

Software Requirement Specification

For

Majorizer: The Student's Personal Assistant

Version 2.0

Prepared By: Jared Heidt, Jeremy Dewey, Josh Gillette, and Andrew H. Shaw

Team Rocket Inc.

Table of Contents

1. Introduction.....	4
1.1 Purpose of <i>Majorizer</i>	
1.2 Noteworthy Document Conventions	
1.3 Intended Audience	
1.4 Projected Scope of <i>Majorizer</i>	
1.5 References and Notes	
2. Overall Description	6
2.1 <i>Majorizer</i> Perspective	
2.2 Main Features of <i>Majorizer</i>	
2.3 User Classes and Characteristics	
2.3.1 Undergrad Students	
2.3.2 Grad Students	
2.3.3 Academic Advisors	
2.3.4 Administrators	
2.4 Operating Environment	
2.5 Design and Implementation Constraints	
2.6 User Documentation	
2.7 Assumption and Dependencies	
3. System Features.....	9
3.1 Login Page	
3.1.1 User Credentials	
3.1.2 Username and Password Requirements	

3.1.3 Account Locking	
3.2 Home Page	
3.3 Academic History Page	
3.4 Scheduling Page	
3.5 Student Profile Page	
3.5.1 Students	
3.5.2 Advisors	
3.6 User Notifications	
3.6.1 Schedule Changes	
3.6.2 Major Changes	
3.6.3 Student/Advisor Changes	
3.6.4 Locked and Unlocked Accounts	
3.6.5 Curriculum Changes	
3.6.6 Account Changes	
3.7 About Us	
3.8 User Accounts Page	
3.9 Curriculum Page	
3.10 Student Search Page	
3.11 Advisees Page	
4. External Interface Requirements.....	14
4.1 User Interfaces	
4.2 Software Interfaces	
5. Other Nonfunctional Requirements.....	15
5.1 Performance Requirements	
5.2 Safety Requirements	
5.3 Security Requirements	

1.0 – Introduction

1.1 – Purpose of *Majorizer*

For an incoming freshman, college can be a very confusing and intimidating environment. Finding the correct lecture halls, organizing one's schedule, and even finding a professor to speak to can all be challenging for a new student. These are all reasons why *Team Rocket Inc.* is developing *Majorizer: The Student's Personal Assistant*. *Majorizer* is an Android phone application that will help students by providing an easy-to-use resource for students to edit and view their curriculums, including class requirements, obtainable degrees, credit systems, as well as advisor information. **Version 1.0** will only be applicable for students attending Clarkson University. Also, initially the curriculums available to users will be limited to a few select programs such as computer science and mathematics.

1.2 – Document Conventions

While reading this Specifications Document, a few details should be noted. First off, if a different section of the document is being referred to, that section number will be mentioned in **Bold** and *Italics*. For example, "The application's Log-in restrictions shown in **5.3** explain how a user . . . " would be a valid reference to Section 5.3 of this hypothetical document. Secondly, items marked with a * at the beginning and ** at the end of their description have been removed and since the original plan, but may be implemented or referenced in the future.

1.3 – Intended Audience & Reading Suggestions

The reading of this document will be ideal for any developer who is going to have to modify or perform maintenance on this software. It would also be advantageous for those taking on the role of *Admin* in this application to read this. This is because the admin has a significantly greater amount of responsibility and capabilities than the other users. Students and advisors should not need to read this documentation as the software should be intuitive to use for the targeted consumers.

As well as the people who will be using and maintaining this software, entire school campuses will be using this software. This means that any roles at a university that

will be using this software should take the time to read this document for a better understanding of the software.

Majorizer, as mentioned in Text **1.1** is a phone application primarily focused on incoming college students. The primary types of user(s) will be either 4-year students, graduate students, academic advisors, and admins. If the reader(s) of this document require information regarding a specific account type, refer to the **Table of Contents** for the section regarding that specific account.

1.4 – Project Scope of *Majorizer*

Majorizer will be used by entire college communities so it will be as intuitive for the users as possible. There will be **4** main groups of users able to have an account once the software is set up. These will include the basic four-year student, the graduate student (If there is a graduate school for the university using *Majorizer*), the student advisors (which could be a wide range of different professors), and finally an *Admin* account (which will have authority over the other accounts).

Accounts and account information will be kept in a private database protected by a basic *Username/Password* system. Some information could be available to one account type, but not another. The combination of a username and a password will also store the account type when logging in.

The 4 different accounts will have different capabilities. For example, the students will be able to edit their schedule, however they have to get their schedule change approved by an advisor. The different account pages will all have the same style/theme, but the pages will all contain different functionalities.

1.5 – References and Notes

Templates Used:

- https://moodle.clarkson.edu/pluginfile.php/673218/mod_resource/content/1/SRS_Example_2_2011.pdf - Siege Towers SRS
- https://moodle.clarkson.edu/pluginfile.php/673216/mod_resource/content/1/SRS_Example_1_2011.pdf - Spilt Pay SRS

2.0 – Overall Description

2.1 – *Majorizer* Perspective

Majorizer will be made from scratch from the developers of *Team Rocket Inc.* No other companies are having any involvement. However, once completed it could be used to replace dysfunctional academic systems such as *Peoplesoft* (used by Clarkson University). *Majorizer* will provide assistance with many important aspects of anybody's college education such as class listings and details or even differences between degrees. This software will be easier to update, easier to maintain, and easier to navigate as a user than most other software made for this purpose.

2.2 – Main Features of *Majorizer*

2.2.1 Login Details

- Username/Password Login
- Every account is 1 of 4 types
- Account information private to user

2.2.2 User Types

- Undergraduate Students
 - View/Edit major(s) and minor(s)
 - View Course History
 - Search Courses
 - *Edit Schedule/Request Schedule Change
- Graduate Students
 - View Course History
 - Search Courses/Seminars
 - *Edit Schedule/Request Schedule Change**
- Academic Advisors
 - Search Students
 - View/Edit Student Information
 - View Requests
 - Switch Students' Advisors
- Administrator
 - Create/Delete/Edit Accounts
 - View/Edit Curriculum
 - Create/Delete/Edit Courses

2.3 – User Classes and Characteristics

The difference in user types is a vital part in what will make *Majorizer* functional. The ideal situation, as mentioned briefly in **Section 1.3** contains a large user base of undergraduate and graduate students, a much smaller amount of advisors (possibly ratio of **10 students : 1 Advisor**), and most likely one or two admins.

2.3.1 Undergrad Student

Undergraduate Students will ideally be the largest group of users for any instance of *Majorizer* at a certain school. These students do not need any previous experience of any sort because the program will be made very intuitive for the students. Undergraduate students however will have a restricted set of capabilities compared to other account types. They will have the ability to view all of their completed courses as well as what courses they still have left for their degree. The students will be able to search all other courses offered as well.

*Finally, a student can make unofficial changes to their schedules as well as their major(s) and minor(s). If the student chooses, they then have the option to request from their academic advisor that the changes become permanent. It should be noted that the undergraduate students don't have the ability to change any information in the system.**

2.3.2 Graduate Students

The account used by graduate students is just like the undergraduate account seen above in **2.3.1** except for a few small differences. First, the two accounts have a completely different set of courses to choose view. These accounts also lack the option for viewing or editing one's major(s) and minor(s) because graduate students realistically do not have the option to change their studies.

2.3.3 Academic Advisors

The account used by academic advisors will be able to provide a series of actions for a student. The academic advisor will be able to approve or deny student major/minor addition/removal requests. *The advisor will also be able to add/remove courses.** The academic advisor account will also be able to view a student's course history. The advisor account will be able to request an administrator to remove a student from their advisees. When a student gets locked out of their account, the academic advisor will receive a notification.

2.3.4 Administrators

The account used by administrators will have a number of privileged actions that they may perform. Each administrator can add, edit, and delete a course from the total curriculum. The administrator will be able to create and delete accounts of students and advisors. Administrators will also be able to create a curriculum for students and advisors. When a student or advisor is locked out of an account, the administrator will receive a notification. The administrator will have privilege to grant access to one's

account after being locked out.

2.4 - Operating Environment

Majorizer was initially planned to be a web-based application for the Google Chrome, Mozilla Firefox, and Microsoft edge web browsers. However, it has been developed as a phone application for Version 1.0. The next step for broadening operating environments would be to extend the app to iPhones. A web-based application may also be added to future versions for more diverse access to its tools.

2.5 - Design and Implementation Constraints

Version 1.0 of *Majorizer* should not cause any problems for outside developers. Most of the application is going to be open-sourced (see 4.2)

2.6 - User Documentation

Once a user logs into their account, on their homepage there will be an option to open and read a brief description of their account. The descriptions will be available on the users' homepages rather than the login screen because each account type is different (see 2.3).

2.7 - Assumptions and Dependencies

Majorizer does not require that much of the user(s) other than having a stable internet connection as well as a phone to connect with.

However, user(s) should not be using *Majorizer* without consulting with their academic advisors first. For universities that decide to use this application, students will only be able to access features as basic as logging in if they are given permission on their account by a higher-up. *Majorizer* is not responsible for directly managing any single user-base. Most responsibilities having to do with student users are held to the admin(s) and the advisors.

3.0 – System Features

Unless otherwise stated, it can be assumed for all features that the page includes a stimulus of clicking on the Android back button which will return to the last page that was viewed.

3.1 – Login

3.1.1 Description and Priority

The login page is the first page displayed when the app is opened. It is a portal to the various user accounts' home pages. Three incorrect login attempts results in the user being locked out of their account, each incorrect attempt and the eventual locking will display a notification to the user. **First Level Priority.**

3.1.2 Stimulus/Response Sequences

To login the user must enter a username and password into the appropriate boxes and click the login button.

3.1.3 Functional Requirements

Requirement 1: Pull database info

Requirement 2: Verify input, username and password must match exactly (case sensitive)

Requirement 3: Track failed login attempts in the database

Requirement 4: Provide failed attempt notifications

Requirement 5: Access user data upon login

Requirement 6: On login redirect to user home page

3.2 – Home Page

3.2.1 Description and Priority

The user home page provides access to all of the user account type's tools. The different tools are listed in the form of tiles. **First Level Priority**

3.2.2 Stimulus/Response Sequences

The user can interact with this page in four ways. First, clicking on tool tiles redirects to the appropriate page. Second, swiping to the left navigates to the user's notifications page. Clicking on the titles of other pages at the top of the page achieves the same as swiping. Lastly, clicking on the Android back button logs out of the user account.

3.2.3 Functional Requirements

Requirement 1: Page redirect, this is the only requirement as the home page is merely a central hub for the user to access their account tools.

3.3 – Notifications Page

3.3.1 Description and Priority

The notifications page displays various types of notifications for different account types. Notifications are displayed in list view. Plain notifications can be dismissed with no further actions, however approval notifications require the user to approve or decline the request before the notification is dismissed. ***Third Level Priority***

3.3.2 Stimulus/Response Sequences

Clicking on notifications on this pages opens a popup for the user to choose how to respond to the notification. This entails dismissing the notification, approving requests, and declining requests.

3.3.3 Functional Requirements

Requirement 1: Pulling from the notifications database

Requirement 2: Updating the notifications database

Requirement 3: Completing user database changes(due to approved requests)

Requirement 4: Creating new notifications in the database about request updates

3.4 – Account Info Page

3.4.1 Description and Priority

This is a simple page that only displays information about the user account. ***First Level Priority***

3.4.2 Stimulus/Response Sequences

No stimulus on this page.

3.4.3 Functional Requirements

Requirement 1: Display information from the user class

3.5 – Create User Account

3.5.1 Description and Priority

This admin tool creates a new user account, these include; advisor, undergraduate student, and graduate student. The account only contains basic user information. ***First Level Priority***

3.5.2 Stimulus/Response Sequences

First, the admin chooses an account type to add. The admin must then enter data into each text field, including; username, first name, last name, password, and major(s)/minor(s).

Once complete the user clicks the create account button.

3.5.3 Functional Requirements

Requirement 1: Input verification, each field must meet certain requirements.

- Username: must be six, lowercase letters
- Password: must have six to twelve characters, with minimum one uppercase, one lowercase, one number, and one special character
- Password again: must match the first password field

- Major: for undergrads minimum one, but possibly two, must be selected. For grad students only one major may be chosen
- Minor; for undergrads minimum zero, but up to two, must be selected. For grad students this field does not appear

Requirement 2: Save new account information to the database

3.6 – Delete User Account

3.6.1 Description and Priority

This admin tool deletes all user data for a chosen account from the database. ***First Level Priority***

3.6.2 Stimulus/Response Sequences

The admin searches for keywords in the account data, i.e. username, first name, and last name. Once the account is found the admin can click on the item in the search results, this causes a pop up to verify that the admin wishes to delete the account.

3.6.3 Functional Requirements

Requirement 1: Dynamic search through the contents of the user database

Requirement 2: Popup verification of account selection

Requirement 3: Delete user data from the database

3.7 – Unlock Account

3.7.1 Description and Priority

This admin tool unlocks a user account which was locked due to failed login attempts. All currently locked accounts are listed on the page with a button to unlock it. ***Second Level Priority***

3.7.2 Stimulus/Response Sequences

The admin only needs to find the account in the list and click on the unlock button.

3.7.3 Functional Requirements

Requirement 1: Pull locked status from user database

Requirement 2: Change locked status of accounts in the database

3.8 – Create Master Course

3.8.1 Description and Priority

This admin tool creates a new course that is not defined yet in the system. This only includes basic information such as the course ID, name, number of credits, and department. ***First Level Priority***.

3.8.2 Stimulus/Response Sequences

First the admin must choose whether the course will be a graduate or undergraduate course. Then they only need to fill out each text field and select a department from the list.

3.8.3 Functional Requirements

Requirement 1: Verify input data

- Course number must be an integer
- Course name must be a string
- Number of credits must be either three or four

Requirement 2: Saving data to the master course database

3.9 – Delete master Course

3.9.1 Description and Priority

This admin tool deletes a master course from all instances of the database ei master course list and curriculums. *First Level Priority.*

3.9.2 Stimulus/Response Sequences

The admin must find the course in a course search, then select the course. This causes a popup to verify that they want to delete the course.

3.9.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Dynamic search through the data

Requirement 3: Delete data from the database

Requirement 4: Pop up verification

3.10 – Add Course to Curriculum

3.10.1 Description and Priority

This admin tool adds a master course to a specific undergrad or grad major, or a specific undergrad minor. *First Level Priority.*

3.10.2 Stimulus/Response Sequences

The admin first chooses whether the course is being added to a grad or undergrad curriculum, then chooses a department, or in the case of undergrad possibly a minor.

After this is chosen the page shows a course search, once the admin has found a course they select it from the list.

3.10.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Dynamic search through the data

Requirement 3: Add data to the database

3.11 – Remove Course from Curriculum

3.11.1 Description and Priority

This admin tool removes a master course from a specific undergrad or grad major, or a specific undergrad minor. ***Second Level Priority.***

3.11.2 Stimulus/Response Sequences

The admin first chooses whether the course is being added to a grad or undergrad curriculum, then chooses a department, or in the case of undergrad possibly a minor. After this is chosen the page shows a course search, once the admin has found a course they select it from the list. This causes a popup to verify selection,

3.11.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Dynamic search through the data

Requirement 3: Delete data to the database

Requirement 4: Pop up verification

3.12 – View Advisee Info

3.12.1 Description and Priority

This advisor tool allows them to view the same info about their advisee as the advisee sees. ***First Level Priority.***

3.12.2 Stimulus/Response Sequences

The page lists all advisees under the advisor, selecting one brings up the same homepage layout that the student would have had they logged into the account. From here interaction follows the same rules specified by other respective pages.

3.12.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Pull specific student data from the database

Requirement 3: Redirect page

3.13 – Add Advisee

3.13.1 Description and Priority

This admin tool allows the advisor to add a student as their advisee. If this student is already under another advisor a request is sent to that advisor for the switch. ***First Level Priority.***

3.13.2 Stimulus/Response Sequences

The page shows a student search, the advisor uses this to find the student. When a student is selected a pop up is shown to verify the selection.

3.13.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Dynamic search

Requirement 3: Creating new notification in database

Requirement 4: Change user data in the database

Requirement 5: Pop up verification

3.14 – Drop Advisee

3.14.1 Description and Priority

This advisor tool allows an advisor to drop an advisee. *Third Level Priority.*

3.14.2 Stimulus/Response Sequences

The page shows a list of the advisors' students. When the advisor clicks on a student it shows a pop up to verify the selection.

3.14.3 Functional Requirements

Requirement 1: Pull data from the database

Requirement 2: Pop up verification

Requirement 3: Delete data from the database

3.15 – View Course History

3.15.1 Description and Priority

This tool simply allows students to view their previously taken courses, the grades that they received in them, how many credits each was, and their cumulative GPA. *Second Level Priority.*

3.15.2 Stimulus/Response Sequences

Clicking on a course in the list brings up a simple pop up that shows the grade the student got in the course and the credits the course is worth.

3.4.3 Functional Requirements

Requirement 1: Pull data from database

3.16 – View Required Courses

3.16.1 Description and Priority

This tool simply allows students to view the courses that they have yet to take but are required for their minor(s) and in the case of the user being an undergrad, minor(s). *Second Level Priority.*

3.15.2 Stimulus/Response Sequences

Clicking on a course in the list brings up a simple pop up with basic information about the course.

3.4.3 Functional Requirements

Requirement 1: Pull data from database

3.17 – View Current Courses

3.17.1 Description and Priority

This tool displays the courses the user is currently enrolled in and the credits each is worth. ***First Level Priority.***

3.15.2 Stimulus/Response Sequences

No stimulus on this page.

3.4.3 Functional Requirements

Requirement 1: Pull data from database

3.18 – Edit Major/Minor Request

3.18.1 Description and Priority

This tool allows undergrad students to request to add, drop, or switch their major(s) and, in the case of the user being an undergrad, minor(s). ***First Level Priority.***

3.18.2 Stimulus/Response Sequences

The page displays a list of all majors and minors available to the student. Selecting these brings up a variety of pop ups. The following table describes them:

Current number of majors/minors	Currently enrolled in major/minor	Pop up
2	Yes	Drop
2	No	Switch
1	Yes	Cannot Drop or Add
1	No	Add
0 (only minor)	No	Add

3.18.3 Functional Requirements

Requirement 1: Pull from Data

Requirement 2: Set data in the database

Requirement 3: Pop up verification

4.0 – External Interface Requirements

4.1 – User Interfaces

As stated, Majorizer is sectioned off into four different pages (administrator, advisor, undergraduate, and graduate). Each page entails different actions, so naturally the layout of the page and its subpages will also differ while keeping a general theme. **Appendix 1** provides all layouts that a user can expect to see and interact with in sequential order. When clicked, each button clicked will perform the stated action for the user, launching a fairly simple layout which the user can interact with. As a user navigates through tools the only way to return to previous pages is with the Android back button at the bottom of the screen. Future designs on iPhones and web browsers will need to supplement this with other navigation features. Basic activity notifications will be given to the user in the form of small, brief popups at the bottom of the screen. Conversely, when a user selects an item on a list to act on a popup will be shown in the center of the screen to verify their selection.

4.2 – Software Interfaces

Coding for *Majorizer* was done in Android studio, a IDE designed specifically for Android app development. The app is completely coded using Java. In terms of storing the data, the data is stored in a *Firebase* database, which uses a JSON format. JSON format allows data to be stored and organized in a predictable manner that can be quickly serialized and unserialized by Javascript. It is recognized that data must be quickly made available to the user, and it is expected that this data can be made immediately available upon a users request.

5.0 – Other Nonfunctional Requirements

5.1 – Performance Requirements

Majorizer has few requirements for its performance. Firstly, the app requires enough storage space on the device to download. In order to use the app the device must have reliable internet access, either over wifi or cellular data.

5.2 – Safety Requirements

Majorizer relies on universities to maintain a safety. The software is innately safe, however the way that a university distributes user information will decide the safety of their users accounts.

5.3 – Security Requirements

Majorizer has secure data storage on *Firebase*, this leaves the only safety concern to be others obtaining access to a user's login credentials. This may be further alleviated in future versions through the implementation of password resets, encryption, and login device verification.

6.0 – Key Milestones

#	Milestone	Target Completion Date	Comments
1.	Set up <i>Firebase</i> database	September 15, 2018	Basic structure
2.	Login functionality	October 12, 2018	Verification and locking
3.	Admin capabilities	October 26, 2018	Needed before setting up courses and other accounts
4.	Complete database	November 30, 2018	Full classes for MA and CS major/minor
5.	Account info pages	November 2, 2018	
6.	Student course history and requirements	November 9, 2018	
7.	Advisor tools	November 16, 2018	
8.	Notifications & approvals	November 23, 2018	Must be done after account setup & functionality
9.	Finalized UI	November 30, 2018	

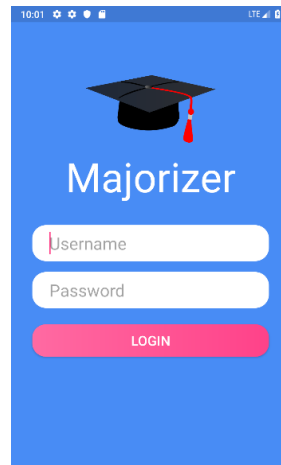
7.0 – Revision History

Name	Date	Reason For Changes	Version
Initial	September 28, 2018	Initial design	1.0
Android App	October 8, 2018	A phone based app is more applicable to students.	2.0

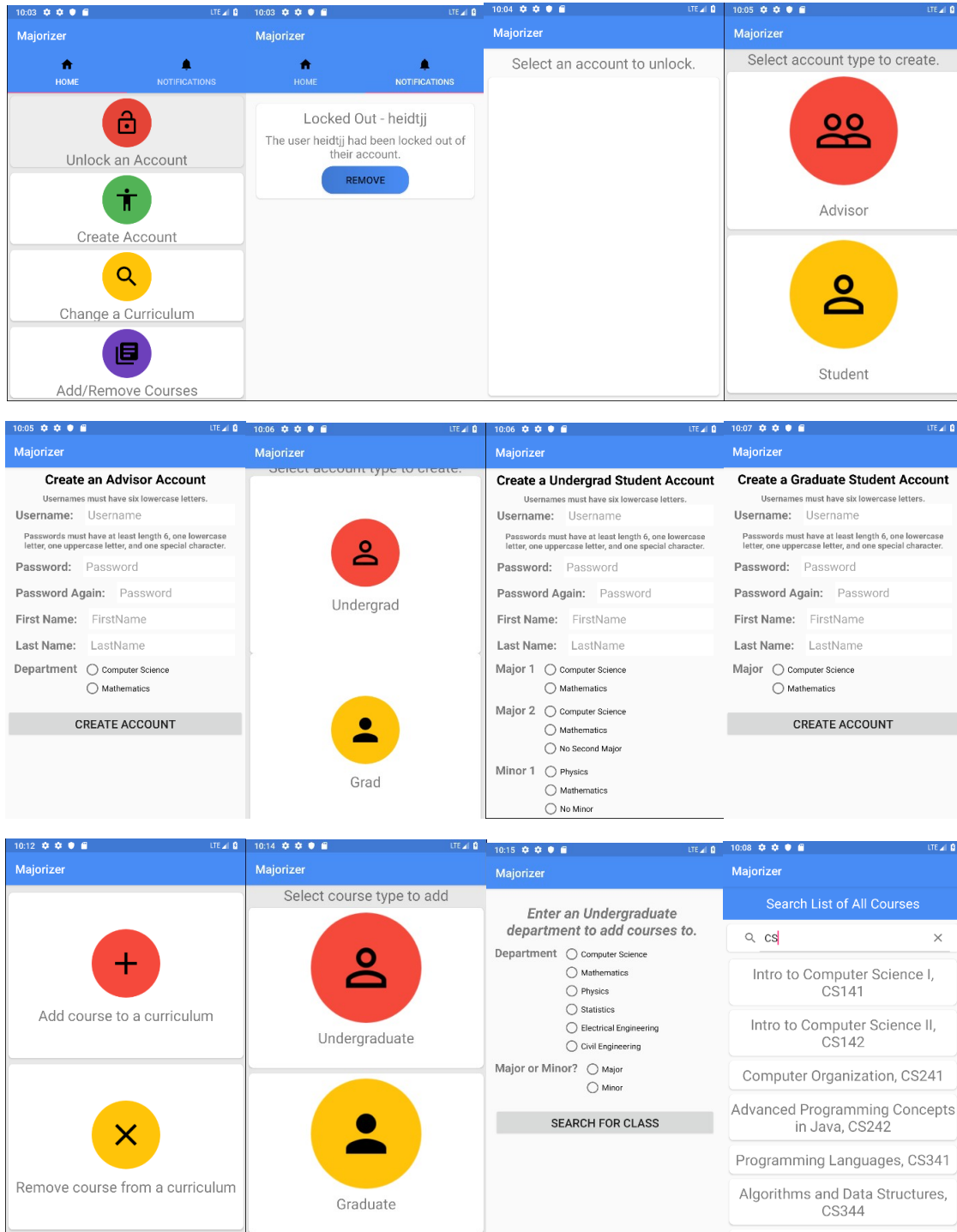
Appendix

1 Screen Layouts

1.1 – Login



1.2 – Admin



10:16

Majorizer

Enter a Graduate department to add courses to.

Department

☐ Computer Science

☐ Mathematics

☐ Physics

☐ Statistics

☐ Electrical Engineering

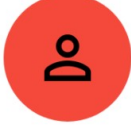
☐ Civil Engineering


SEARCH FOR CLASS

10:17

Majorizer

Select course type to remove

Undergraduate

Graduate

10:17

Majorizer

Enter an Undergraduate department to remove courses from.

Department

☐ Computer Science

☐ Mathematics

☐ Physics

☐ Statistics

☐ Electrical Engineering

☐ Civil Engineering

Major or Minor? ☐ Major ☐ Minor

SEARCH FOR CLASS

10:18

Majorizer

Enter a Graduate department to remove courses from.

Department

☐ Computer Science

☐ Mathematics

☐ Physics

☐ Statistics

☐ Electrical Engineering

☐ Civil Engineering

SEARCH FOR CLASS

10:10

Majorizer

Select course type to add

Undergraduate

Graduate

10:11

Majorizer

Create a Graduate Course

Course numbers must be 500-999.

Course Number: 500 - 999

Course Name: Intro to Math

Course credits must be 3-4.

Number of credits: 3 or 4

Department

☐ Computer Science

☐ Mathematics

☐ Physics

☐ Statistics

☐ Electrical Engineering

☐ Civil Engineering

CREATE COURSE

10:11

Majorizer

Create an Undergraduate Course

Course numbers must be 1-499.

Course Number: 1 - 499

Course Name: Intro to Math

Course credits must be 3-4.

Number of credits: 3 or 4

Department

☐ Computer Science

☐ Mathematics

☐ Physics

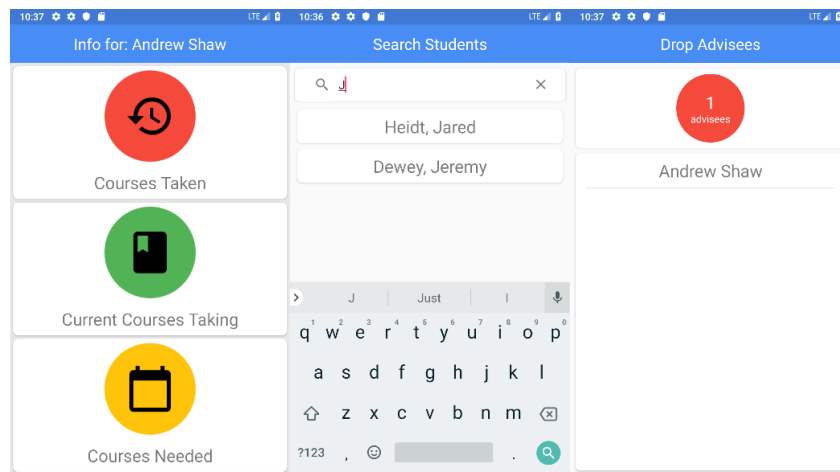
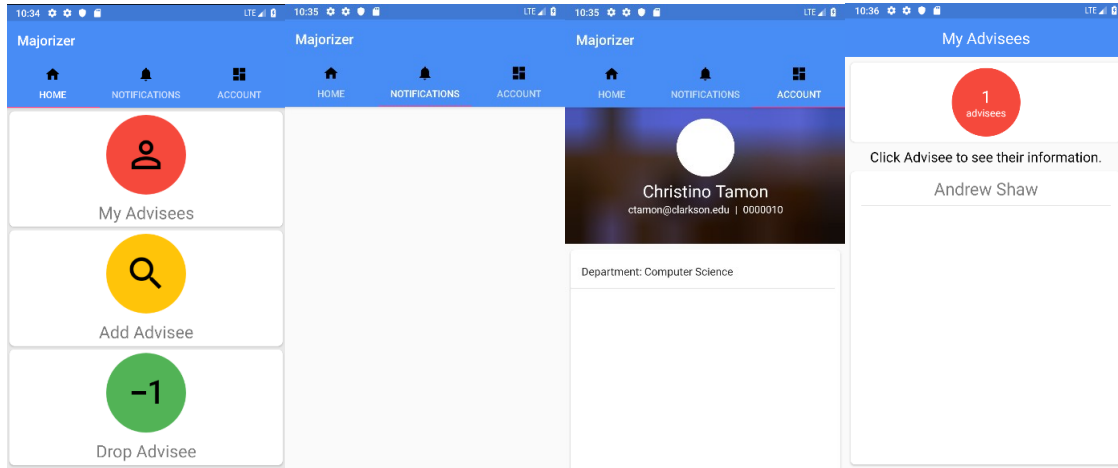
☐ Statistics

☐ Electrical Engineering

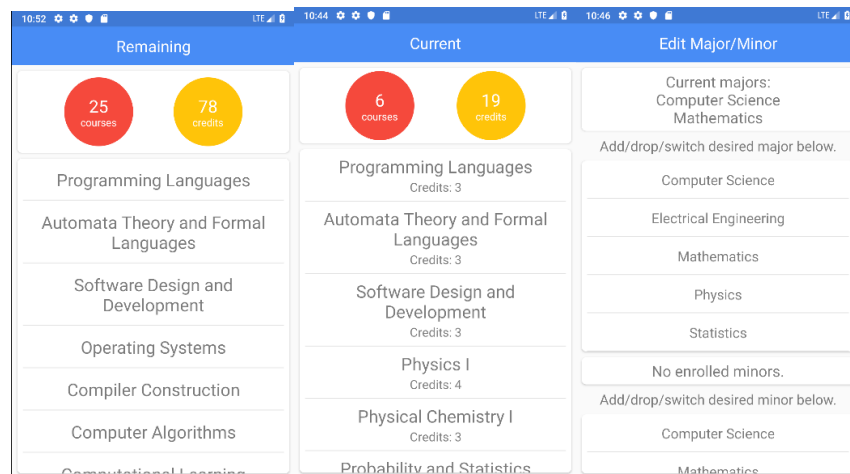
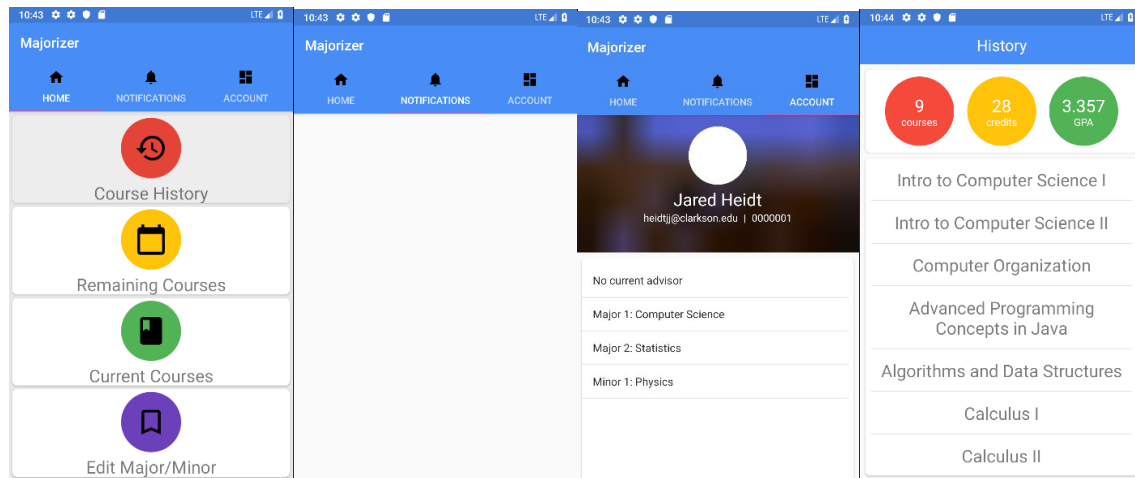
☐ Civil Engineering

CREATE COURSE

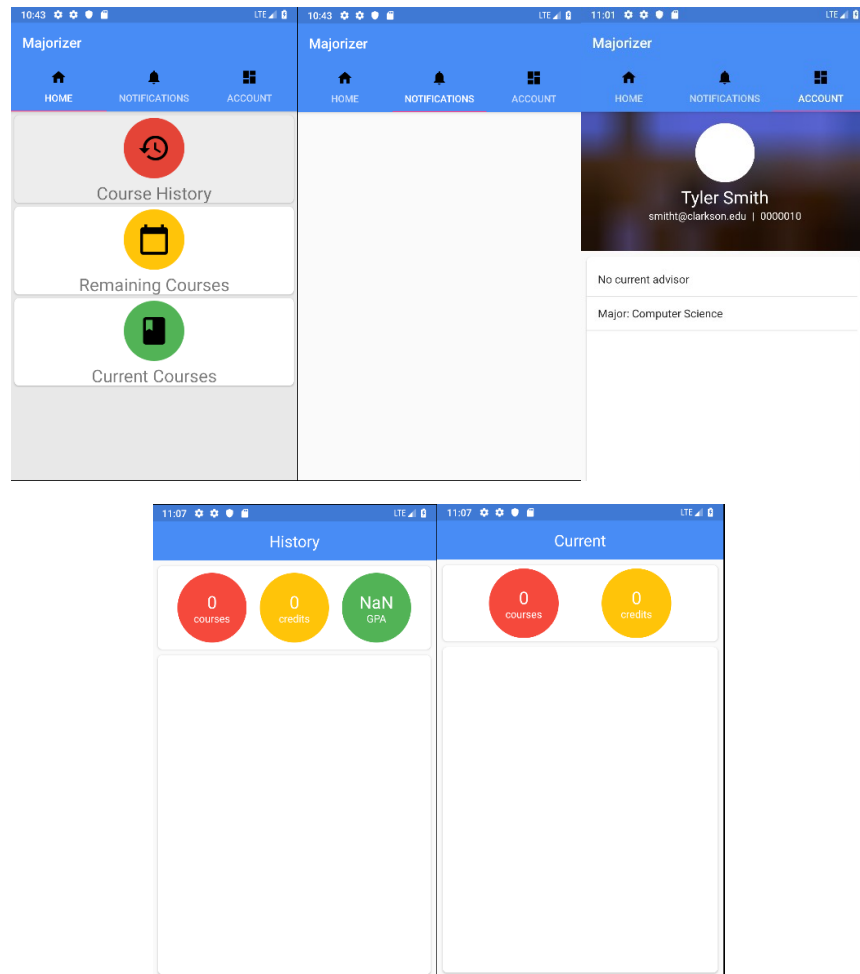
1.3 – Advisor



1.4 – Undergraduate

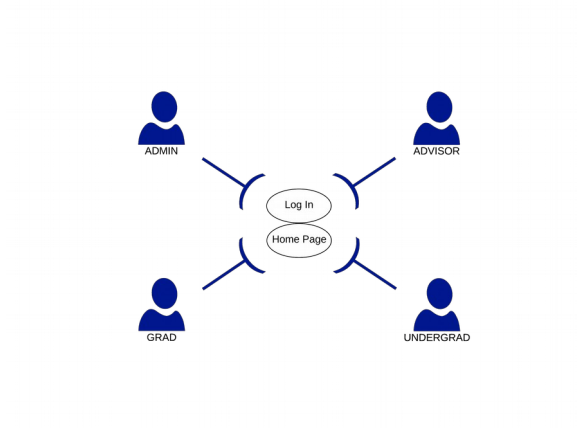


1.5 – Graduate

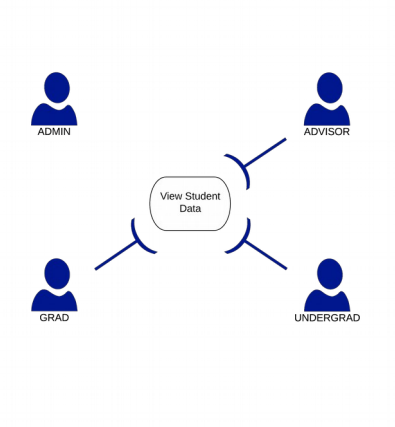


2 Case Diagrams

2.1 – All Accounts



2.2 – User Accounts



2.3 – Individual Accounts

