README

Aaron Hamilton

Aaron.Hamilton@mavs.uta.edu

Joshua Clark

Joshua.Clark@mavs.uta.edu

(214) 784 – 0677

Robert Aron

Robert.Aron@mavs.uta.edu

(214) 448 – 2263

Ji-Yeon Jeong

Ji-Yeon.Jeong@mavs.uta.edu

(817) 350 – 7504

Ian Strnad

Ian.Strnad@mavs.uta.edu

Project:	MSS (Maverick Scheduling Service)
Team No.:	Omicron
Class:	CSE 3310; Spring 2015
Module:	System Requirements Analysis (SRA)
Deliverable:	SRA Document

Version: [1.1] **Date: [03/31/2015]**

Contributors:

Aaron Hamilton
Ian Strnad
Ji-Yeon Jeong
Joshua Clark
Robert Aron

Revision History

Version number	*Date*	*Originator*	*Reason for change*	*High level description of changes*
1.0	03/19/2015	Aaron Hamilton Ian Strnad Ji-Yeon Jeong Joshua Clark Robert Aron	Initial draft	
1.1	3/31/15	Aaron Hamilton Ian Strnad Ji-Yeon Jeong Joshua Clark Robert Aron	Update	Added UML diagrams; Modified requirements; Updated Objectives

TABLE OF CONTENTS

1. Introduction and Project Overview

MSS (Maverick Scheduling Service) is a collaborative effort by Team Omicron to help students of UTA plan a complex schedule with minimal effort.

Students will be able to plan a school schedule and tailor it to their personal preferences. They will have the ability to block out days and/or time blocks to help facilitate their time management.

2. Objectives

2.1 BUSINESS OBJECTIVES

The following is a list of business objectives:

Objective 1: Account Management

1.1 Member Registration: All members must provide the following information prior to using the Application:

- Account Username
- Account Password
- First Name, Last Name
- UTA Student ID
- E-mail address (Optional)

1.2 Edit Account: A user can edit his/her information if he/she needs to.

1.3 Delete Account: A user can delete his/her account including all information.

Objective 2: Login functionality: All members must login to the system with a Username/Password that was established during member registration stage.

Objective 3: Logout functionality: All members can log out from the application.

Objective 4: Add/Remove Class: Classes should be added to a list and removed from a list as the user intends.

4.1 Add Class: A user can add classes into a list for schedule management.

4.2 Remove Class: A user can remove classes from the list for schedule management.

Objective 5: Schedule Management

5.1 Create Schedule: With classes in a list, a user can create all possible available schedules.

5.2 Select Schedule: A user can select a schedule from the possible verified schedules produced.

5.3 Verify Schedule: The application will verify schedule to show whether schedule is valid or not.

5.4 Store Schedule: A user can store schedules into his/her data storage.

5.5 Remove Schedule: A user can delete schedules from his/her data storage.

2.2 SYSTEM OBJECTIVES

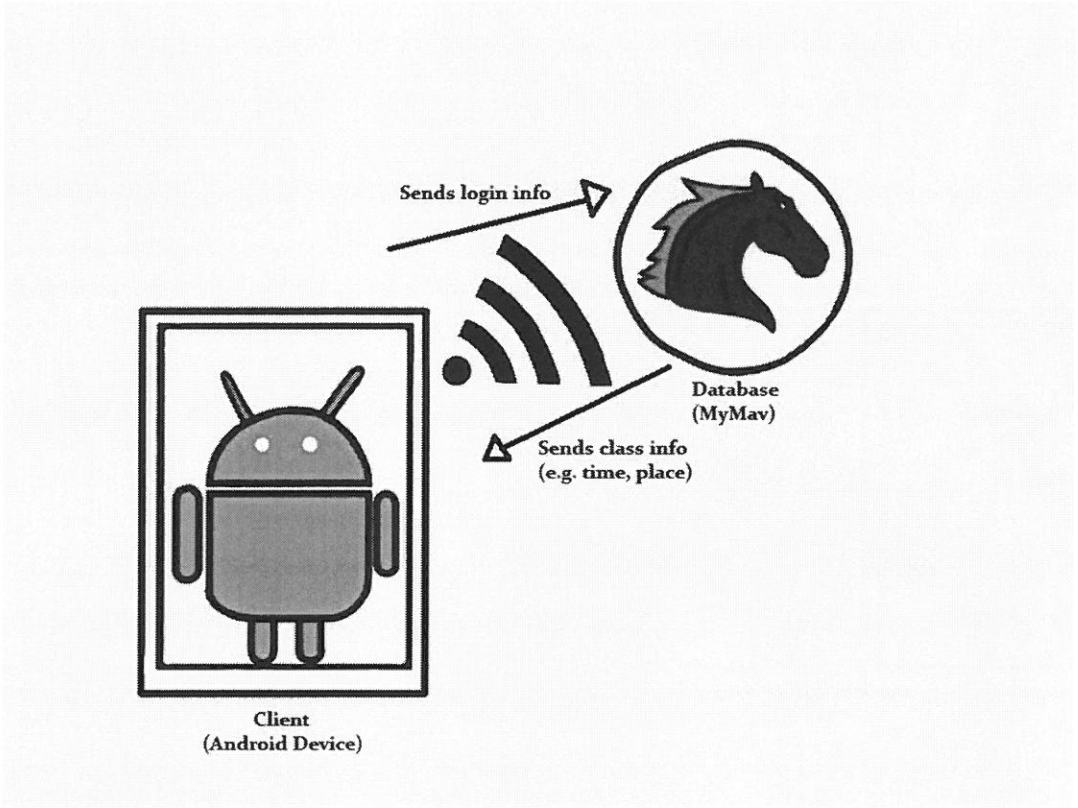
The following is a list of system objectives:

Objective 1: System will be web based.

Objective 2: Operating system will be Android with at least version 4.1.0 (JellyBean).

Objective 3: Online server will be updateable with new class lists.

3. Project Context Diagram



4. Systems Requirements

4.1 "Add Account" Requirements

The user will enter their desired account credentials, and assuming they meet the requirements, the application will add a new user to the database with those credentials. If a user already exists in the system with the same username, the application shall prompt the user to re-enter their data with new credentials.

For this to function properly, the user must be a current UTA student.

4.2 "Edit / Delete Account" Requirements

If the user selects to edit their account, the application shall move to the edit account screen. If the user chooses to edit their password, the application shall prompt the user to re-enter their password to confirm their identity. If the password entered matches the password in the database, the application shall prompt the user to enter a new password and new password confirmation. If the new password entered matches the requirements, the password shall be changed in the database. If the user chooses to delete their account, the application shall warn the user and prompt for a confirmation. If the action is confirmed by the user, the application shall delete that account from the database.

For this to function properly, the user must be logged in to the application.

4.3 "Login" Requirements

If the user chooses to log in, assuming data has already been entered into the username/password fields, the application shall check the entered data with the database for verification. If the username matches a username already in the database, the application shall check the password tied to it. If the password matches, the application shall log the user in and move to the main screen. If the data checks fail at any point, the user shall be notified and prompted to re-enter the data.

For this to function properly, the user must have an account already registered in the database.

4.4 "Logout" Requirements

If the user chooses to log out of the system, the application shall end the current session and return to the login screen.

For this to function properly, the user must be logged in to the system.

4.5 "Add Class" Requirements

When the user types a class into the application, assuming valid data was entered, the application shall search the database on the MyMav system. If any open classes are found matching the data entered, the application shall add all available classes to the list of possible choices for that class when being entered into schedule creation. If no open classes are found, the application shall return that no open classes were found, and add nothing.

For this to function properly, the user must first be logged in, the application must have access to the internet, and it must have access to MyMav.

4.6 "Remove Class" Requirements

When the user selects to remove a class from their schedule, the application shall prompt the user to confirm the action. If the user confirms the action, the application shall remove the class from the schedule. If the user cancels the action at this prompt, the application shall not prompt the user and will not remove anything.

For this to function properly, the user must first be logged in, and the application must have at least one class already added to the schedule.

4.7 "Create Schedule" Requirements

If the user chooses to create a schedule, the application shall sort the classes that it retrieved from the MyMav database into all possible combinations and display the choices for the user to choose. If no combinations can be made, the application shall notify the user that it failed and return to the create schedule screen.

For this to function properly the user must have at least one class added to their class choices.

4.8 "Select Schedule" Requirements

After the schedules with the specified classes are generated, the application shall display the possible schedules on screen. The user will then have the ability to choose one of the schedules to be their schedule. After the user selects a schedule, the option will be given to save it (see Store Schedule). If the user either saves a schedule or cancels, the application will take them to the create schedule page.

4.9 "Verify Schedule" Requirements

If the user chooses to verify the schedule they have stored, the application shall connect to the MyMav database and search through the classes the user has selected. If the selected courses and sections that are in the stored schedule are no longer available, the application shall notify the user that the schedule they have is no longer available. The user shall then be given the option to delete that schedule, create a new one with the same classes, or keep the schedule.

For this to function properly, the user must be logged in and have a schedule already stored.

4.10 "Store Schedule" Requirements

The application shall display a 'store' button on screen when a possible schedule is selected. If the user presses this button, the application will attempt to save the selected schedule. If the application could save schedule, it shall notify the user that the schedule was saved. If the application could not save schedule, it shall notify the user that the schedule could not be saved. The application shall then remain on the select schedule page if it failed, or take the user back to the main page if it succeeded.

After this is performed, assuming it was successful, the user will be able to view or re-verify the saved schedule.

4.11 "Remove Schedule" Requirements

The application shall display a button to remove the currently saved schedule on the view schedule page. If the user presses remove schedule button. Application shall prompt the user to confirm this action. If the user confirms, the application shall delete selected schedule and take the user back to the main page. If the user does not confirm, the application shall not delete selected schedule, instead the application shall return the user to view schedule page.

For this to function properly, the user must have a schedule stored on the application.

4.12 "Block Out Time" Requirements

If the user chooses to block out time in their schedule, the application shall prompt the user to input a block of time and the day(s). The application shall then get data from the user interface and check the data for validity. If the application detects empty input, the application shall take the user back to input a new time/day(s) slot. If the application detects half empty input, the application shall default the missing input. If the missing input is the starting time, the application shall default the starting time to 00:01. If the missing input is the ending time, the application shall default the ending time to 23:59. If the missing input is the day(s), the application shall default to blocking out time slot for the entire week. If the user input is valid, the application then shall add a blank class with the time/day(s) associated with it to the schedule.

For this to function properly, the user must be logged in.

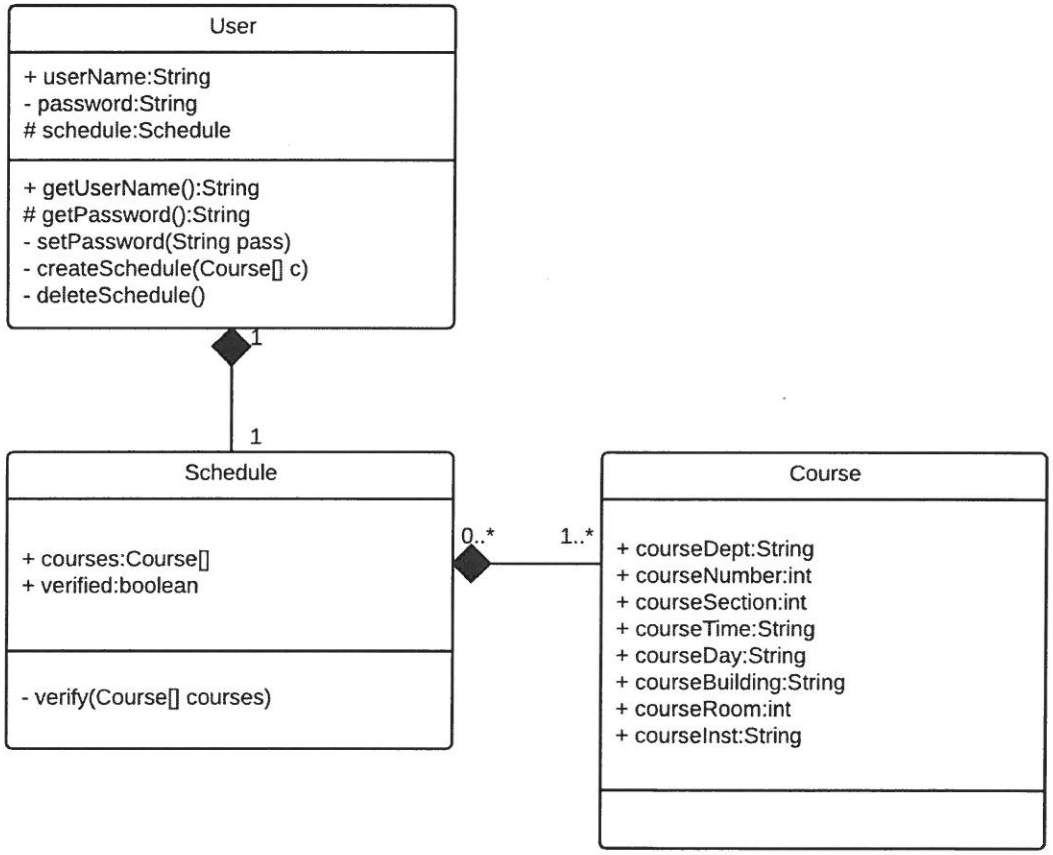
5. Software Processes and Infrastructure

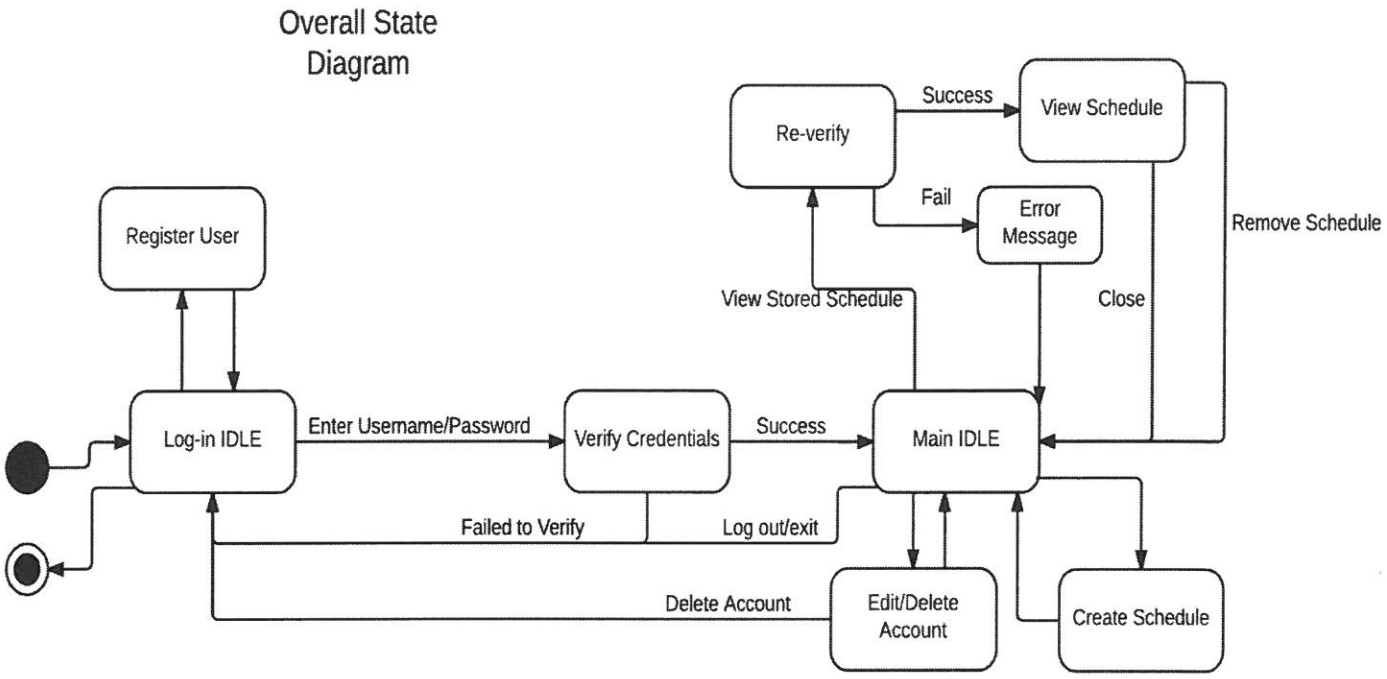
5.1 Hardware and Infrastructure

The system will need to be able to run on Android's 4.1 API "JellyBean". The hardware will work similar to three tier architecture. The phone will handle all the organizing of classes. It will receive the information about the classes from a server that is hosted on a data center. The data center will gather the information from the UTA's servers.

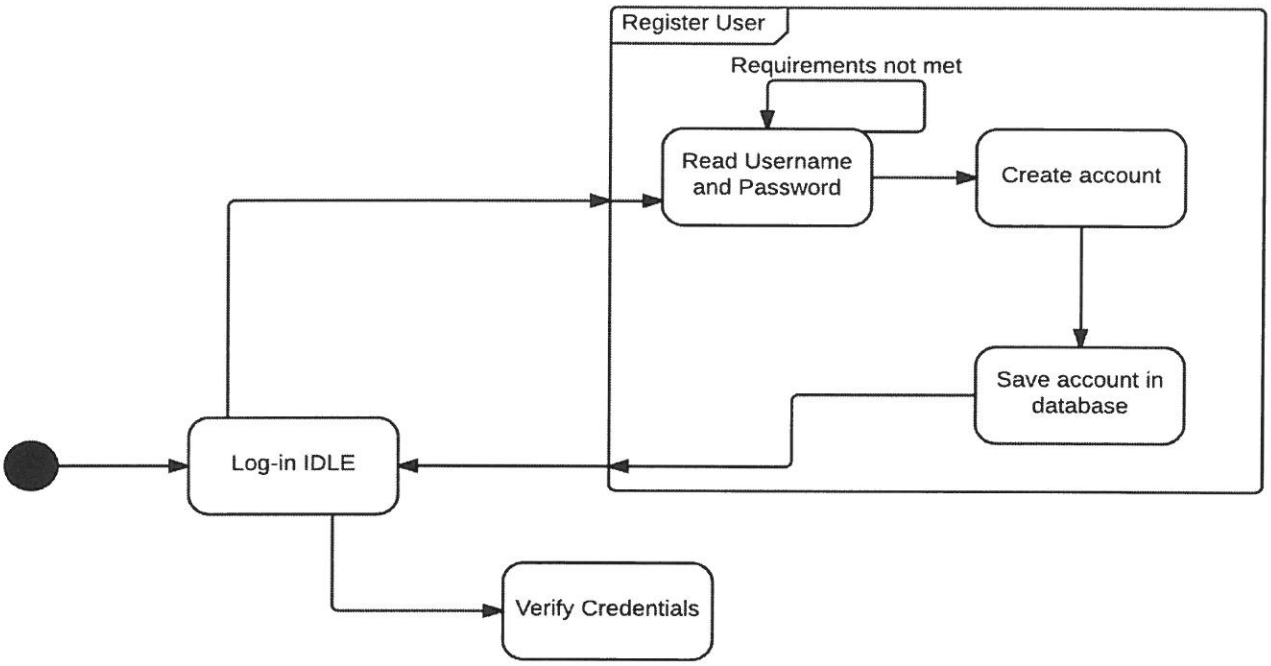
5.2 UML Diagrams

Class Diagram:

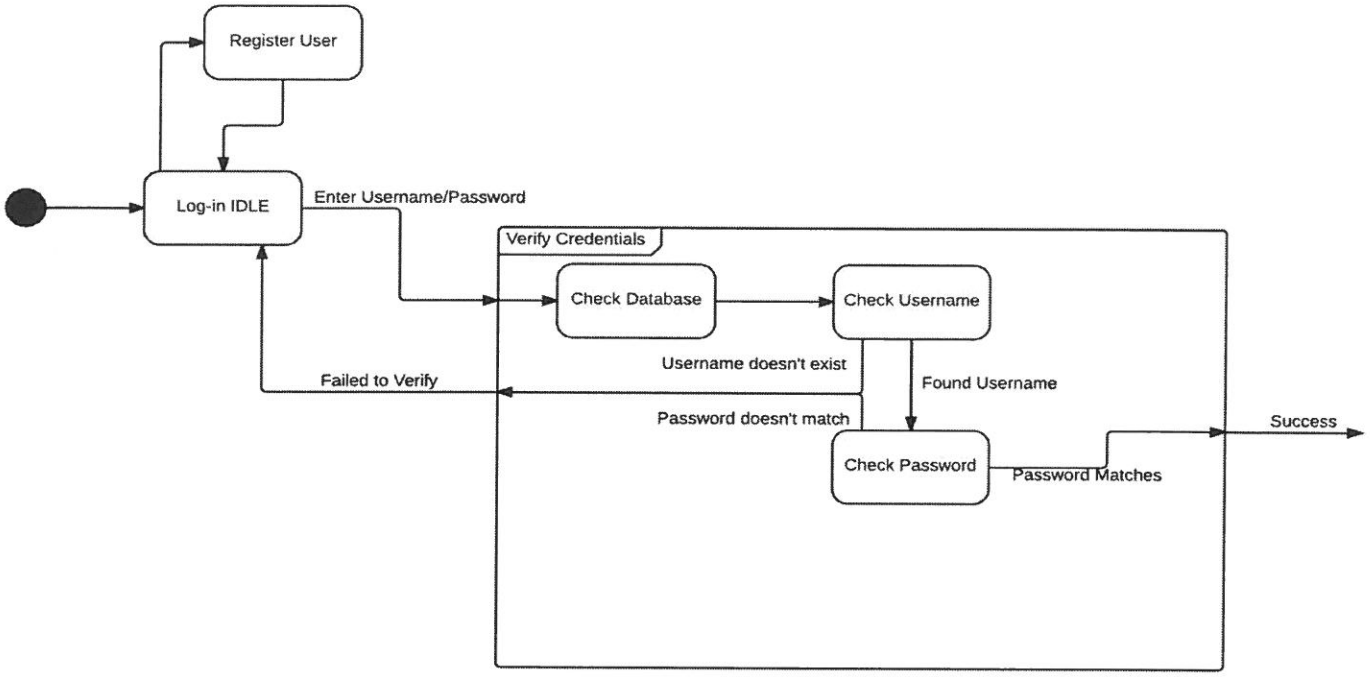



Overall State Diagram
Re-verify
Success
View Schedule
Fail
Error Message
Remove Schedule
Register User
View Stored Schedule
Close
Log-in IDLE
Enter Username/Password
Verify Credentials
Success
Main IDLE
Failed to Verify
Log out/exit
Delete Account
Edit/Delete Account
Create Schedule

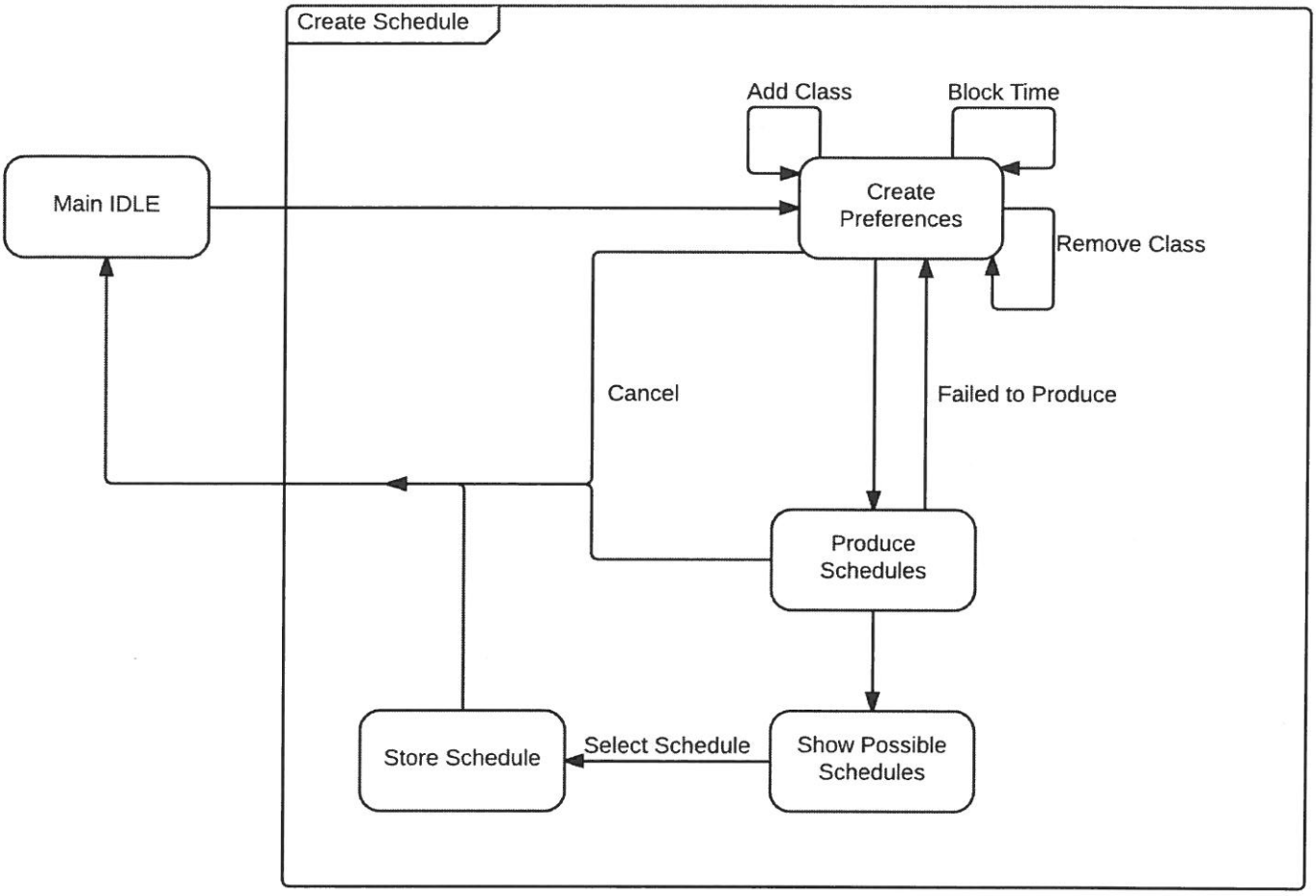
REGISTER USER DIAGRAM:



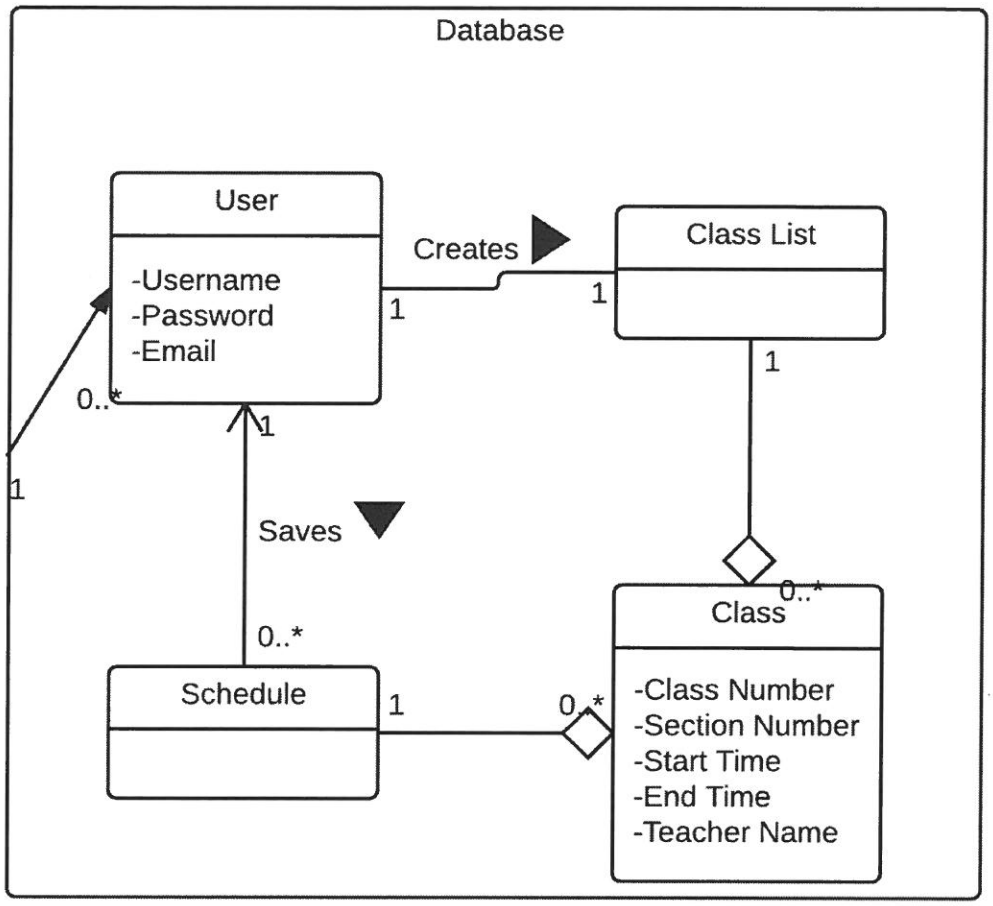
LOGIN DIAGRAM:



CREATE SCHEDULE DIAGRAM:



DATABASE DIAGRAM:



5.3 Conceptual Data Model - Database

The database will contain login information about users, their saved schedules and their class list. The database will also contain the file that updates that keeps the class information. The file that contains the updates on class information will be updated manually.

5.4 Screen Shots

None available at this time.

5.5 Test Plan

A test plan will be provided at a later stage of the project.

6. Assumptions and Constraints

6.1 ASSUMPTIONS

The following is a list of assumptions:

- All users are students of UTA.
- All users are over the age of 18.
- Ignore any legal issue for this nonprofit project.
- Ignore post project maintenance issues.

6.2 CONSTRAINTS

The following is a list of constraints:

- Project will run on android.
- Schedule is very aggressive.
- Project will be required to use minimal data.
- Project will be required to use minimal battery.

6.3 OUT OF SCOPE MATERIAL

The following is a list of "out of scope" material:

- Post project maintenance is not covered.
- Updates to student's actual UTA schedule.
- Changing of UTA class times.

7. Delivery and Schedule

Task/Milestone Description	Anticipated Start Date	Anticipated End Date	Status	Comments
Prepare Requirements and UML diagram	1/29/2015	2/26/2015	Completed	
SRA document (Includes project objectives, Requirements and UML diagrams)	2/27/2015	3/31/2015	Completed	Deliverable will be the SRA document. All stakeholders agree on the content of the SRA by signing in section 8.
Presentation of SRA	3/31/2015	3/31/2015	Completed	Robert Aron and Ian Strnad presented.
Application Design		TBA	In Progress	
Database Design		TBA	In Progress	
Test Data Entry		TBA	In Progress	
Test Plan Delivery		4/23/2015	In Progress	
... List all tasks ...				
External Documentation (i.e. User Manual)		5/7/2015	In Progress	
Final Milestone: project delivery		5/7/2015	In Progress	

8. Stakeholder Approval Form

Stakeholder Name	Stakeholder Role	Stakeholder Comments	Stakeholder Approval Signature and Date
Joseph Wigner	Development Mgr		
Elise Austin	Project Assistant		
Aaron Hamilton	Developer		
Ian Strnad	Developer		
Ji-Yeon Jeong	Developer		
Joshua Clark	Developer		
Robert Aron	Developer		

Appendix:

None

Project:	**M.S.S (Maverick Scheduling Service)**
Team No.:	**Omicron**
Class:	CSE 3310; Spring 2015
Module:	Test Plan
Deliverable:	Test Plan Document

Version: [1.0] **Date: [04/21/2015]**

Prepared by:

Aaron Hamilton
Ian Strnad
Ji-Yeon Jeong
Joshua Clark
Robert Aron

Contributors:

Aaron Hamilton
Ian Strnad
Ji-Yeon Jeong
Joshua Clark
Robert Aron

Revision History

Version number	*Date*	*Originator*	*Reason for change*	*High level description of changes*
1.0	04/21/2015	Aaron Hamilton Ian Strnad Ji-Yeon Jeong Joshua Clark Robert Aron	Initial draft	

TABLE OF CONTENTS

1. Introduction and Plan of Approach

Provide a brief description for the following areas:

- Project overview
  - M.S.S (Maverick Scheduling Service) is a collaborative effort by Team Omicron to help students of UTA plan a complex schedule with minimal effort.
  - Students will be able to plan a school schedule and tailor it to their personal preferences. They will have the ability to block out days and/or time blocks to help facilitate their time management.
  - Students will be able to create an account which stores all of their schedules and preferences on a database.

- List components that will be covered in your Test Plan
  - Logon and Entry
  - Add User
  - Delete Account
  - Log Out
  - Add Class
  - Remove Class
  - Add Schedule
  - Select Schedule
  - View Schedule
  - Remove schedule
  - Add Blocked Out Time
  - Verify Schedule
  - Save Schedule

- List any particular approach or test tools that you intend to use (e.g. test cases, using a testing tool such as TestStudio part of the Rational test suite, refer to this link for more testing options: http://www.aptest.com/resources.html). Please note the sample tables (and approach) provided in the next few pages are just suggestions, you do not have to follow this format and come up with your own.

- Include any assumptions and anomalies that you have in your Test Plan
- Any other comments relevant to this Test Plan (e.g. notations used). Remember Test Plan is a live document that gets continuously updated until the final system is delivered. A Test plan should consider positive as well as negative test cases. Stress/Volume testing typically is required; however due to low usage of this product, ignore that at this stage.

2. Test Cases: "Logon and Entry"

Project Name: MSS
Test Case Name: Logon and Entry
Test Case Id: CSE3310/Spring 2015/GroupOmicron/LogonAndEntry

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Ideal Case (Enter a valid user id/password into the User and password fields)	System should log the user into their account and move them into the default page.	
TC2	Enter user name that isn't in the database	Warn that user account does not exist.	
TC3	Enter user name and password that doesn't match.	Application will warn the user that the password and username did not match.	
TC4	Enter information, but cannot connect to server.	Application should warn user that database connection is not available.	

3. Test Cases: "Add Account"

Project Name: MSS
Test Case Name: Add User
Test Case Id: CSE3310/Spring 2015/GroupOmicron/AddAccount

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Enter valid information into id/password fields and click Create button.	System will create account in the database and alert user that account has been created.	
TC2	Enter no username.	System will warn user that a username is a required field.	
TC3	Enter no password	System will warn user that no password was Entered.	
TC4	Enter username/password that has characters that are not accepted in	System will warn or the correct characters that are accepted by the system.	

4. Test Cases: "Delete Account"

Project Name: MSS
Test Case Name: Delete Account
Test Case Id: CSE3310/Spring 2015/GroupOmicron/DeleteAccount

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Click delete Account	Will ask user to check user's intention.	
TC2	Press "yes" in check box	Will close check box and delete user's information.	
TC3	Press "No" in check box	Will close check box	
TC4	Account does not exists	Warns the user that the account does not exists and moves them to the log out screen.	

5. Test Cases: "Log Out"

Project Name: MSS
Test Case Name: Log Out
Test Case Id: CSE3310/Spring 2015/GroupOmicron/LogOut

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Ideal Case (User already logged in)	System moves to the log in screen.	
TC2	User not logged in	Throw error/Access denied. Notifies User.	

6. Test Cases: "Add Class"

Project Name: MSS
Test Case Name: Add Class
Test Case Id: CSE3310/Spring 2015/GroupOmicron/AddClass

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Ideal Case (department exists, proper class number format, class exists, class is open)	Application adds class to list	
TC2	Department does not exist	Application gives an error, notifies user, doesn't add anything, requests new input	
TC3	Improper department format	Application gives an error, notifies user, doesn't add anything, requests new input	
TC4	Improper class number format	Application gives an error, notifies user, doesn't add anything, requests new input	
TC5	Class does not exist	Application gives an error, notifies user, doesn't add anything, requests new input	
TC6	Class is not open	Application gives an error, notifies user, doesn't add anything, requests new input	

7. Test Cases: "Remove Class"

Project Name: MSS
Test Case Name: Remove Class
Test Case Id: CSE3310/Spring 2015/GroupOmicron/RemoveClass

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Ideal Case (department exists, proper class number format, class is in list)	Application removes class from list	
TC2	Department does not exist	Application gives an error, notifies user, doesn't remove anything, requests new input	
TC3	Improper department format	Application gives an error, notifies user, doesn't remove anything, requests new input	
TC4	Improper class number format	Application gives an error, notifies user, doesn't remove anything, requests new input	
TC5	Class is not in list	Application gives an error, notifies user, doesn't remove anything, requests new input	

8. Test Cases: "Add Schedule"

Project Name: MSS
Test Case Name: Add Schedule
Test Case Id: CSE3310/Spring 2015/GroupOmicron/AddSchdule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Click the create schedule button to make schedule with correct information	Produces three different schedules for the user to pick from	
TC2	Click the create schedule button to make schedule with wrong information	Will tell you invalid entry and to check your list of classes.	
TC3	Click the create schedule button to make schedule and nothing in the class section	Tell user to add classes to search for	
TC4	Click the create schedule button to make schedule and cannot create schedule from the information	Tell the user can not create schedule from current class list	

9. Test Cases: "Select Schedule"

Project Name: MSS
Test Case Name: Select Schedule
Test Case Id: CSE3310/Spring 2015/GroupOmicron/SelectSchdule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comment)
TC1	Ideal Case (Data retrieved from the server no errors)	Move to view schedule screen	
TC2	Schedule not found in the database	Throw database error. Alert user that schedule does not exist.	

10. Test Cases: "View Schedule"

Project Name: MSS
Test Case Name: View Schedule
Test Case Id: CSE3310/Spring2015/GroupOmicron/ViewSchedule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	User Presses "View Schedule" button on main screen and schedule is previously stored in memory	Application displays view schedule screen.	
TC2	User Presses "View Schedule" button on main screen and schedule is not previously stored in memory	Application displays "No schedule to view, Please create new schedule" and returns user to main screen	

11. Test Cases: "Remove Schedule"

Project Name: MSS
Test Case Name: Remove Schedule
Test Case Id: CSE3310/Spring 2015/GroupOmicron/RemoveSchedule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	User Presses "Remove stored schedule" button and then confirms.	Application removes stored schedule and displays "successfully removed schedule" then returns user to main screen	
TC2	User Presses "Remove stored schedule" button and then does not confirm.	Application shall not remove stored schedule. Application should return user to View schedule screen	
TC3	User Presses "Remove stored schedule" button then confirms. Database cannot find schedule.	Throws cannot find schedule error. Alerts user.	

12. Test Cases: "Block Out Time"

Project Name: MSS
Test Case Name: Block Out Time
Test Case Id: CSE3310/Spring 2015/GroupOmicron/BlockOutTime

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	User fills out days of the week, start time, end time with valid information.	Application adds blocked time to the list.	
TC2	User fills out start time, end time with valid information but leaves days of the week blank.	Application fills in days of the week with Mon-Sat, then application adds blocked time to the list.	
TC3	User fills out days of the week and start time with valid information but leaves end time blank.	Application fills in end time with 23:59, then application adds blocked time to the list.	
TC4	User fills out days of the week and end time with valid information but leaves start time blank.	Application fills in start time with 00:01, then application adds blocked time to the list.	
TC5	User fills out days of the week with valid information but leaves start time and end time blank.	Application fills in start time with 00:01 and end time with 23:59, then application adds blocked time to the list.	
TC6	User fills out start time with valid information but leaves days of the week and end time blank.	Application fills in days of the week with Mon-Sat, then end time with 23:59, then application adds blocked time to the list.	

TC7	User fills out end time with valid information but leaves days of the week and start time blank.	Application fills in days of the week with Mon-Sat, then start time with 00:01, then application adds blocked time to the list.	
TC8	User leaves but leaves days of the week, start time, and end time blank.	Application displays error message "Can't add blank time block" then returns user to schedule preferences screen.	
TC9	Invalid data is detected	Application displays error message "Can't add time block with invalid days/time" then returns user to schedule preferences screen.	

13. Test Cases: "Verify Schedule"

Project Name: MSS
Test Case Name: **Verify Schedule**
Test Case Id: CSE3310/Spring2015/GroupOmicron/VerifySchdule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Click the verify schedule button to verify schedule with correct information	Will tell user "Verified".	
TC2	Click the verify schedule button to verify schedule with wrong information	Will tell user "Invalid" and inform what classes make conflicts.	
TC3	Click the verify schedule button to verify schedule and no classes in schedule.	Will tell user "Invalid" and inform what classes make conflicts.	

14. Test Cases: "Save Schedule"

Project Name: MSS
Test Case Name: **Save Schedule**
Test Case Id: CSE3310/Spring 2015/GroupOmicron/SaveSchdule

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Click the save schedule button to Save schedule with correct information	Data will be saved in application directory. And tell user "saved".	
TC2	Click the save schedule button to save schedule with wrong information	Will tell user invalid data and to check details in schedule.	
TC3	Click the save schedule button to save schedule and no schedule in list.	Tell user to create schedule before save	

User Manuel
Maverick Scheduling Service
(M.S.S)
Group Omicron
5/7/2015

Contents

1. Register Account: Before using Maverick Scheduling Service (M.S.S), user needs to obtain authority to access into the application. To obtain authority, user should register his/her account Id, password and email.

.

1. To start register account, user needs to switch to register page from Log in page. In Log in page, user can find "Need to register?" on bottom of Log in page.
2. User should fill out the blanks with required information. The required information are account Id, password, confirm password and email address.
3. After user fill out the required information, user can create his/her own account by touching REGISTER button.

2. Log In: To use Maverick Scheduling Service (M.S.S), user needs to log into the application with his/her own account. To log in, user need to fill out two blanks in Log in page.

1. To log in, user needs to fill out two blanks with required information as requirements are shown in blanks.
2. After user fill out the required information, user can access into the application by clicking LOGIN button.

3. Log Out: When user wants to log out, user can find LOG OUT button on action bar, which is on right corner top of Main page.

1. To log out, user can click the action bar to find LOG OUT button.
2. On action bar, there are Setting button and LOG OUT button.

3. User can log out his/her account simply by touching LOG OUT button.

4. Change Password: User can change his/her password with Edit option in the application. User can find EDIT button in Main page. Once, user switched page from Main to Edit User. In Edit User page, user can find Change Password button, Delete Account Button and Back button. Edit, Change

1. In Change Password, User can see two blanks and Save New Password button and Back button.
2. To change password, user should fill out blanks with required information.
3. And then, just touch Save New Password button to save new password.

5. Delete Account: When user doesn't want to use the application anymore, user can delete his/her account with Delete Account option. User can find this option in Edit User page. Edit Account

1. To delete user account, user can delete his/her account simply by touching DELETE ACCOUNT button on Edit user page.

6. Add a Class: When a user wishes to add a class they can navigate to the create schedule screen of the app from there they click the Add Class Button.

1.After arriving at the Add Class Screen the user is given the ability to search by entering search criteria of department and class numbers.

2. After entering the information that they wish to search for the user can clicks the add class button.

- This step utilizes the Parse library and servers. The data for all UTA classes has been uploaded by our admins in order to give a fully functioning library of all available classes.

3.The user is shown a list of classes which meet their criteria.

-When the user's class list is full and they click on the application will warn the user that the class cannot be added to their class list. When the class has the ability to be added to their schedule, the app will save their selection onto our parse database for safe keeping. The data can be re-obtained be logging in to your MSS account. Up to ten classes can be stored in this list.

4. This completes the user adding a class to their class list. Their list will be displayed for overview.

7. Remove Class: Removing a class is much more simple than adding one. Clicking from the Create schedule activity the User has the ability to manage the classes they wish to remain in the class list.

1.The user can search through their class list by sliding their finger up and down the class list box.

-Open slots for classes to be added are notated by the slot saying "empty". This allows the user to see that they have the ability to add more classes if they wish.

2.After selecting a class that they wish to delete. The user can tap the name of the class they wish to remove, then click the "Remove Selected Class" button.

-This action will notify our parse database and notify it to remove it from your class list.

--The following implementation is not complete--

8. Create Schedules: Saving a class is a simply task after a class list has been formed. The create schedule function is used after a complete class list has been formed and the user wishes to save their class onto our database. The schedule creation also picks out the sections that work best for the uesr allowing them to take all of their classes.

1.The first step in creating a schedule in createing a class schedule is to complete a class schedule.

-The class schedule creation is a crucial step in creating a class. It's needed in order to know which classes the user wishes to be put into.

2. After the user has completed their class list, they can navigate to the class schedule page. Here the user selects their save state by taping the save state list to select a state, then clicks the save button.

-Thats it! The schedule is created and saved to the save location on our parse servers for safe keeping.

9. Add Blocked Time: User might not want to see specific days or time on his/her schedule. Add Blocked Time can be found in Create Schedule page. Create Schedule page can be switched from Main page through CREATE Schedule button in Main page.

1. User can freely decide to choose block out days by checking the check box.
2. Also, user can decide to block out time by typing specific time in Start Time and End Time.

.