Group Name: Half Moon Studio

Project Idea: Study Scheduling App - PrioritU

Scrum Team:

* Dominic Smart 816007605 (Scrum Master)
* Shereece Victor 816001671 (Product Owner)
* Jerrel Williams 816009134
* Selah Lewis 816012230

Introduction:

The main goal of this project is to develop a working prototype of the PrioritU Scheduling app outlined in the requirements document. Initially the project was intended to be an improved version of the My study Life App available on Android devices and through a web application. However, further refinement of the groups ideas created a simpler web interface aimed at making it easier for students to manage their time.

The purpose of this design document is to illustrate and provide details about the architectural design, interface design and core functionalities via sequence diagrams. This document will also define the projected timeline for implementation of the various design elements using a scrum approach.

Purpose:

This purpose of this document is to outline the approach decided upon by Team Half Moon Studio to implement the PrioritU application according to the requirements specified. To date, no particular architecture pattern has been chosen for this project.

Scope: (To be done)

(Give details on how much of the projects designs will be covered by this document. This section is a great way to cover our asses for if we are unable to do all the work required for submission.)

Definitions and Abbreviations: (To be done)

Project Timeline:

Development period

Week 1 - Oct 13th - 19th

Set up database, website, establish link

Week 2 - Oct 20th - 26th

Add APIs and Middleman classes

Week 3 Oct 27th- Nov 2nd

Test Implemented functions, refine code

Week 4 Nov 3rd -9th **MILESTONE 4 DUE**

Test and refine GUI

Week 5 Nov 10th - 16th

Add and Test secondary functionality, prepare presentation

Week 6 Nov 17th - 23rd

Finalise Documentation, present project

Design Elements (Based on Requirements)

* Database / Server for storing user information, settings and configuration
  + Includes manager for information requests, additions, updates, deletions changes.
  + User information
    - Name
    - Email
    - Number of Courses
    - Courses information
  + Courses (Course code, name, duration, colour?)
    - Classes (times, days, type, teacher?, room)
    - Exams (time, date, duration, type - final/coursework)
    - Assignments (time, date, name, complete/ incomplete)
* Front end Dashboard and user interface
  + Settings?
  + Calendar (Google api)
  + Recommendations
  + Lists of upcoming tasks and exams
  + Sign out button
  + FAQ tab or menu option (I don’t think we should do a menu, that implies several pages. Would a dashboard be easier to make than several pages?)
* Middleman/ Processing server
  + Sorting Classes for displaying the tasks and assignments due in order.
  + Display manager for applying colours and themes to displays
  + Reminder plugin
  + Calendar plugin
  + Admin access or control portal? (ignore this)
  + Login plugin and authentication
  + Add or modify exams
  + Add or modify assignments

Outline of Use Cases

1. Student Logs in
2. Student signs out
3. Student adds New Course
4. Student adds new Class (extending: add new course)
5. Student modifies course
6. Student adds new exam
7. Student modifies exam
8. Student adds new Assignment
9. Student modifies assignment
10. Student marks assignment as complete
11. Student changes sorting order for tasks due list
12. Student chooses colour scheme

Extended Use Cases (One example added for formatting)

Use Case 3: Student Adds new course

|  |  |
| --- | --- |
| Description | A student adds one of their courses to the app |
| **Actors** | Student |
| **Type** | Primary, Essential |
| **Pre-Conditions** | - The Student has logged into the application |
| **Flow of Events** | |
| Basic Path | |
| Actor Action | System Response |
| 1. The use case begins when the student selects the option to add a new course | 1. The system displays the add course form |
| 1. The student enters the course code. | 1. The system checks if the course code already exists. |
| 1. The student enters the course name |  |
| 1. The student selects a colour for the course |  |
| 1. The student selects the duration during which the course runs. |  |
| 1. The student may enter the lecturer's name. |  |
| 1. The student submits the form. | 1. The system acknowledges the completion of the form and updates the database and dashboard. |
| Alternative Paths | |
| Line 2: The course already exists. (see Student modifies course) | |
| Post-Conditions | A new course has been added to the app |
| **Related Use Cases** | Student modifies course |
| Used Use Case | None |
| Extending Use Cases | None |