**Elementary School Pickup Services System**

# **SOFTWARE TESTING**

# **AND**

# **QUALITY ASSURANCE**

# **DOCUMENT**

# 

# 

# 

# 

# **Team 3**

# **RoNeil Boyce**

# **William Curtis**

# **Elvis Mack**

# 

# **Software Engineering CSCI 5530,**

# **Section A**

# **Professor Andrew Allen**

# 

# **Abstract**

Many schools from K-12 have a car rider line for their students. The difference is not the grade level but how fast the students will be able to leave the school from the car rider line. With better technology we are able to improve the arrival and departure time of car rider students. By having more free time this means everyone will be able to spend their extra time with another task. The Elementary School Car Ride Pickup Line services is designed to allow staff to see new parents and their vehicles and belongs to the parent and the student. By using the Elementary School Car Ride Pickup Line services will result in a smoother transition from the end of the school day and to returning a parent to their child. This document will explain all the functionality and capabilities of the new system being implemented.

# 

# **TABLE OF CONTENTS**

[**Chapter 1**](#_g7kg4pokw9kf) **4**

[1.0 Introduction](#_985xe48gzc95) 4

[1.1 Purpose of System](#_4nlg6d7n1cl1) 4

[1.2 Functional and Nonfunctional Requirements](#_8fjtn8imdths) 4

[1.3 Definitions, acronyms, and abbreviations](#_mvkub3n95bmh) 4

[1.4 Overall Testing Approach](#_osk1uykhaja6) 4

[1.4.1 Unit Testing](#_orbnkmkv6rx8) 4

[1.4.2 Subsystem Testing](#_2mw7vwz24xb3) 4

[1.4.3 System Testing](#_l1nlb72g5jfo) 5

[1.5 Overview](#_kgp80ese0b3t) 5

[**Chapter 2**](#_u0s10mm9472x) **6**

[2.1 Organization](#_5r78ygp6y2fv) 6

[2.1.1 Team Lead](#_5l53s0s7ekqc) 6

[2.1.2 Testers](#_5ihyxcdidu3u) 6

[2.1.3 Minute Taker](#_jgzswurx3cav) 6

[2.1.4 Documentation](#_1o76alzh4qvw) 6

[2.2 Test Reference Items](#_elmxceiqiiwi) 6

[2.3 Tested Features](#_asvklne70pc7) 7

[2.4 Features Not Tested](#_gdv72prn6cx8) 7

[**Chapter 3**](#_5ylvmfx2mzi) **8**

[3.1 Unit Test Cases](#_1wi0h9755sem) 8

[3.1.1 Test Identification and Objective (Summary)](#_7n0a118e232m) 8

[3.1.2 Test Cases](#_nniw6lgmza34) 8

[**Subsystem Testing**](#_yh9cq97stxuz) **16**

[4.1 Subsystem Test Cases](#_xihyy9s00acz) 16

[4.1.1 Test Identification and Objective](#_ku54ch10rcgi) 16

[4.1.2 Test Criteria and Procedures](#_7525s9afihoo) 16

[4.1.3 Test Cases](#_nx75taxbj49t) 16

[com.esps.dashboards (Username: em03834@georgiasouthern.edu)](#_njhc40tngbq0) 16

[4.2 Actual Test Results](#_6794gwpu7otf) 17

[Test](#_tvyr8yn1rmrv) 17

[**Chapter 5**](#_hpxnpxvbq11h) **18**

[5.1 System Test Cases](#_3fwpf7bdk06f) 18

[5.1.1 Test Identification and Objective (Summary)](#_3fwpf7bdk06f) 18

[5.1.2 Test Criteria and Procedures](#_3fwpf7bdk06f) 18

[5.1.3 Test Cases](#_mns7s84hq5c8) 18

[5.2 Actual Test Results](#_wea35q75h53d) 19

[**Chapter 6**](#_keorqj81xop5) **21**

[**Chapter 7**](#_tvx56mcf2j0s) **22**

[**Chapter 8**](#_xxyrr7gixpk8) **23**

[8.1 Appendix A – Test Schedule (Pert chart) 23](#_71go863b4wx2)

[8.2 Appendix B – Use Case Diagram](#_g1cf6wxy8m2t) 24

[8.3 Appendix C – Use Cases 24](#_g1cf6wxy8m2t)

[8.4 Appendix D – Diary of Meeting and Tasks](#_jvzugl1y1wpj) 34

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# Chapter 1 Introduction

## **1.0 Introduction**

The Elementary School Pickup Line system is constructed to provide an efficient experience for car riders and for the staff. In the following sections we will discuss how this is possible and provide many examples and descriptions for this system. Chapter 1 will provide a brief and simplified outline of how this is done, the software requirements for the system, the definitions to help readers understand the terminology. Overall Chapter 1 will help you understand how this is feasibility possible.

## **1.1 Purpose of System**

The elementary School Car Ride Pickup Line originally searched for various cars to find the kids' parents. Once the outside staff has found the parent they must return to the inside staff to notify them of the specific child to meet their parent. Now the purpose of our system is to make this process more smoother for both the parents and the staff members. Making the process smoother means that there will be more time spent for a variety of tasks among everyone who is involved.

## **1.2 Functional and Nonfunctional Requirements**

|  |  |
| --- | --- |
| FR#0 | System includes login functionality |
| FR#1 | Admin can add or remove any Parent or Staff account |
| FR#2 | Admin can edit any Vehicle or Student information |
| FR#3 | Admin can send alerts to Parent and Staff accounts |
| FR#4 | Admin can verify additional relatives and their Vehicles |
| FR#5 | Admin can edit the calendar for sending daily alerts |
| FR#6 | Admin can send Emergency Shutdown alerts |
| FR#7 | Staff can edit the vehicles in the list |
| FR#8 | Staff views the list updates in real time |
| FR#9 | Staff can confirm a vehicle by inputting it’s make, model, color, and license plate # |
| FR#10 | Staff can view the daily PT for a unique student and how many students remain |
| FR#11 | Staff can mark a student “Absent” preventing the daily email from being sent to the parent |
| FR#12 | Staff can confirm a student has boarded the correct vehicle |
| FR#13 | Staff will receive an update for whether the Parent is Confirmed, Unconfirmed, Late, Absent |
| FR#14 | Staff can view vehicles in the car rider line |
| FR#15 | Parent can check into the car rider line |
| FR#16 | Parent can sign up for an account upon receiving an invitation |
| FR#17 | All Parent accounts receive an email 30 minutes before PT |
| FR#18 | Parents can respond Confirmed to the email. “Unconfirmed” will be the status until the response is received |
| FR#19 | Parent can notify the Staff that they are running late via the email alert |
| FR#20 | Parent can request to add multiple relatives/vehicles for one account |
| FR#21 | Staff/Admin are the only User’s permitted to view Parent/Student Data (Exception: Parent’s own User Data) |
| NFR#22 | Data is stored permanently |
| NFR#23 | Data will update in real time |
| NFR#24 | The system is only accessible on campus |
| NFR#25 | The program shall display the status of the parent in relation to picking up Students. |
| NFR#26 | The systems Date/Time will take into account leap year and daylight savings time |

## **1.3 Definitions, acronyms, and abbreviations**

|  |  |  |
| --- | --- | --- |
| **Term** | **Abbreviation** | **Description** |
| Elementary School Pickup Services System | ESPSS | Title of the given software |
| Elementary School Pickup Services | ESPS | Project name |
| Parent |  | User that provides data to Admin/Staff. |
| Pickup Time | PT | Time of student departure and event-based program. |
| Software Requirement Specification | SRS | Document to describe and visualize the entire functionality of a proposed system. |
| System Test |  | A set of cases that display whether the test has failed or has pass |
| SubSystem Test |  | focused on integrative, functional testing of several components of the system. |
| Unit Test Case |  | where the smallest part of individual **unit**/component (called **unit**) is tested to determine if they are fit for use |
| User |  | Admin, Staff, or Parent |
| Administrator | Admin | The boss that has the highest privilege. Must be the principal or whoever owns the application. |
| Application | App | A program that can be used on a mobile device and/or the computer. |
| Database | DB | Collection of all information monitored by the system. |
| Functional Requirement | FR | Areas of functionality that the system must have. |
| Model |  | An abstraction of a complex entity or process. |
| Nonfunctional Requirement | NFR | Constraints on the system. |
| Password |  | A secret phrase or word only known by the person who created it and the person who manages the database. |
| Software Requirements Specification | SRS | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. |
| Unified Modeling Language | UML | A general purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system. |
| User |  | Admin, Parent, Staff. Those who use the system. |
| Role |  | The responsibility of the given User. |
| Interface Event Handler |  | The connection between User interaction to the system. |
| Authentication |  | The correct identification of a user. |
| Feature |  | Functionality within the software that can be added. |
| Data |  | Information that is being collected and stored within the database. |
| Limitation |  | A defect that stop the creation of a feature or an understanding of the design. |
| Problem |  | A situation preventing something from being achieved. |

# 

## **1.4 Overall Testing Approach**

### **1.4.1 Unit Testing**

Unit testing is meant to test the classes and subsystems to see if they are able to communicate with each other.Our tests are done within Netbeans IDE and Firebase. Since Netbeans is able to hold our code and Firebase is to allow us to connect to our database.

### **1.4.2 Subsystem Testing**

The subsystem testing begins before the System testing. We decide to do the subsystem testing is focus on a specific area in our overall system. Any problems that we see in the subsystem testing should not appear in the overall system test, due to it being a part of its own smaller system. Our Elementary school Pickup Line will implement this style of testing for security purposes.

### **1.4.3 System Testing**

The system testing is when we are testing the entirety of the Elementary School Pickup line system. It will require each scenario to be broken down. The admin, staff and the parent features will be broken down with each option given.. Once all scenarios are successful and completed the Elementary School Pickup Line will be functionally ready to be a software product.

## **1.5 Overview**

Within the next chapters, the Elementary School Pickup Line will be explored in much greater detail. In Chapter 2 we will discuss the tested features and the non tested features. And if the tested features worked. In chapter 3 will have the Unit testing in much detail and sorted by case by case. The tested case are all the subsystem, System and Unit testing. Chapter 4 will contain the glossary to define the technical words that will be used in this document. In the last chapter which chapter 5, will contain the Appendix. The appendix is mainly for additional information and display of each diagram drawn out to represent the Elementary School Pickup Line service.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# **Chapter 2 Test Plans**

This section will describe the specific test plan for ESPSS. This chapter will be broken down into four sections: organization, test reference items, tested features, and features not tested. First, the distinct testing roles are depicted in the section below. The documents listed underneath are the references used to coordinate systems testing. Tested features will include which features of the application were tested out of the 25 functional and non-functional requirements. Features not tested will include which features of the application were not tested out of the 25 functional and non-functional requirements.

## **2.1 Organization**

#### **2.1.1 Team Lead**

William Curtis

#### **2.1.2 Testers**

William Curtis - Unit Testing

Elvis Mack - Subsystem Testing

RoNeil Boyce - System Testing

#### **2.1.3 Minute Taker**

RoNeil Boyce

#### **2.1.4 Documentation**

William Curtis

Elvis Mack

RoNeil Boyce

## **2.2 Test Reference Items**

● ESPSS Software Requirements Specification document

● ESPSS Design document

● ESPSS Meeting Minutes documents

## **2.3 Tested Features**

1. The system shall have login functionality

3. The Admin shall have the ability edit any Vehicle or Student Data

4. The Admin shall be able to send alerts to Parent accounts

5. The Admin shall be able to verify additional relatives and their Vehicles

6. The Admin shall have the ability to edit the calendar for sending automatic alerts

7. The Admin shall have the ability to send custom alerts

8. The Staff shall be able to edit the vehicles in the Car Rider Line list

10. The Staff shall be able to confirm a ride by inputting its make, model, year, and license tag #

11. The Staff will have access to data regarding Student’s pickup time to view remaining students

12. The Staff will be able to mark a student “absent”, removing them from the queue and alert list

13. The Staff shall have the ability to confirm a student has boarded the correct vehicle

15. The Staff shall have access to view the vehicles in the car rider line

16. The Parent shall be able to check in to the car rider line

17. The Parent will be allowed to sign up for an account upon receiving an invitation

19. The Parent shall be able to confirm their itinerary to pick up their Student

21. The Parent shall have the ability to send a “Running Late” alert to their Student’s corresponding Staff

23. Data shall be retrieved by connecting to an external database

## **2.4 Features Not Tested**

2. The Admin shall have the ability to add and remove users to/from and Parent/Staff accounts

9. The Staff shall receive updates in real time

14. The Staff shall receive updates on the pickup status of the corresponding Parent accounts

18. The Parent will receive automated alerts 30 minutes before pickup time

20. The Parent shall have the ability to request multiple associated vehicles to their account

22. The Staff/Admin shall be the only User with access to Student data

24. The system shall only be accessible on campus

25. The system will take into account leap year/DST

# **Chapter 3**

Unit Testing

## **3.1 Unit Test Cases**

#### **3.1.1 Test Identification and Objective (Summary)**

❖ **com.espss.dashboards.TestAdminProfile**

➢ Test cases are identified as ESPSS\_TAP\_XXX.

➢ The objective of testing this class is to ensure that all corresponding functionalities and privileges are accessible via the Admin profile.

❖ **com.espss.dashboards.TestStaffProfile**

➢ Test cases are identified as ESPSS\_TSP\_XXX.

➢ The objective of testing this class is to ensure that all corresponding functionalities and privileges are accessible via the Staff profile.

❖ **com.gsgas.dashboards.TestParentProfile**

➢ The cases are identified as ESPSS\_TPP\_XXX.

➢ The objective of testing this class is to ensure that all corresponding functionalities and privileges are accessible via the Parent profile.

❖ **com.gsgas.dashboards.TestSystem**

➢ The cases are identified as ESPSS\_TS\_XXX.

➢ The objective of testing this class is to ensure that all functionalities of the system security are performing nominally.

#### **3.1.2 Test Cases**

1. AdminProfile(UID: ESPSS\_TAP\_XXX)
   1. Test Case: ESPSS\_TAP\_001
      1. Method Signature: testEditProfile()
      2. Purpose: Test that Admin can add or remove any Staff or Parent account
      3. Precondition(s):

a) Instantiated User

* + 1. Input(s):

a) Admin.addAccount(User)

b) Admin.deleteAccount(User)

* + 1. Expected Output:

a) Not Null

* 1. Test Case: ESPSS\_TAP\_002
     1. Method Signature: testManageStudent()
     2. Purpose: Test that Admin can edit Student/Vehicle Data
     3. Precondition(s):

a) Instantiated Student/Vehicle

* + 1. Input(s):

a) Admin.manageStudent(student)

b) Admin.manageVehicle(vehicle)

* + 1. Expected Output:

a) Not Null

* 1. Test Case: ESPSS\_TAP\_003
     1. Method Signature: boolean testSendAlert(alert)
     2. Purpose: Test that Admin can send alerts to Parent accounts
     3. Precondition:

a) “Test”

* + 1. Input:

a) Admin.getAlertMessage()

* + 1. Expected Output:

a) True

* 1. Test Case: ESPSS\_TAP\_004
     1. Method Signature: boolean testRelativeApproval(relativeName),
     2. Purpose: Test that Admin can verify additional relatives and their vehicles
     3. Precondition:

a) Instantiated User object

* + 1. Input:

a) Admin.manageVehicle(vehicle)

* + 1. Expected Output

a) true/false

b) Not Null

* 1. Test Case: ESPSS\_TAP\_005
     1. Method Signature: testEditCalendar()
     2. Purpose: Test that Admin can edit the Alert calendar
     3. Precondition

a) Instantiated Data object

b) Instantiated Connection object

c) Instantiated Alert object

* + 1. Input

a) Admin.setNewCalendarAlert(Alert)

b) Admin.deleteCalendarAlert(Alert)

c) Admin.silenceCalendarAlert(Alert)

* + 1. Expected Output

a) true

b) true

c) true

* 1. Test Case: ESPSS\_TAP\_006
     1. Method Signature: boolean testSendAlert()
     2. Purpose: Test that Admin can send custom Alerts
     3. Precondition

a) Instantiated Alert object

* + 1. Input

a) “Test”

* + 1. Expected Output

a) true

1. StaffProfile(UID: ESPSS\_TSP\_XXX)
   1. Test Case: ESPSS\_TSP\_001
      1. Method Signature: testEditVehicle(vehicle, int position)
      2. Purpose: Test that Staff can edit cars in the Car Rider list.
      3. Precondition

a) Instantiated carLine[] object

b) Instantiated Vehicle object

c) Instantiated int

* + 1. Input

a) Staff.addVehicle(vehicle, position)

b) Staff.moveVehicle(vehicle, position)

* + 1. Expected Output

a) parameter “carLine” has at least 1 variable

* 1. Test Case: ESPSS\_TSP\_002
     1. Method Signature: bool testConfirmRide(vehicle)
     2. Purpose: Test that Staff can confirm Vehicles for pickup
     3. Precondition

a) Instantiated vehicle object

* + 1. Input

a) Staff.verifyRide(make, model, year, license#)

* + 1. Expected Output

a) true

* 1. Test Case: ESPSS\_TSP\_003
     1. Method Signature: testClassroomView()
     2. Purpose: Test that Staff can view the remaining students to be picked up.
     3. Precondition

a) Instantiated listOfStudents[] object

* + 1. Input

a) Staff.getRemainingStudent()

b) Staff.displayStudentData(Student)

* + 1. Expected Output

a) int

b) Not Null

* 1. Test Case: ESPSS\_TSP\_004
     1. Method Signature: testEditStudentStatus()
     2. Purpose: Test that Staff can mark students “Absent”
     3. Precondition

a) Instantiated Student object

* + 1. Input

a) Staff.setStudentStatus(Student, “Absent”)

* + 1. Expected Output

a) null

* 1. Test Case: ESPSS\_TSP\_005
     1. Method Signature: bool testConfirmPickup()
     2. Purpose: Test that Staff can confirm Student has boarded the correct vehicle.
     3. Precondition

a) Instantiated vehicle object

b) Instantiated student object

* + 1. Input

a) Staff.verifyPickup(vehicle, student)

* + 1. Expected Output

a) true

* 1. Test Case: ESPSS\_TSP\_006
     1. Method Signature: testCarLineView()
     2. Purpose: Test that Staff can view the entire Car Rider Line
     3. Precondition

a) Instantiated carLine[] object

* + 1. Input

a) Staff.viewCarLine()

* + 1. Expected Output

a) Not Null

1. ParentProfile(ESPSS\_TPP\_XXX)
   1. Test Case: ESPSS\_TPP\_001
      1. Method Signature: testParentCheckIn ()
      2. Purpose: Test that Parent can check in to Car Rider Line
      3. Precondition

a) Instantiated Parent object

* + 1. Input

a) Parent.checkIn(true)

* + 1. Expected Output

a) null

* 1. Test Case: ESPSS\_TPP\_002
     1. Method Signature: testParentRegister()
     2. Purpose: Test that Parent can sign up upon receiving invitation
     3. Precondition

a) Instantiated Parent object

* + 1. Input

a) Parent.createAccount(name, email, password, accessCode)

* + 1. Expected Output

a) not null

b) number of course sections == length of returned course sections list

* 1. Test Case: ESPSS\_TPP\_003
     1. Method Signature: testParentConfirm()
     2. Purpose: Test that Parent can confirm they have received an alert and are on the way.
     3. Precondition

a) Instantiated Parent object

* + 1. Input

a) Parent.confirmRide()

* + 1. Expected Output

a) null

* 1. Test Case: ESPSS\_TPP\_004
     1. Method Signature: testParentSendAlert()
     2. Purpose: Test that Parent can send a “Running Late” alert to corresponding staff.
     3. Precondition

a) Instantiated Parent

* + 1. Input

a) Parent.runningLate()

* + 1. Expected Output

a) Null

1. System (UID: ESPSS\_TS\_XXX)
   1. Test Case: ESPSS\_TS\_001
      1. Method Signature: testLogin()
      2. Purpose: Test that User can login with verified DB credentials
      3. Precondition

a) Instantiated Data object

b) “Username”

c) “Password”

* + 1. Input

a) Data.signIn(“Username”, “Password”)

* + 1. Expected Output

a) not null

* 1. Test Case: ESPSS\_TS\_002
     1. Method Signature: testDatabaseConnection()
     2. Purpose: Test that System can connect to the DB
     3. Precondition

a) Instantiated Data object

* + 1. Input

a) Data.connectToDatabase()

* + 1. Expected Output

a) Not Null

**3.2 Test Case Results**

Admin Profile - Coverage 80%

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity | Summary |
| ESPSS\_TAP\_001 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TAP\_002 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TAP\_003 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TAP\_004 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TAP\_005 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TAP\_006 | 6/9/2020 | William Curtis | Pass | N/A | N/A |

Staff Profile - Coverage 85%

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Date | Tester | Pass/Fail | Severity | Summary |
| ESPSS\_TSP\_001 | 6/9/2020 | William Curtis | Pass | N/A | N/A05 |
| ESPSS\_TSP\_002 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TSP\_003 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TSP\_004 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TSP\_005 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TSP\_006 | 6/9/2020 | William Curtis | Pass | N/A | N/A |

Parent Profile - Coverage 68%

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity | Summary |
| ESPSS\_TPP\_001 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TPP\_002 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TPP\_003 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TPP\_004 | 6/9/2020 | William Curtis | Pass | N/A | N/A |

System - Coverage 74%

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case ID | Date Tested | Tester | Pass/Fail | Severity | Summary |
| ESPSS\_TS\_001 | 6/9/2020 | William Curtis | Pass | N/A | N/A |
| ESPSS\_TS\_001 | 6/9/2020 | William Curtis | Fail | High | Could not join DB connection with front end model. |

# Chapter 4 Subsystem Testing

## **4.1 Subsystem Test Cases**

### **4.1.1 Test Identification and Objective**

Subsystem testing is to be conducted on the security subsystem, including the corresponding requirements relating to the privacy of all users. The subsystem testing focused on integrative, functional testing of several components of the system.

### **4.1.2 Test Criteria and Procedures**

Tests are conducted through pyTest and require access to the subsystem’s interfaces used by other subsystems. The test coverage is curated by Coverage.py.

### **4.1.3 Test Cases**

### **com.esps.dashboards (Username: Mack32)**

* 1. SS\_Dash\_01
     1. **Purpose:** Redirect user from Admin Dashboard to Staff Dashboard
     2. **Preconditions:**
     3. Admin is already logged in and is on the account page.
     4. **Inputs:** User clicks “Change Roles > Staff”
     5. **Expected Output:** Session attribute for “username” should be set to Mack32. Request attribute for the “program” should contain all objects in the Admin’s Dashboard. Staff Dashboard is displayed to the user.
     6. **Post Conditions:** The dashboard is now showing the staff dashboard to the user.
  2. SS\_Dash\_02
     1. **Purpose:** Generate a report for staff
     2. **Preconditions:** User adds Chrysler 300c to the report and checks vehicle information for type, all information types referring to the vehicle, and selects the radio button for column chart type. The user clicks the Generate Report button.
     3. **Inputs:** vehicleName-report = Chrysler 300c, parentName-report = Marie Jones, typeofInfo-report = All
     4. **Expected Output:** Name = {Marie Jones}, Maker = {Chrysler}, Model = {300c}, plateNumber = {GUS2503},Color = {Gray}
     5. **Post Conditions:** The user reviews the report for pickup.
  3. SS\_Dash\_03
     1. **Purpose:** Filter Status on Student Pick Up List
     2. **Preconditions:** User is on the Student Pick Up page
     3. **Inputs:** onStatus = Here,
     4. **Expected Output:** program.Status size = 3
     5. **Post Conditions:** The user has an updated list of parent pickups with relevant information for their arrival.

## **4.2 Actual Test Results**

### **Test**

* 1. **SS\_Dash\_01** - Redirect user from Admin Dashboard to Staff Dashboard
     1. **Date:** 05/27/20
     2. **Version:** 1.01
     3. **Ran by:** Admin 3
     4. **Code Coverage:** 100%
  2. **SS\_Dash\_02** - Generate a report for staff
     1. **Date:** 05/31/20
     2. **Version:** 1.03
     3. **Ran by:** Staff 2
     4. **Code Coverage:** 100%
  3. **SS\_Dash\_03** - Filter Status on Student Pick Up List
     1. **Date:** 06/05/20
     2. **Version:** 1.03
     3. **Ran by:** Staff 2
     4. **Code Coverage:** 100%







# 

# Chapter 5 System Test Cases

## 5.1 System Test Cases

### 5.1.1 Test Identification and Objective (Summary)

The system tests are conducted on the entire system using the three use case scenarios in this section.

## 

### 5.1.2 Test Criteria and Procedures

The tests are conducted manually and represent how the end users would interact with the system upon official release.

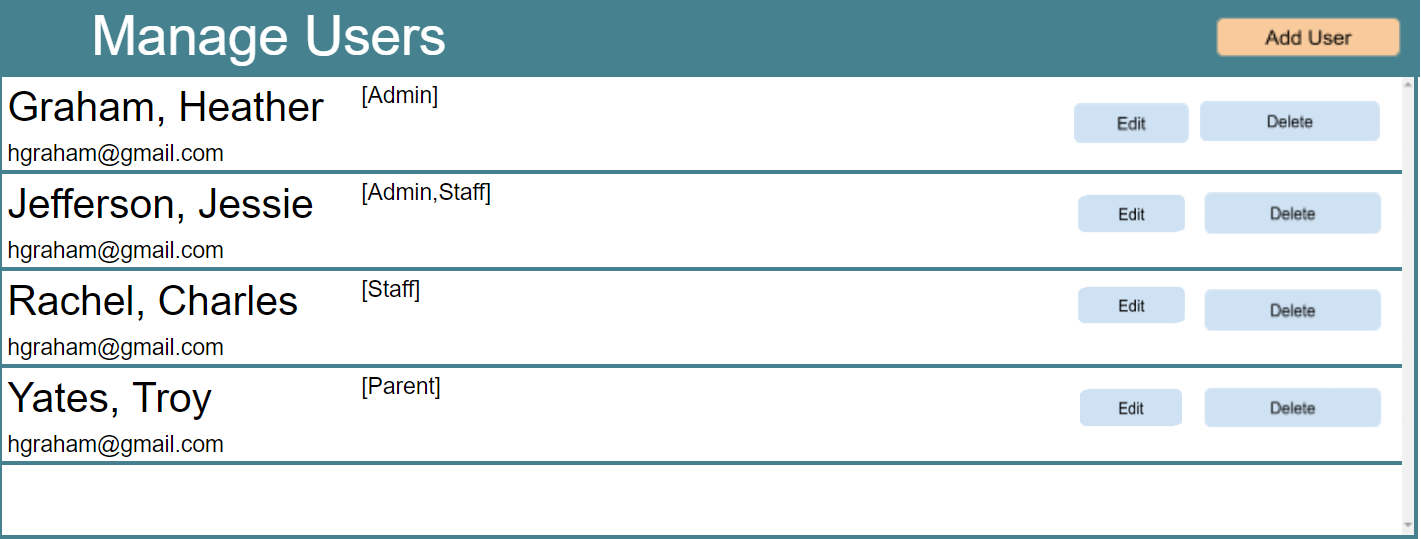
### 5.1.3 Test Cases

1. **System (UID: Elementary School Admin, adding and removing accounts)**
   1. **Test Case: ES\_ADMIN\_001**
      1. Purpose: Ensure that the system can add a new user into the system, remove user from the system and assign its roles.
      2. Precondition(s): User creating a new user is validated as an administrator.
      3. Input(s): User name, User username, User password, User email, User roles
      4. Expected Output: user is created, email is sent to new user with generated password, or the user is terminated and erased from the database.
      5. Postcondition(s):
         1. New user is added to system with generated password
         2. Email is sent to new user with their login info
         3. Page is refreshed with new user in the list of users
2. **System (UID: Elementary School Admin, adding and removing programs)**
   1. **Test Case: ES\_ADMIN\_001**
      1. Purpose: Ensure that the system can add, or remove new programs into the system.
      2. Precondition(s): User creating a new user is validated as an administrator.
      3. Input(s): Program name, program functionality
      4. Expected Output: Program name and functionality is implemented into the system
      5. Postcondition(s):
         1. New program is added to system with new functionality
         2. Program is viewable to those who have access to it
         3. Page is refreshed with new Program options.
3. **System (UID: Parent, request for multiple vehicles)**
   1. **Test Case: ES\_Parent\_001**
      1. Purpose: Ensure that the system can handle accounts with multiple vehicles.
      2. Precondition(s): User requesting for an added vehicle must be a parent.
      3. Input(s): User vehicle model, vehicle make, and vehicle vin number.
      4. Expected Output: user request has been delivered to the admin.
      5. Postcondition(s):
         1. The admin accepts the request
         2. An email is sent to the Parent stating the request has been approve
         3. Page is refreshed with the new list of vehicles

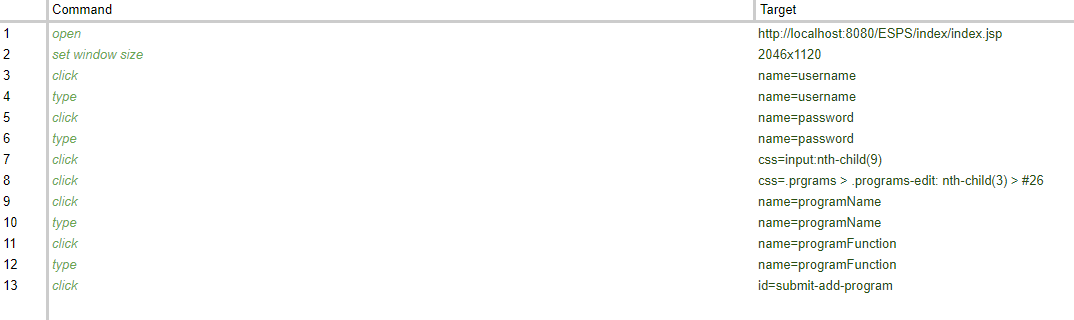
## 5.2 Actual Test Results

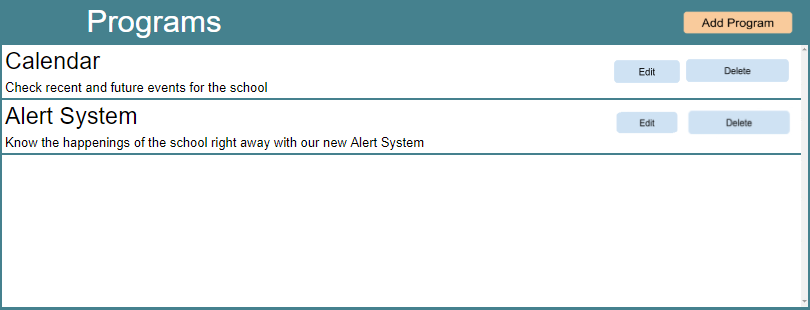
1. **System (UID: Elementary School Admin, adding and removing accounts)**
   1. **Elementary School Admin,**
      1. **Date:** 6/1/2020
      2. **a)** **Result(s):** Expected Output returned
      3. **b)** **Version:** 1.01
      4. **c)** **Ran by:** William Curtis



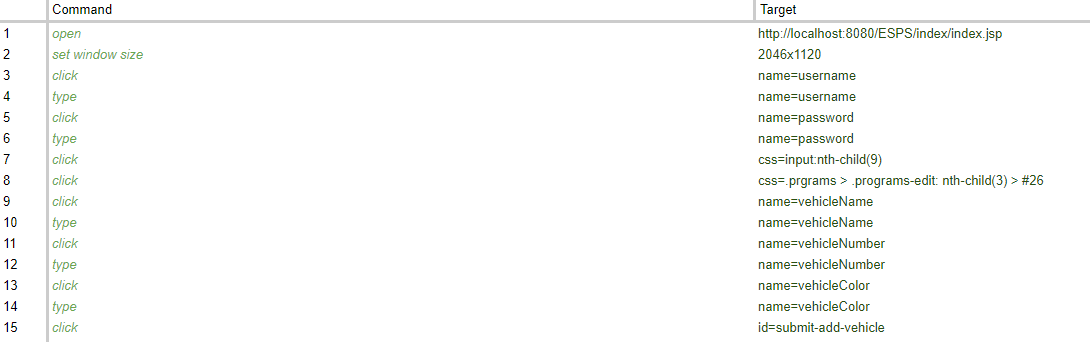


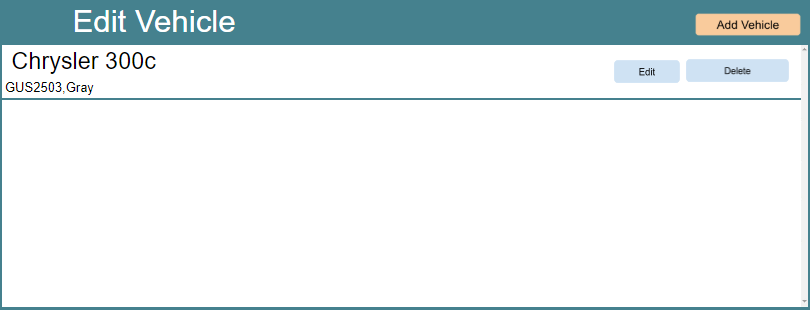
1. **System (UID: Elementary School Admin, adding and removing programs)**
   1. **Elementary School Admin,**
      1. **Date:** 6/5/2020
      2. **a)** **Result(s):** Expected Output returned
      3. **b)** **Version:** 1.01
      4. **c)** **Ran by:** RoNeil Boyce





1. **System (UID: Parent, request for multiple vehicles)**
   1. **Parent**
      1. **Date:** 6/6/2020
      2. **a)** **Result(s):** Expected Output returned
      3. **b)** **Version:** 1.01
      4. **c)** **Ran by:** Elvis Mack





## 

# 

# 

# Chapter 6 Glossary

|  |  |  |
| --- | --- | --- |
| **Term** | **Abbreviation** | **Description** |
| Elementary School Pickup Services System | ESPSS | Title of the given software |
| Elementary School Pickup Services | ESPS | Project name |
| Parent |  | User that provides data to Admin/Staff. |
| Pickup Time | PT | Time of student departure and event-based program. |
| Software Requirement Specification | SRS | Document to describe and visualize the entire functionality of a proposed system. |
| System Test |  | A set of cases that display whether the test has failed or has pass |
| SubSystem Test |  | focused on integrative, functional testing of several components of the system. |
| Unit Test Case |  | where the smallest part of individual **unit**/component (called **unit**) is tested to determine if they are fit for use |
| User |  | Admin, Staff, or Parent |
| Administrator | Admin | The boss that has the highest privilege. Must be the principal or whoever owns the application. |
| Application | App | A program that can be used on a mobile device and/or the computer. |
| Database | DB | Collection of all information monitored by the system. |
| Functional Requirement | FR | Areas of functionality that the system must have. |
| Model |  | An abstraction of a complex entity or process. |
| Nonfunctional Requirement | NFR | Constraints on the system. |
| Password |  | A secret phrase or word only known by the person who created it and the person who manages the database. |
| Software Requirements Specification | SRS | A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. |
| Stakeholder |  | Any person with an interest in the project who is not a developer. |
| Unified Modeling Language | UML | A general purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system. |
| User |  | Admin, Parent, Staff. Those who use the system. |
| Role |  | The responsibility of the given User. |
| Interface Event Handler |  | The connection between User interaction to the system. |
| Authentication |  | The correct identification of a user. |
| Feature |  | Functionality within the software that can be added. |
| Data |  | Information that is being collected and stored within the database. |
| Limitation |  | A defect that stop the creation of a feature or an understanding of the design. |
| Problem |  | A situation preventing something from being achieved. |
| Event Driven Architecture |  | Provides loose coupling and high cohesion to maintain a distributed environment. |
| Three Tier Architecture |  | Consisting of a presentation layer, a business layer, and a database layer. |
| Presentation Layer |  | Responsible for presenting the look and design for the user to handle when using the application. |
| Business Layer |  | The responsibility of executing the algorithms and connecting the presentation and Database layer. |
| Database Layer |  | The database layer will store all the data used in the system. |
| Coupling |  | Strength of connection between modules. |
| Cohesion |  | The glue that keeps modules together. |

# 

# 

# Chapter 7 Appendix

## 8.1 Appendix A – Test Schedule (Pert chart) Expected time = (Optimistic +4\*Normal + Pessimistic)/6

## 

## 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Activity | Predecessor | Optimistic time | normal time | pessimistic time | expected days |
| unit testing | none | 4 | 7 | 10 | 7 |
| Subsystem Testing | unit testing | 6 | 10 | 14 | 10 |
| System Testing | Subsystem testing | 6 | 10 | 14 | 10 |

## 

## 

## 8.2 Appendix B – Use Case Diagram

## 8.3 Appendix C – Use Cases **Use Case ID:** UC1 – Add or Remove Users From Elementary School Pickup Service

**Scenario:**

**Actor:** Admin, System

**Pre-conditions:**

Admin already has accessed the system and has been authenticated as an Admin.

**Description:**

1. The Admin chooses to assign the user’s role

2. The system displays the screen to edit roles and privileges.

3. The Admin selects a user and assigned a role and privileges.

4. The system asks the Admin to confirm role and privilege changes.

5. The Admin confirms role and privilege changes.

**Post-conditions:** The specified users are added or removed from their respective Elementary School Pickup Service.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None.

**Criticality:** High. Usefulness of application is dependent on the ability to execute instance models.

**Risk:** Low

**Constraints:**

1. The user interface should explicitly present the user only the available users that could be added to or removed from the Elementary School Pickup Service.

**Use Case ID:** UC2 – Add or Remove Programs From Elementary School Pickup Service

**Scenario:**

**Actor:** Admin, System

**Pre-conditions:**

Admin already has accessed the system and has been authenticated as an Admin.

**Description:**

1. The Admin chooses to add a program.

2. The system displays the screen to assign a functionality.

3. The Admin selects a functionality of the program.

4. The system asks the Admin to confirm program.

5. The Admin confirms program.

**Post-conditions:** The specified programs are added or removed from their respective Elementary School Pickup Service.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None.

**Criticality:** High. Usefulness of application is dependent on the ability to execute instance models.

**Risk:** Low

**Constraints:**

1. The user interface should explicitly present the user only the available users that could be added to or removed from the Elementary School Pickup Service.

**Use Case ID:** UC3 – Manages Alert for Elementary School Pickup Service.

**Scenario:**

**Actor:** Admin, System

**Pre-conditions:**

Admin already has accessed the system and has been authenticated as a Admin.

**Description:**

1. The Admin chooses when to declare an alert, and the reasoning.

2. The system displays the date, time, and reasoning.

3. The Admin selects the time and date and the reasoning for the alert.

4. The system asks the Admin to confirm the alert.

5. The Admin confirms the alert.

**Post-conditions:** All users will receive an alert from their Elementary School Pickup Service.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None.

**Criticality:** High. Usefulness of application is dependent on the ability to execute instance models.

**Risk:** Low

**Constraints:**

1. The user interface should display only the upcoming alert within ten seconds once issued by the Admin..

**Use Case ID:** UC4 – View/Edit Car Rider Line

**Scenario:**

**Actor:** Staff, System

**Pre-conditions:**

Staff has already accessed the system and has been authenticated as an Staff.

**Description:**

1. The Staff can open “Pick-up Line” page to view the current list of cars waiting.

2. The system displays a Confirm/ Delete option when a Staff selects a vehicle from the list.

3. The system updates by changing the status/color of confirmed cars from red (unconfirmed) to green (confirmed).

**Post-conditions:** The system updates the list in real time allowing for other Staff to input data accordingly.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** UC5 - Edit/Display remaining Student

**Criticality:** Medium. Staff should be able to edit the list in real time, while the system will still function if an error occurs.

**Risk:** Low

**Constraints:**

1. The system will only show the staff the vehicles remaining in the pickup line.

**Use Case ID:** UC5- Edit/Display remaining Student

**Scenario:**

**Actor:** Staff, System

**Pre-conditions:**

Staff has already accessed the system and has been authenticated as an Staff. Students who have been marked “Absent” have been updated in the system by Admin.

**Description:**

1. The Staff can open “My Classroom” page to see the list of students linked to their classroom that have not been picked up yet.

2. The staff can select a Student from the list, highlighting their information.

3. The Staff can select Release, to release the corresponding student to their vehicle after that vehicle has been confirmed by the other staff.

4. A counter displaying the number of students remaining will count down as students are released.

**Post-conditions:** The system will update the Admin whenever the list is complete or if there are outlying students still awaiting rides.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None

**Criticality:** High, as this is one of the primary features of the application.

**Risk:** Low

**Constraints:**

1. The system will only show the Staff information about the students in their own classrooms.

**Use Case ID:** UC6- Parent Create Account

**Scenario:**

**Actor:** Parent, System

**Pre-conditions:**

Admin already has accessed the system and has been authenticated as an Admin.

**Description:**

1. The Parent can open a “Create Account” Window

2. The System will prompt the user for specific information for creating an account

3. The Parent will be prompted to submit their information (Name, Email, Password, Number of Children, Children’s names, Vehicle Description)

4. The System will ask if the Parent is sure that the information is correct.

5. The Parent can click “Submit”, sending the information to Admin for verification.

**Post-conditions:** The system will update the Admin whenever the submission is complete.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None

**Criticality:** Low, the functionality of the app relies on some Parent data.

**Risk:** Low

**Constraints:** None

**Use Case ID:** UC7- Request Multiple Vehicles

**Scenario:**

**Actor:** Parent, System

**Pre-conditions:**

Parent has already accessed the system and has been authenticated as a Parent.

**Description:**

1. The Parent can open a “Add Vehicle” Window

2. The System presents the Add vehicle menu and request the specific information

3. The Parent will be prompted to submit their information (Name, Email, Password, Vehicle Description)

4. The System will verify that this is the correct information

5. The Parent will click “Yes” or “No” to verify that they did not enter any mistakes.

6. The System will send the request to the Admin for approval

**Post-conditions:** The system will update the Admin whenever the submission is complete.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None

**Criticality:** Low

**Risk:** Low

**Constraints:**

1. The system will recognize a maximum of 3 vehicles per account.

**Use Case ID:** UC8- Parent receives Alert

**Scenario:**

**Actor:** Parent, System

**Pre-conditions:**

Parent has already accessed the system and has been authenticated as a Parent.

**Description:**

1. The Parent will receive a scheduled alert 30 minutes before PT to notify them of their appointment.

2. The system will send a message to the Parent verifying their status.

3. The Parent’s receive the alert and confirm if they are “Running Late” or “Confirmed” for on time pickup.

4. The system will update the Parent’s status.

**Post-conditions:** The system will update the Staff whenever the Parent responds with their status.

**Alternative Courses of Action:**

1. The user may choose to close the application at any time.

**Concurrent Uses:** None.

**Related Use Cases:** None

**Criticality:** Low

**Risk:** Low

**Constraints:**

1. The system will only send these alerts to Parents of Students who have not been labeled “Absent”.

## 

## 8.4 Appendix D – Diary of Meeting and Tasks

|  |  |
| --- | --- |
| Date | May 20, 2020 |
| Time | 1:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis |
| Task | Greeting and starting SRS Document |
| Major Topics | If we are getting a 3rd member  What platform should we use  What is our strength. |

|  |  |
| --- | --- |
| Date | May 21, 2020 |
| Time | 11:00 A.M to 12:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis |
| Task | Continuation of SRS |
| Major Topics |  |

|  |  |
| --- | --- |
| Date | May 22, 2020 |
| Time | 11:00 A.M to 12:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis, Elvis Mack |
| Task | Greeting the new member and catching him up on what happen |
| Major Topics | Figuring out his weakness and what to do for the SRS document |

|  |  |
| --- | --- |
| Date | May 24, 2020 |
| Time | 1:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis. Elvis Mack |
| Task | Review SRS Document |
| Major Topics | Deciding on the approach with the project  Deciding if we are implementing the daycare or Athletic survey  Deciding on if we should do website or mobile application |

|  |  |
| --- | --- |
| Date | May 27, 2020 |
| Time | 11:00 A.M to 1:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Starting The Software Design Documentation |
| Major Topics | Breaking up the task evenly  Deciding what will be needed for the Design |

|  |  |
| --- | --- |
| Date | May 28, 2020 |
| Time | 11:00 A.M to 12:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Status Update on individual parts with the Design implementation |
| Major Topics | When should this be completed |

|  |  |
| --- | --- |
| Date | June 2, 2020 |
| Time | 12:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for Sprint 1 |
| Major Topics | What platform will we officially use  Who will do which end of the software  Status updates every 3 hours. |

|  |  |
| --- | --- |
| Date | June 3, 2020 |
| Time | 11:00 A.M to 12:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for Sprint 1 |
| Major Topics | What platform will we officially use  Who will do which end of the software  Status updates every 3 hours. |

|  |  |
| --- | --- |
| Date | June 4, 2020 |
| Time | 12:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for Sprint 1 |
| Major Topics | What requirements should we focus on? |

|  |  |
| --- | --- |
| Date | June 7, 2020 |
| Time | 12:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for Sprint 2 |
| Major Topics | What will we pivot to if we can’t connect with Java. |

|  |  |
| --- | --- |
| Date | June 8, 2020 |
| Time | 12:00 P.M to 1:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for STQA Document |
| Major Topics | Who will do which set of Chapters for the STQA  The group personal due date |

|  |  |
| --- | --- |
| Date | June 9, 2020 |
| Time | 12:00 P.M to 1:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Connecting the Database |
| Major Topics | Research for database connection |

|  |  |
| --- | --- |
| Date | June 10, 2020 |
| Time | 12:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Final checks for STQA Document |
| Major Topics | What platform will we officially use  Who will do which end of the software  Status updates every 3 hours. |

|  |  |
| --- | --- |
| Date | June 11, 2020 |
| Time | 12:00 P.M to 2:00 P.M |
| Location | Discord |
| Attendance | RoNeil Boyce, Liam Curtis Elvis Mack |
| Task | Preparation for Sprint 2 |
| Major Topics | Connecting to Pyrebase |