,
  - For more details, I will follow these style guides:
    - JavaScript: https://google.github.io/styleguide/jsguide.html
    - Java: https://google.github.io/styleguide/javaguide.html
    - Python: https://google.github.io/styleguide/pyguide.html
- Artifact Size Metric Standard
  - 40 classes, 10,000 lines of code, 5 API packages
  - Tangible progress through goals
  - It is difficult at this stage in the development process to determine an exact number of classes required. It could likely be 20 functions for the front end and back end. Progress will be monitored through setting up the database, the website, log in, connecting of the API, and so on.

Deliverables

Artifact	**Due Dates** <some will have multiple deliveries>
Individual Weekly Progress Reports	Weekly (Fridays) submission throughout the semester through webcourses
Concept of Operations	Friday, Sept 18
Software Project Management Plan (SPMP)	Friday, Sept 18
Software Requirements Specification (SRS)	Friday, Sept 18
Test Plan	Friday, Oct 30
High-Level Design	Friday, Oct 30
Detailed Design	Friday, Oct 30
Test Results	Monday, Nov 30
Source, Executable, Build Instructions	Monday, Nov 30

Software Life Cycle Process: Agile & Scrum

To quote the Manifesto for Agile Software Development:

1. Customer satisfaction by early and continuous delivery of valuable software.
2. Welcome changing requirements, even in late development.
3. Deliver working software frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

- https://agilemanifesto.org/

Using the Scrum format, to break the big picture ideas into smaller goals with indexed time slots.

Tools and Computing Environment

- Java
- Python
- JavaScript
- MongoDB
- Node
- Git
- AWS
- Andriod Studio

OS: Linux (WSL & Ubuntu), Windows, Mac

Configuration Management: Git and GitHub. Flow GitFlow.

Quality Assurance: Unit Tests for back end code. Manually test the front end code as more features are integrated. Postman for testing API.

Risk Management: Risk for not completing on time. AWS goes down during demo. Planning on using external libraries, so if it has a bad patch, it could make the project not work as well.

Table of Work Packages, Time Estimates, and Assignments:

- Development (50 hours - Self)
    - Front End - UI
    - Back End
        - Authentication
        - Business Logic (Messaging, Searching)
    - Database - Schema
- Testing (50 hours - Self)
    - Front End - Manual Testing

    - Back End
        - Unit Testing (Jest)
        - Manual Testing (Postman)
- Deployment (50 hours - Self)
    - Deployment Scripts
    - AWS EC2 Configuration

Technical Progress Metrics:

I will measure progress by how many requirements have been completed versus how many requirements are still in the backlog. Each requirement will have its own definition of completion.

Plan for tracking, control, and reporting of progress

- I will write down all tasks needing to get done and their requirements for being completed, then report back my completion rate for that week and the overall project.
- I will post your progress report weekly through webcourses: the weekly report should include time spent in each activity, completed task(s), tasks in-progress, tasks planned for the following week, and individual issues and problems.
- Each week, you read and analyze the logs; examine the technical content of the work done to date; examine the technical progress metrics; consider the QA results; reassess the potential project risks; and take corrective action if necessary.

Software Requirements Specification

Melanie Brady

Project 4: Messaging Web Application

COP4331 Fall 2020

Contents of this Document

Section 1: Introduction

Software to be Produced:

Develop a mobile app where users can register with their email and password and send messages to each other. The registered user will have to create a unique username in the messaging app, but they can also add their first and last name to the account, which can be seen by other users. A user can search for another user by their username and send a text message to the receiver. The website should also be mobile compatible. As long as the user knows the username of the receiver, they can send any message to that person. The passwords should be stored encrypted in the database.

Applicable Standards

- User should be using browsers Chrome, FireFox or Microsoft Edge.

Definitions, Acronyms, and Abbreviations

- None.

Section 2: Product Overview

Assumptions:

- The application will be interacting with a website and database hosted on an AWS server.
- The external APIs will be dependent on third party libraries.
- The mobile app would most likely end up on Google Play.

Stakeholders:

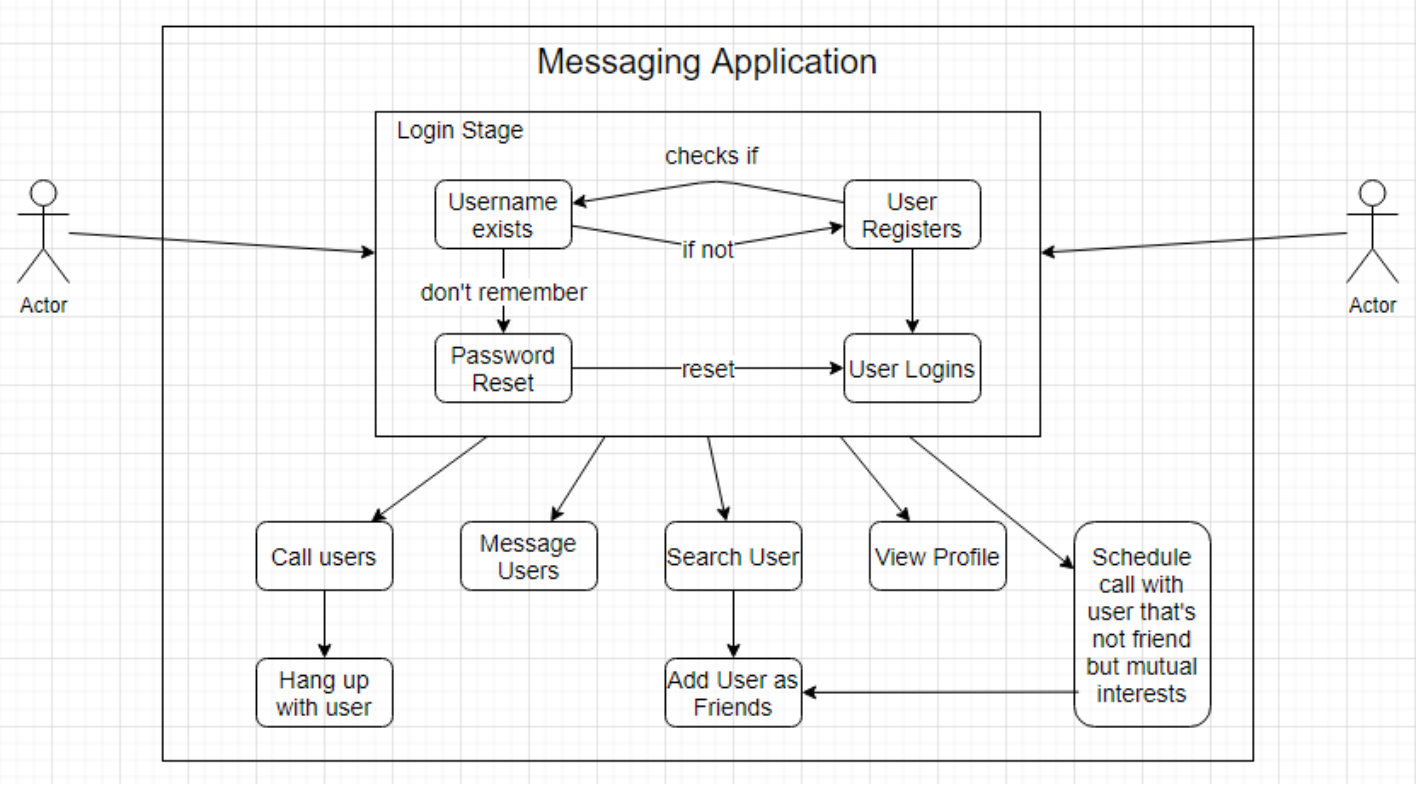
- Users – A future user would want something as streamlined and easy to use as possible.
- Developer – My grade in this class is dependent on how this project turns out.

Event Table:

Event Name	External Stimuli	External Responses	Internal data and state
User Registers	MongoDB & server	Updates database	Account becomes registered
User Logins	MongoDB & server	Verifies Account Info	Authentication of User
Password Reset	MongoDB & server	Updates database	Account updated
View Profile	MongoDB & server	Retrieve user information Modify user profile	Data is updated

Search User	MongoDB & server	Searches through database to find the username	Returns the user profile
Add User as Friend	MongoDB & server	User requests for another user to be their friend	Updates database username to username
Message user	MongoDB & server	User messages another user	Pushes from server to user
Call user	MongoDB & server	User calls another user	Notify users and starts a connection
Schedule call with user	MongoDB & server	Users schedule a call	Creates connection at a scheduled time
Hang up with user	MongoDB & server	Connection ends immediately	Returns to the home page of the app

Use Case Diagram:



Use Case Descriptions:

- **Register to become a user:** The person will register a unique login name, email, and password. The database will search if the login name is already taken, if not taken, then the user will register with that account name.
- **Username already exists:** When creating a new account, the database will check first if the username exists, if it already exists, it will inform the user that the username is taken and to try a different username.
- **Password reset:** The database will tell them that the combination of username and password does not exist. Then prompt an option to change their password after inputting a valid username an email will be sent to the associated email so that the user can change their pw.
- **User Logins:** Users can access other user profiles, search users, update their profile and so on.
- **User Settings:** User profile will allow for customizations such as their name, interests, languages they speak, location, time zone, profile picture, and so on.
- **User searches for another user and add another user as a friend:** The user will type a username into the search bar, then the database will look up that string, if a user is associated with that string, they can add each other as friends otherwise it will tell the user that username is invalid.
- **Send instant messages to other users:** Users can send each other instant messages and send links that will be temporarily stored in the server then pushed to the receiving user.
- **Call users:** Users that are friends with each other can call each other.
- **User schedule calls with other users:** Schedule a call with other users around the world, mindful of different time zone, the users can choose a topic of interest and/or just be paired up for the time they'd like to speak.

Section 3: Specific Requirements

3.1 Functional Requirements

No: 1
Statement: The application shall verify it has stable internet connection.
Source: Client
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: The program will check if there's internet connection. If not then it will notify the user to try again later.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 2
Statement: The user can create a unique username and password.
Source: Client
Dependency: Requirement 1
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Check the database to see if the username exists. If the username does not exist, then update the database with the new username and password.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 3
Statement: The user can log in using their username and password.
Source: Client
Dependency: Requirements 1 & 2
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify the user can log in with their credentials.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 4
Statement: The user can search for other registered users.
Source: Client
Dependency: Requirements 1-3

Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can successfully search for other users.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 5
Statement: Users can message other users.
Source: Client
Dependency: Requirements 1-4
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can send and receive messages from other users.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 6
Statement: Users can customize their profile
Source: Client
Dependency: Requirements 1-3
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can store their first name, last name, time zone, location, interests, so on…
Revision History: Melanie Brady – 9/18/2020 – Created

No: 7
Statement: Users can call other users.
Source: Client
Dependency: Requirements 1-4
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can send and receive messages from other users.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 8
Statement: Users can schedule calls with other users regardless of friend status with an optional conversation prompt based off of their interests.
Source: Client
Dependency: Requirements 1-4
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can schedule calls with other users mindful of their time zone.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 9
Statement: Users can reset their password.
Source: Client
Dependency: Requirements 1-2

Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: User can reset their password.
Revision History: Melanie Brady – 9/18/2020 – Created

3.2 Interface Requirements

No: 10
Statement: All data stored in the database will be accurate.
Source: Client
Dependency: None.
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Compare data stored in the database compared to the user input.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 11
Statement: When logging in, the database should return the correct profile's information.
Source: Client
Dependency: Requirements 1-4 & 10.
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify the correct profile is retrieved from the database.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 12
Statement: Verify the server is pushing newly sent messages to the receiving user.
Source: Client
Dependency: Requirements 1-11.
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify messages are successfully received right after sending.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 13
Statement: Verify data stored in the database is being stored as the correct data type.
Source: Client
Dependency: Requirements 1-12.
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Username, password, location as strings. Time zone in int. Interests as either Boolean or strings.
Revision History: Melanie Brady – 9/18/2020 – Created

3.3 Physical Environment Requirements

No: 14
Statement: The programs should be able to run as long as they are connected to the internet.

Source: Client
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify it works with internet connection regardless of location.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 15
Statement: The programs should be able to work on Android devices.
Source: Client
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Test on android devices.
Revision History: Melanie Brady – 9/18/2020 – Created

3.4 User and Human Factors Requirements

No: 16
Statement: The application will have user profile.
Source: Requirements 1-4 & 6
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify user profile has successfully been created.

Revision History: Melanie Brady – 9/18/2020 – Created

No: 17
Statement: The application will have admin profile.
Source: Requirements 1-4 & 6 & 16
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify admin profile has successfully been created to test beta mode.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 18
Statement: Users are expected to be familiar with Android and websites. If not, there will be instructions for how to use the app.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Test with different users to verify they have a basic understanding of Android devices and websites to practice using the application. Include instructions on how to use the app.
Revision History: Melanie Brady – 9/18/2020 – Created

3.5 Documentation Requirements

No: 19
Statement: All advanced documentation will be available through GitHub.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: All documentation will be accessible through GitHub.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 20
Statement: The users will be able to click a “help me” mode for instructions on how to use the app!
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify with test users that the instructions are easy to understand.
Revision History: Melanie Brady – 9/18/2020 – Created

3.6 Data Requirements

- <Describe any data calculations: what formula will be used? to what degree of precision must the calculations be made? >
- <Describe any retained data requirements: exactly what must be retained?
- ...>

No: 21
Statement: Only simple math will be needed for this program.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify calculations.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 22
Statement: The program will use low precision due to solely using integers for calculation.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify correct data is being stored.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 23
Statement: The program will most retain strings in the form of user data. The only numbers will be related to time stamps, userIDs, and time zones.
Source: None
Dependency: None

Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify calculations.
Revision History: Melanie Brady – 9/18/2020 – Created

3.7 Resource Requirements

No: 24
Statement: The program will be made by one developer.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: One person will be developing this project.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 25
Statement: A website will be needed for accessibility.
Source: None
Dependency: Requirement 18
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Have a user interface accessible online.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 26
Statement: A database will be needed to maintain data accuracy.
Source: None
Dependency: Requirements 1 - 14
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify accurate data within the database.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 27
Statement: The application will be compatible with android devices.
Source: None
Dependency: Requirements 1 - 10
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify accurate data within the database.
Revision History: Melanie Brady – 9/18/2020 – Created

3.8 Security Requirements

No: 28
Statement: The access to the system will be controlled by the application and its users.
Source: None
Dependency: Requirement 1-3

Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Access will only be permitted to registered users.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 29
Statement: User data will be isolated from each other within the database.
Source: None
Dependency: Requirements 1-15
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Each user will have their own ID.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 30
Statement: The Operating System will handle keeping the program separate from other programs.
Source: None
Dependency: Requirements 1-3
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Successful running of the application.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 31
Statement: The system will back up daily in a separate directory.
Source: None
Dependency: Requirements 1-19 & 30
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Successful backing up
Revision History: Melanie Brady – 9/18/2020 – Created

No: 32
Statement: The system will be safe from fire, water, damage, theft due to be stored on a third party database.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Third party databases maintain safe servers.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 33
Statement: User profiles would be backed up daily and restored if needed.
Source: None
Dependency: None

Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Test data losses to verify all data is restored.
Revision History: Melanie Brady – 9/18/2020 – Created

3.9 Quality Assurance Requirements

No: 34
Statement: Quality assurance test to make sure the application meets reliability, availability, maintainability, security, and portability standards.
Source: None
Dependency: Requirements 1 - 33
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Test thoroughly to verify there are no bugs.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 35
Statement: The application will backup data so that no data is lost.
Source: None
Dependency: Requirements 1 - 22
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Remove data and test to see if it copies over recovered data properly.

Revision History: Melanie Brady – 9/18/2020 – Created

No: 36
Statement: The application will have continuous uptime and will restart if any updates are needed.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Verify any local files needed to be saved are locally saved.
Revision History: Melanie Brady – 9/18/2020 – Created

No: 37
Statement: The application will detect errors and update the user as needed.
Source: None
Dependency: None
Conflicts: None
Supporting Materials: Required for customer requests.
Evaluation Method: Check for errors, log errors, maintain continuous uptime with python pulse checker
Revision History: Melanie Brady – 9/18/2020 – Created

Section 4: Supporting Material