

PLANiT Scheduler

PLANiT

Sam Flinkfelt
Xuanru He
Lidel Mendoza
Anne Poso
Brianna Sharpe

CPSC 362-01/02
Sara Ghadami

Table of Contents

Document Revision History Table	4
Project Plan	5
User Story (15 user stories)	6
User-Stories (5 non-functional requirements)	7
Use-case (textual) (15)	8
Use Case: Create Account	8
Use Case: Login	9
Use Case: Add task	10
Use Case: Check off tasks	11
Use Case: Add class subjects	12
Use Case: Password Recovery	13
Use Case: View Agenda	14
Use Case: View Individual Tasks	15
Use Case: Upload Class Pictures	16
Use Case: Edit Tasks	17
Use Case: View Weather	18
Use Case: Study Timer	19
Use Case: Edit Class	20
Use Case: Delete Class	21
Use Case: Sign Out	22
Use-case diagram	23
Sprint Backlog Screen shots	24
Class Diagram	25
CRC Cards	26
Test Cases (10)	28
Test Case: Create Account	28
Test Case: Login	29
Test Case: Forgot Password	30
Test Case: Add Task	31
Test Case: Check Off Task	32
Test Case: Edit Task	33

Test Case: Delete Task	34
Test Case: Add Class	35
Test Case: Upload Picture	36
Test Case: Edit Class	37
Sequence Diagrams (3)	38
Activity Diagrams	39
Pre-game Planning	41
Staging/Grooming	42
Development Process	43
User Manual	44
References	56

Document Revision History Table

REVISION HISTORY			
Rev.	Description of change	Page No.	Date
0	Document First Iteration	--	
1	Added 5 user stories	5	3/23/2020
2	Added 5 use cases	12-16	3/23/2020
3	Updated sprint backlog screenshots	18	3/23/2020
4	Updated use-case diagram	17	4/2/2020
5	Updated user manual screenshots	27-37	4/3/2020
6	Added class diagram page	19	4/5/2020
7	Added CRC card page	20-21	4/6/2020
8	Added activity diagrams page	22-23	4/6/2020
9	Added sequence diagrams	37	4/19/2020
10	Added 10 test cases	27-36	4/23/2020
11	Revised project plan for sprint 3	4	5/1/2020
12	Added 5 more Use-Cases	16-21	5/1/2020
13	Added 5 more User-Stories	5-6	5/1/2020
14	Updated user manual screenshots	43-54	5/2/2020

Project Plan

As a new semester begins, students are in need of a more organized way of managing their time. We have come up with an application called PLANiT that will help with this. We will go over the benefits, time frame of completion, team members, and targeted group for the application.

Benefits

Our initiative is to deliver a mobile app that will provide these essential needs: alerts/reminders, uploading files (e.g. syllabus, pdf assignments), add tasks, schedule appointments and project deadlines. This mobile app is meant to assist students in organizing their class schedules, as well as keep them up to date with personal tasks, appointments, and homework assignments.

Time Frame

The proposed time frame will be around three months, broken up into three phases including application support, updates, and new features.

Team Members

We have a team of five students with an educational background in Computer Science: Sam Flinkfelt, Xuanru He, Lidel Mendoza, Anne Poso, and Brianna Sharpe (Project Manager). We all have prior experience in creating a student database applications in SQL, and inventory GUI applications in C#, PHP web applications and Java applications.

Client Communication

Our main goal is to communicate with the client to meet their needs through the development process of the application.

Customer

Mainly students will see a benefit from this product, as they can import their class schedules. Our future considerations are not limited to students but anyone who wants a little more organization in their life.

Changes from Sprint 2

Since our last sprint, we plan to get our classes page fully working with an edit, view, and delete course functionality. We also plan to implement more features that will make our app different from other time management apps, such as a weather widget and a study timer to assist students in improving their studying habits. As this is our last iteration, the GUI also needs to be updated with a design that will be more attractive to customers for our demo.

User Story (15 user stories)

- Create account
As a user, I want to create an account so that I can have my own personalized profile.
- Login
As a user, I want to be able to login, so that I can access my personalized profile.
- Add task
As a user, I want to add tasks to my task list, so that I can organize my workload.
- Check off task
As a user, I want to check off my tasks, so I can keep track of what I completed.
- Add class subjects
As a user, I want to add my class subjects so that I can track the information of my semester classes.
- Password Recovery
As a user, I want to be able to reset my password, so that I can login if I forget my credentials.
- View detailed task
As a user, I want to be able to view detailed information of my tasks, so that I can recall what I was doing at a later date.
- Edit Individual Tasks
As a user, I want to be able to edit an individual task, so that I can keep my task list updated.
- Upload Class Pictures
As a user, I want to add pictures to individual classes, so that I can keep track of class documents.
- View Weather
As a user, I want to view the weather, so that I can plan my day better.
- Countdown Study Timer
As a user, I want to be able to track my time studying, so that I can balance my time between breaks and studying.

- Edit Class

As a user, I want to be able to edit my classes, in case they change or have the wrong information.
- Delete Class

As a user, I want to be able to delete my classes, so that I can have an updated class list for the semester.
- View Weekly Agenda

As a user, I want to be able to view a weekly agenda, so that I can organize my schedule on a weekly basis.
- Sign out

As a user, I want to be able to sign out of my account, so other users cannot view my information.

User-Stories (5 non-functional requirements)

- Login authentication
 - As a user, I want the system to give me access to my homepage as soon as I login with my credentials.
- Retrieve user information.
 - As a user, I want the system to get my information even when I am using another device to log in.
- Display stored information.
 - As a user, I want the system to show my added tasks and events so I can keep track of them.
- Save information.
 - As a user, I want the system to save my information so that I can view my added tasks and events when I log in on other devices.
- Store class related information.
 - As a user, I want the system to be capable of holding information for all my classes this semester.

Use-case (textual)

Use Case: **Create Account**

Id: UC-01

Description

Before users are able to view their profile, they are required to create an account. The information needed to create an account is the following: first name, last name, username, email, and password. The system will verify this new account doesn't already exist. Once verified, user can now login to their account.

Primary Actor

Student and system.

Pre-Conditions

User must not have an existing account, which means system has to verify that user information is unique.

Post Conditions

Success end condition

User creates an account.

Failure end condition:

Username and password already exist and user cannot create account.

Password does not meet requirements, so user cannot create account.

Main Success Scenario

1. User is on login page and clicks on **Create Account**, located at the bottom of the page.
2. The system renders the page
3. User fills in the appropriate fields(first name, last name, username, email, password)
4. User clicks on **Create Account** button
5. System creates an account for the user.

Extensions

- 3a. Password must be minimum 6 characters long, otherwise the password is not accepted.
 1. User enters a password at least 6 characters long
 2. Use case resumes on step 3

Use Case: Login

Id: UC-02

Description

End user enters an email address and password, then selects the “Login” button so they can access their PLANiT scheduler homepage. If the user does not have an account created, then they will select the “Create Account” button that will redirect them to the create account screen so they may create a login.

Primary Actor

Student

Pre-Conditions

Student must have an existing account in order to login to the app.

Post Conditions

Success end condition

User has an existing account and enters correct email and password, successfully reaching the PLANiT scheduler homepage where they can access all of the app’s features.

Failure end condition:

- User does not have an existing account, so they need to go to “Create Account” screen by clicking on “Create Account” button on login screen.
- User enters incorrect credentials, so they are unable to reach scheduler homepage.

Main Success Scenario

1. System displays login screen
2. User types in correct username and password
3. User selects “Login” button and is redirected to app home screen to access all of the app’s features.

Extensions

2a. In step 2, if the user does not have a username or password, they must first create an account before they can login.

1. On the login screen, user selects “Create Account” button
2. User is redirected to Create Account screen, and creates a username and password.
3. Once verified, user is redirected back to login screen
4. Use case resumes on step 2

2b. In step 2, if the user does not remember their password, they can get a link sent to their email to reset their password.

1. On the login screen, user selects "Forgot Password" button
2. User is redirected to Reset Password screen
3. User types in email address and presses "Submit"
4. System sends user an email with a link to change their password
5. User opens email and clicks on link
6. User types in new password and presses "Save"
7. User re-opens PLANiT application
8. Use case resumes on step 2

Use Case: **Add task**

Id: UC-03

Description

Adding a Task

Creating a list of tasks, the person can press a button, write a description or notes about what task needs to be completed. Pop up a dialogue to type in what needs to be done. After submitting the task, create a running list to have a view of what tasks are available. After a list of tasks are already populated, the person using this app will be able to check off the task after they are finished with it.

Primary Actor

Whoever is using the app, mainly students

Pre-Conditions

Homepage with task button to direct the user to the task page.

Post Conditions

Success end condition

Task Completed - submission of the task into the list, being able to manipulate the task or delete it or check it off

The user will have the satisfaction of viewing a ongoing list of their tasks after they've been added to the list

Failure end condition:

The while list of tasks are unaltered.

The program crashes

The whole list of tasks are deleted.

Main Success Scenario

1. The user creates a list of tasks pressing the “add task” button
2. The user clicks on the currently populated task
3. The user selects delete task by tapping on the task in the list.
4. The user selects Remove or checks off Task and ends up back at the task page.

Extensions

A search box at the top, looking for populated lists.

1. User logs into their account
2. User clicks on the specific add task button to view their list of current tasks
3. User can view the task list
4. User either adds a task by clicking on the same add task button
 - 4a. User wants to delete task from list by clicking on the task on the list
 - 4b. Prompts user if they would like to remove task from the list and re populates list
5. Back button to exit tasks list back to homepage

Use Case: **Check off tasks**

Id: UC-04

Description

After the user is finished with a task, they are able to check off the task by clicking the circle in front of each task. Or if the user wants to completely delete the task, they can long press the circle, and a pop up to remove the task is initiated with the choice of deleting task from the existing list.

Primary Actor

Students

Pre-Conditions

Users must be directed to the task page with at least one task on it.

Post Conditions

Success end condition

The task is successfully being checked off.

The user removes the task from the existing task list.

Failure end condition:

The user is unable to check off the task.

All tasks are checked off.

The task list remains unchanged.
All tasks are deleted.

Main Success Scenario

1. User is on task page and clicks on the circle before the task they are finished with.
2. User long presses on the task which they want to delete.
3. An alert dialog of removing the task from the existing list pops up.
4. If the user selects "YES", the task is deleted from the list.
5. If the user selects "CANCEL", the list of tasks remains unchanged.

Extensions

In step 2, if the user wants to store the task they are finished with in a separate "done-list"

1. User is directed to the task page
2. User double clicks the task that they want to move it into a separate file -- "done-list", instead of removing it from the database permanently.
3. Use case resumes on step 2

Use Case: Add class subjects

Id: UC-05

Description

Student adds the information of class subjects necessary to create color coded class-related tasks.

Primary Actor

Student

Pre-Conditions

Student must be logged in.

Post Conditions

Success end condition

Class is added and can now be referred to when adding tasks.

Failure end condition:

Class is not added. Other features remain unchanged.

Main Success Scenario

1. Select the class tab
2. Select the top right '+' symbol to add class.
3. Enter class name, description, day/time, room (optional), instructor (optional).

4. Select 'add class' once necessary fields are filled.

Extensions

N/A

Use Case: Password Recovery

Id: UC-06

Description

User does not remember their password to login to the system, so the user clicks on "Forgot password" button to reset their existing password.

Primary Actor

Student (end user)

Pre-Conditions

User must have an existing account in order to reset their password.

Post Conditions

Success end condition

User changes their password and successfully logs into the system with new password.

Failure end condition:

User is unable to reset their password, so they are also unable to login to the system.

Main Success Scenario

1. System displays login page
2. User clicks on "Forgot Password" button and renders page to Reset Password screen
3. User enters their email address in text field and clicks "Submit" button
4. System sends email to user with a link to change the user's password
5. User opens email and clicks on link
6. User is redirected to a password reset page.
7. User enters new password and clicks "Save" button
8. User re-opens PLANiT application
9. System displays login page
10. User logs in with username and new password

Extensions

2a. In step 2, if the user does not have an existing username and password, they must first create an account before they can reset their password.

1. On the login screen, user selects "Create Account" button

2. User is redirected to Create Account screen, and creates a username and password.
3. Once verified, user is redirected back to login screen
4. Use case resumes on step 2.

Use Case: **View Agenda**

Id: UC-07

Description

View agenda lists tasks or events for the week and are ordered by due date.

Primary Actor

User is primary actor

Pre-Conditions

User needs to be logged in.

Post Conditions

Success end condition

User is able to see tasks that need to be completed for today.

Main Success Scenario

1. User logs in
2. System displays home page
3. User navigates to the middle of the page
4. System displays agenda
5. User views agenda

Use Case: **View Individual Tasks**

Id: UC-08

Description

User is able to expand an individual task to view more information about the task, including its description and due date.

Primary Actor

Student (end user)

Pre-Conditions

- Student must be logged into system
- Student must have existing tasks

Post Conditions

Success end condition

Student taps on task name and system displays a page of a detailed task view that lists the task name, due date, and description. There are also options to edit or delete the task in this page.

Main Success Scenario

1. User logs into system
2. System displays homepage screen
3. User presses the “+” icon
4. System displays task list page
5. User single taps on an existing task
6. System displays page with a detailed view of task

Extensions

6a. In step 6, user will be able to edit the task information

1. User selects “Edit” icon at bottom right of pop up
2. User will be able to edit the fields of the task information such as description, name, due date, and upload any photos.
3. User selects “Confirm” icon and returns to viewing mode
4. Use case resumes on step 6.

Use Case: Upload Class Pictures

Id: UC-09

Description

User adds pictures to individual classes to be displayed when viewing the class information page.

Primary Actor

User

Pre-Conditions

User must be logged in. User must have already added a class to the list of classes.

Post Conditions

Success end condition

User uploads a picture to a class.

Failure end condition:

Picture is too large to be uploaded.

File type uploaded is not acceptable.

Main Success Scenario

1. System displays the home page.
2. User navigates to the classes page.
3. System displays the classes page.
4. User selects the class of choice.
5. System displays the information of that class on a new page.
6. User selects the "+" under the "photos" header.
7. User uploads the photo from their photo library.
8. System displays the photo under the "photos" header.

Extensions

N/A

Use Case: **Edit Task**

Id: UC-10

Description

Student will be able to edit the due date and description of their existing tasks

Primary Actor

Student

Pre-Conditions

User is logged in.

User has at least 1 existing task.

Post Conditions

Success end condition

User confirms the changes and updates the information of a task.

Failure end condition:

User cancels the changes and does not update the information of a task.

Main Success Scenario

1. The user logs into the app.
2. System displays app's homepage
3. User presses "+" icon
4. System displays task list page
5. User single taps on an existing task
6. System displays page with a detailed view of task
7. User presses "Edit" button
8. System displays edit task page
9. User enters new description and due date of task
10. User presses "Confirm" button
11. System updates new task information to database
12. System redirects user to task list page

Extensions

- 10a. Instead of confirming task changes, the user can cancel changes made to a task.
 1. User presses "Cancel" button
 2. System does not update new task information to database
 3. Use case resumes on step 12.

Use Case: **View Weather**

Id: UC-11

Description

A student would be able to have a view of the weather for the day, also the week to come up ahead.

Primary Actor

Student

Pre-Conditions

The user would have to be logged in, click on the Schedule button to view the calendar and weather.

Post Conditions

Success end condition

The user has a view of the weather and upcoming days of weather.

Failure end condition:

Schedule button fails, Weather view is not available, View cannot be changed.

Main Success Scenario

1. The user logs into the app.
2. The system renders the page.
3. The user navigates to the right corner of the page and clicks on the calendar icon.
- 3.The system fetches the weather data from the database and displays it.

Extensions

N/A

Use Case: **CountDown Study Timer**

Id: UC-12

Description

Student times a task they are working on for 25 uninterrupted minutes with a countdown timer that begins at 25:00 minutes.

Primary Actor

Student

Pre-Conditions

User is logged in.

Post Conditions

Success end condition

User completes an interval of 25 study minutes when timer hits 00:00.

Failure end condition:

User pauses study timer and never reaches 00:00.

Main Success Scenario

1. System displays login page
2. User logs in to their account
3. System displays app home page
4. User presses play button on Study Timer
5. Timer begins counting down from 25:00 minutes
6. Timer reaches 00:00

Extensions

- 5a. In step 5, user is able to pause the timer
 1. User presses pause button
 2. Timer stops counting down and pauses
 3. User presses pause button again
 4. Timer resumes from where it was paused
 5. Use case resumes on step 6
- 6a. In step 6, user is able to reset timer back to 25:00 minutes
 1. User presses replay button or play button
 2. Timer resets to 25:00 minutes and begins counting down
 3. Use case resumes on step 5

Use Case: **Edit Class**

Id: UC-13

Description

Student able to edit the current information of class subjects necessary to create color coded class-related tasks.

Primary Actor

Student

Pre-Conditions

The user would have to be logged in, click on the schedule button to view the tasks.

Post Conditions

Success end condition

Class new updated information for editing is added to the class list and can now be referred to when adding tasks.

Failure end condition:

The edit class button does not work.

Main Success Scenario

1. Select the class tab
2. Select the top right '+' symbol to add class.
3. Clicks on Edit class name, description, day/time, room (optional), instructor (optional).
4. Select 'Save' once necessary fields are filled.

Extensions

N/A

Use Case: **Delete Class**

Id: UC-14

Description

Student able to delete the current information of class subject and will no longer show up on their list of classes

Primary Actor

Student

Pre-Conditions

The user would have to be logged in, click on the schedule button to view the class page.

Post Conditions

Success end condition

The Class page newly updated information and does not show on the class list with full functionality of the class page.

Failure end condition:

Adding editing or deleting a task no longer will work.

Main Success Scenario

1. The user logs into the app.
2. The system renders the page.
3. The user navigates to the right corner of the page and clicks on the class icon.
- 3.The system fetches the users data from the database and displays it.

Extensions

N/A

Use Case: **Sign out**

Id: UC-15

Description

A student would be able to sign out of their account by pressing the sign out button. This will bring them back to the login page, not being able to access their account unless logging in again.

Primary Actor

Student

Pre-Conditions

The user would have to be logged in, click on the three little dots at the top right of the home page.

Post Conditions

Success end condition

The user is back at the login page, being able to log back in.

Failure end condition:

Sign out button does not work, does nothing, re-directs to a different page.

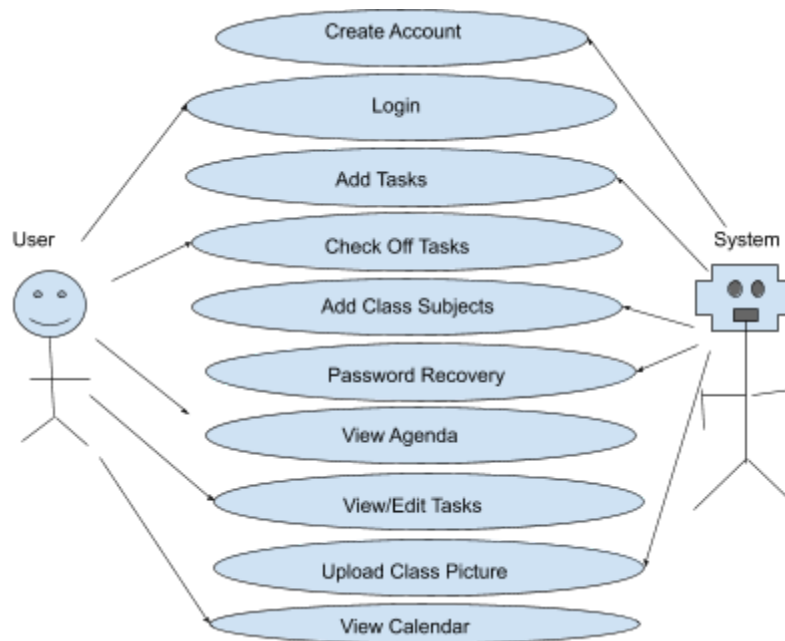
Main Success Scenario

1. The user logs into the app.
2. The system renders the page.
3. The user navigates to the top right corner of the page and clicks on the three dots icon.
3. The system logs the user out of the client, back to the home page and displays it.

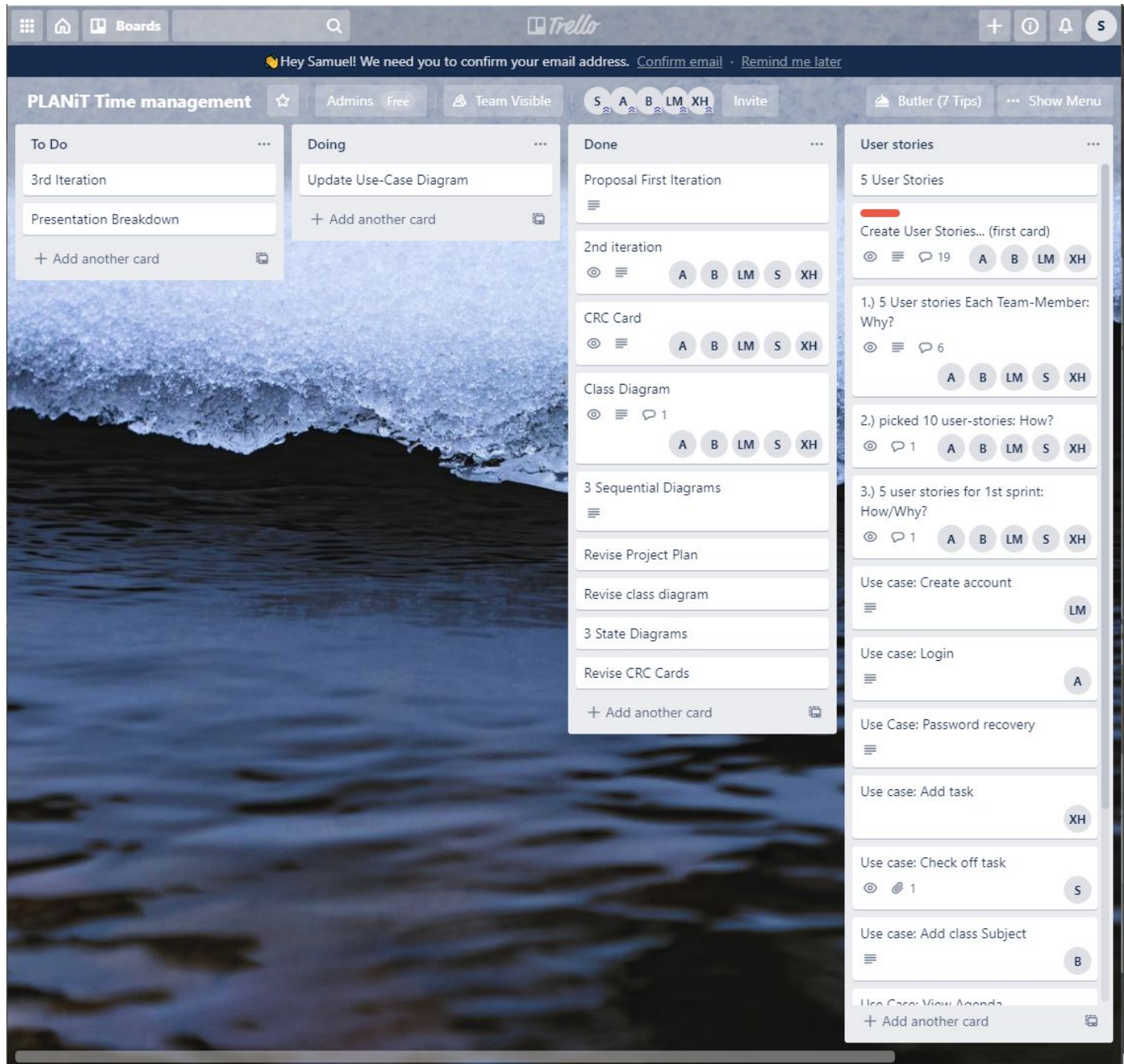
Extensions

N/A

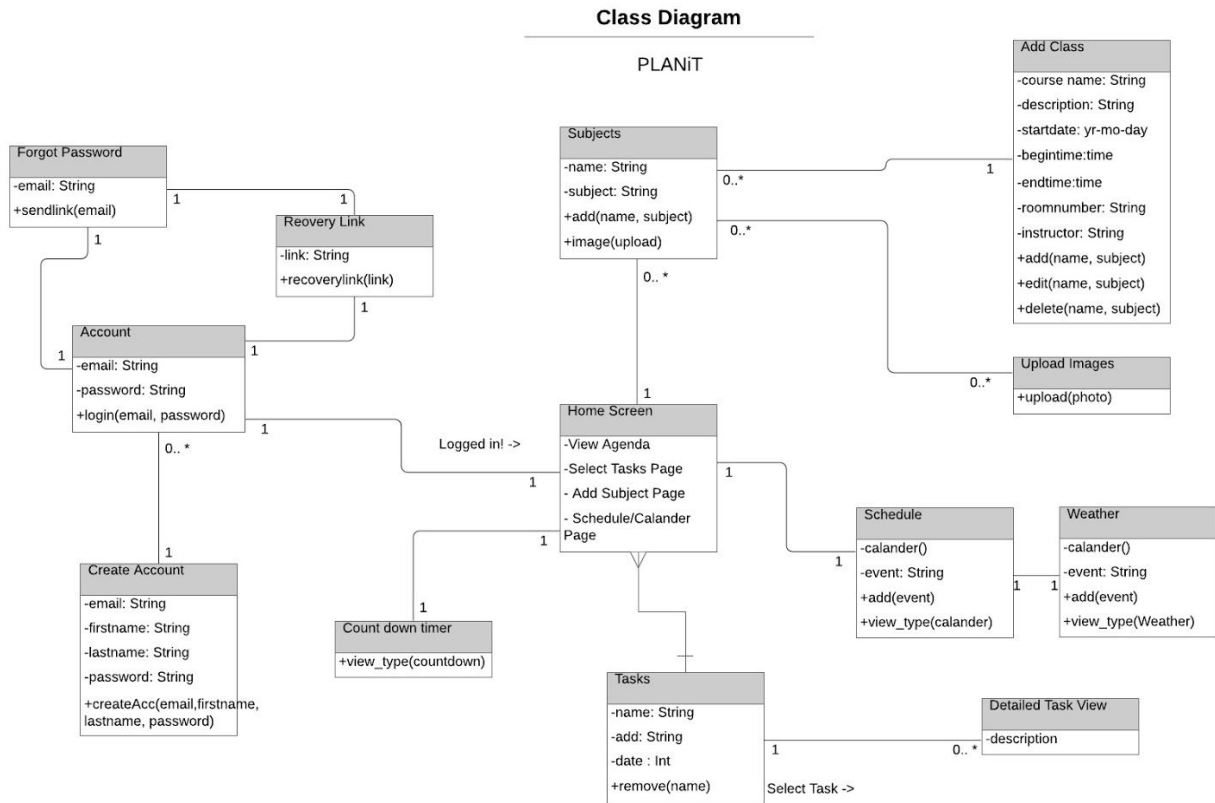
Use-case diagram: Schedule Planner



Sprint Backlog Screen shots



Class Diagram



CRC Cards

Class Name: Forgot Password	ID: 2	Type: Concrete	Attributes:		
Description: When a user does not remember their password, they can request a password reset using Forgot Password.		Associated Use Cases: 02, 06	email (text)		
Responsibilities		Collaborators			
knows email of the user account		Recovery Link			
			Relationships:		
			Generalization (a-kind-of):		
			Aggregation (has-parts):		
			Account		
			Other Associations:		
			Recovery Link		

Class Name: Forgot Password	ID: 2	Type: Concrete	Attributes:		
Description: When a user does not remember their password, they can request a password reset using Forgot Password.		Associated Use Cases: 02, 06	email (text)		
Responsibilities		Collaborators			
knows email of the user account		Recovery Link	Relationships:		
			Generalization (a-kind-of):		
			Aggregation (has-parts): Account		
			Other Associations: Recovery Link		

Class Name: Account	ID: 3	Type: Concrete	Attributes:		
Description: The accounts store information a user adds onto their schedule.		Associated Use Cases: 02	email (text) name (text)		
Responsibilities		Collaborators	password (text)		
knows user details (email, name, etc...)			Relationships:		
stores user information			Generalization (a-kind-of):		
knows added tasks and events		Task, Event	Aggregation (has-parts): Tasks, Event, Schedule		
			Other Associations:		

Class Name: Recovery Link	ID: 4	Type: Concrete	Attributes:		
Description: When a user does not remember their password, they can request a password reset using Forgot Password and receive a Recovery Link to complete the change.		Associated Use Cases: 06	link (text)		
Responsibilities		Collaborators	Relationships:		
knows email of user account		Forgot password	Generalization (a-kind-of):		
sends a link for password reset/recovery		Forgot password	Aggregation (has-parts): Account		
			Other Associations: Forgot Password		

Class Name: Create Account	ID: 5	Type: Concrete	Attributes:		
Description:		Associated Use Cases: 01	email (text) password (text)		
Responsibilities		Collaborators	first name (text) last name (text)		
creates user account		Account	Relationships:		
			Generalization (a-kind-of): Account		
			Aggregation (has-parts):		
			Other Associations:		

Class Name: Subject		ID: 6	Type: Concrete	Attributes:		
Description:			Associated Use Cases: 05	name (text)		
Responsibilities			Collaborators	subject (text)		
provides class information				Relationships:		
knows class subject name				Generalization (a-kind-of):		
				Aggregation (has-parts):		
				Other Associations: Event		

Class Name: Home Screen	ID: 7	Type: Domain	Attributes:
Description:	Associated Use Cases: 07		tasks (task)
Responsibilities	Collaborators		
provides a platform to view agenda	Subject, Schedule		Relationships:
allows navigation to other pages (classes, calendar)			Generalization (a-kind-of):
			Aggregation (has-parts): Tasks, Detailed Task View
			Other Associations: Schedule

Class Name: Schedule	ID: 8	Type: Concrete	Attributes:
Description:	Associated Use Cases: 10		event (name)
Responsibilities	Collaborators		
displays calendar	Event		Relationships:
adds events			Generalization (a-kind-of):
			Aggregation (has-parts): Event
			Other Associations:

Class Name: Event	ID: 9	Type: Concrete	Attributes:
Description:	Associated Use Cases: 10		name (text)
Responsibilities	Collaborators		description (text)
provides information on occasions to attend	Schedule		Relationships:
provides information on important dates			Generalization (a-kind-of):
knows event names			Aggregation (has-parts):
			Other Associations: Schedule

Class Name: Tasks	ID: 10	Type: Concrete	Attributes:
Description:	Associated Use Cases: 03, 07		name (text)
Responsibilities	Collaborators		date (date)
has the name of the task	Home Screen		Relationships:
has the description of the task			Generalization (a-kind-of):
			Aggregation (has-parts):
			Other Associations: Home Screen

Class Name: Detailed Task View	ID: 11	Type: Concrete	Attributes:
Description:	Associated Use Cases: 08		description (text)
Responsibilities	Collaborators		
knows all the information from tasks in agenda	Home Screen, Tasks		Relationships:
provides specific information (description, etc...)			Generalization (a-kind-of): Task
			Aggregation (has-parts):
			Other Associations: Home Screen

Class Name: Countdown Timer	ID: 12	Type: Concrete	Attributes:
Description:	Associated Use Cases: 12		description (text)
Responsibilities	Collaborators		
Once started, counts down from 25:00 minutes	Home Screen		Relationships:
and stop at 00:00.			Generalization (a-kind-of): Timer
			Aggregation (has-parts):
			Other Associations: Home Screen

Class Name: Weather	ID: 13	Type: Concrete	Attributes:
Description:	Associated Use Cases: 11		description (text)
Responsibilities	Collaborators		
Provides current weather update based on user's	Home Screen		Relationships:
phone GPS location			Generalization (a-kind-of): Weather
			Aggregation (has-parts):
			Other Associations: Home Screen

10 Test Cases

Test Case: CREATE ACCOUNT

Test Case #: 1	Test Case Name: Create Account	Page: 1 of 10
System: Mobile Device	Subsystem:	
Designed by: PLANIT	Design Date: 03/03/2020	
Executed by:	Execution Date:	
Short Description: Create an account for the PLANiT app		

Pre-conditions

The user does not have an existing account.
The system displays login page

Step	Action	Expected System Response	Pas s/ Fail
1	Tap the "Create Account" button	The system displays the Create Account page	P
2	Enter email address and password of an account that already exists	The system asks the user to enter their information	P
3	Tap the "Create Account" button	The system validates if all fields are not empty	P
4	Check post-condition 1		P
5	Enter email address and password of an account that does not exist	The system asks the user to enter their information	P
6	Tap the "Create Account" button	The system validates if all fields are not empty	P
7	Check post-condition 2		P

Post-conditions

1. System displays message to enter another email address.
2. The email address and password is saved in the database, and user is automatically redirected to app's homepage.

Test Case: LOGIN

Test Case #: 2

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: User logs into the app

Test Case Name: Login

Subsystem:

Design Date: 03/03/2020

Execution Date:

Page: 2 of 10

Pre-conditions

User must have previously made an account

Step	Action	Expected System Response	Pas s/ Fail
1	Enter username 'lidel@gmail.com' and password '123456'	The system asks for login credentials	P
2	Tap "Login" button	The system checks if login fields are not empty	P
3	Check post-condition 1		P

Post-conditions

1. User 'lidel' is directed to homepage

Test Case: FORGOT PASSWORD

Test Case #: 3

System: Mobile Device

Designed by: PLANit

Executed by:

Short Description: If user forgets password, they can reset it

Test Case Name: Forgot Password

Subsystem:

Design Date: 04/07/2020

Execution Date:

Page: 3 of 10

Pre-conditions

User must have an account

Step	Action	Expected System Response	Pas s/ Fail
1	Click on 'forgot password' button	System displays a message to enter email	P
2	Type in 'samflinkfelt@gmail.com'	The System will display the users email address	P
3	Click the Submit button	System does nothing	P
4	User opens email and clicks link	Page is redirected to type '12345'	P
5	Type in 'samflinkfelt@gmail.com' '12345'	The System will be able to login to homepage	P

Post-conditions

1. New password is updated into the server

Test Case: ADD TASK

Test Case #: 4

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: User be able to add tasks

Test Case Name: Add Task

Subsystem:

Design Date: 03/03/2020

Execution Date:

Page: 4 of 10

Pre-conditions

The user has to have a valid account, and has access to the system.
Homepage with task button to direct the user to the task page.

Step	Action	Expected System Response	Pas s/ Fail
1	Tap "+" button	User is directed to the "Add Task" page	P
2	Enter task name "Exam", and description "hard"	The systems shows the task name "Exam" and description "hard"	P
3	Select due date "05/05/2020"	The systems shows the selected date "05/05/2020"	P
4	Click "Confirm"	The new task "Exam" displays on the "Task List" page	P

Post-conditions

1. The new added task saved in the database

Test Case: CHECK OFF TASK

Test Case #: 5

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: User be able to check off tasks when it is completed

Test Case Name: Check off Task

Subsystem:

Design Date: 04/07/2020

Execution Date:

Page: 5 of 10

Pre-conditions

User is logged in

User must have at least one existing task

Step	Action	Expected System Response	Pas s/ Fail
1	User taps on the empty circle next to the task name "Exam"	System marks the task with a check mark icon	P
2	Check post-condition 1		P

Post-conditions

1. Task is marked as completed with a check mark icon.

Test Case: EDIT TASK

Test Case #: 6

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: User can change any information in their task

Test Case Name: Edit Task

Subsystem: Task

Design Date: 04/07/2020

Execution Date:

Page: 6 of 10

Pre-conditions

User has to be logged into system, and user must have existing tasks

Step	Action	Expected System Response	Pas s/ Fail
1	Click an existing task name "Exam"	The system shows the detailed task view	P
2	Click "Edit" button	User is directed to the "Edit Task" page	P
3	Change any description as "easy" and due date as "05/05/2020"	The system shows the updated description as "easy" and due date as "05/05/2020"	P
4	Click "Update" button	User is directed to the "Task List" page	P

Post-conditions

1. The updated task information is saved in the database. Updated description is "easy" and due date is changed to "05/06/2020"

Test Case: DELETE TASK

Test Case #: 7

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: This will delete a task from your tasks list

Test Case Name: Delete Task

Subsystem: Tasks

Design Date: 04/07/2020

Execution Date:

Page: 7 of 10

Pre-conditions

User must have created a task

Step	Action	Expected System Response	Pas s/ Fail
1	User clicks an existing task name "Exam"	The system shows the detailed task page	P
2	User then clicks "Delete" button	User is directed to the "Task List" page, and the task "Exam" is removed from the list	P

Post-conditions

1. Task "Exam" is deleted from the database

Test Case: ADD CLASS

Test Case #: 8

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: This will Add a class to your list of Classes

Test Case Name: Add Class

Subsystem: Class

Design Date: 04/07/2020

Execution Date:

Page: 8 of 10

Pre-conditions

User must be logged in

Step	Action	Expected System Response	Pas s/ Fail
1	Select the class tab	System displays the class page	P
2	Select the top right '+' symbol to add class.	System displays the add class page	P
3	Enter class name, description, day/time, room (optional), instructor (optional).		P
4	Select 'add class' once necessary fields are filled.	System stores the class information in database	P

Post-conditions

1. A class subject tab is displayed on the screen

Test Case: UPLOAD PICTURE

Test Case #: 9

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: Test uploading images to the server

Test Case Name: Upload Picture

Subsystem: Picture

Design Date: 04/01/2020

Execution Date:

Page: 9 of 10

Pre-conditions

User must have added a class

Step	Action	Expected System Response	Pas s/ Fail
1	User navigates to the classes page.	System displays the classes page.	P
2	User selects the class of choice.	System displays the information of that class on a new page.	P
3	User selects the "+" under the "photos" header.	System opens file/photo library	P
5	User selects picture of a book	System highlights the picture	P
6	User presses Done	System stores and displays the photo under the "photos" header.	P

Post-conditions

1. The user picture is uploaded and saved into the database

Test Case: Edit Class

Test Case #: 10

System: Mobile Device

Designed by: PLANIT

Executed by:

Short Description: Edit the Event and Change Class

Test Case Name: Edit Class

Subsystem: Edit Class

Design Date: 4/29/2020

Execution Date:

Page: 10 of 10

Pre-conditions

The user is logged into their own account

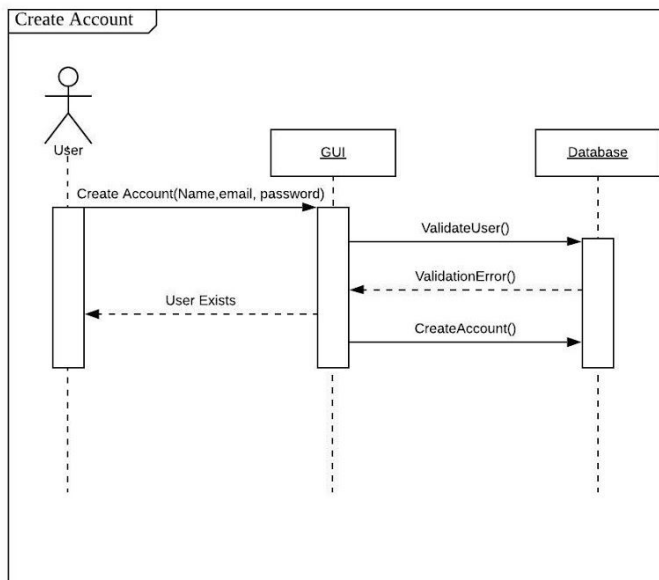
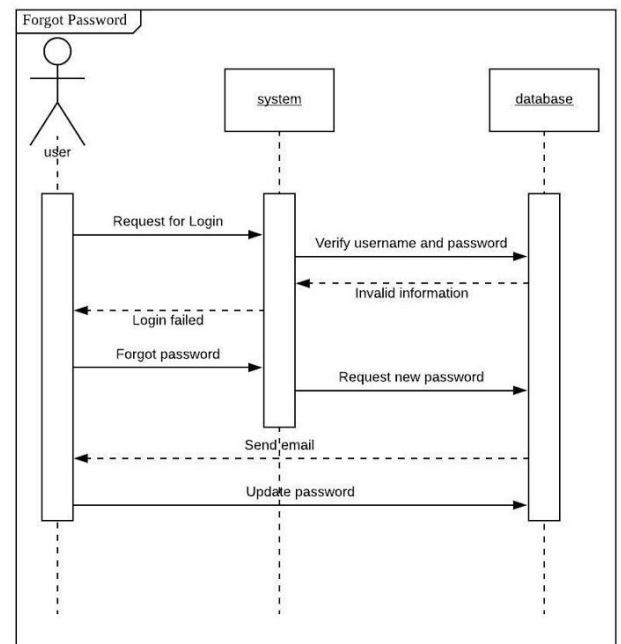
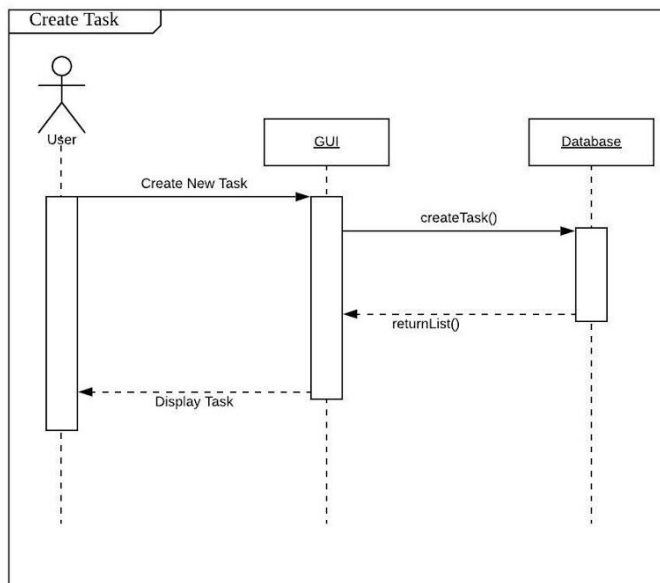
Step	Action	Expected System Response	Pas s/ Fail
1	Click on the Class Page	The system will display the class page	P
2	Click on the Edit Class button	The system will open a new page for editing the class	P
3	Type in 'CPSC 362'	The system will input the description in the text box	P
4	Select a Date 4/20/2020	The system will show the date selected	P
5	Click Save	The system will bring back the class page and display the new class	P

Post-conditions

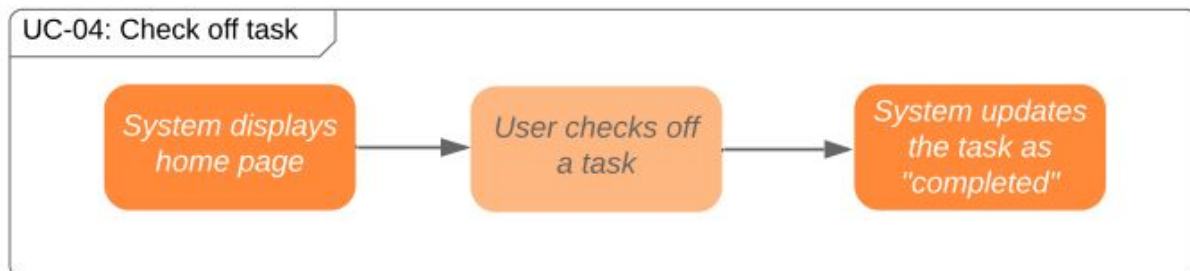
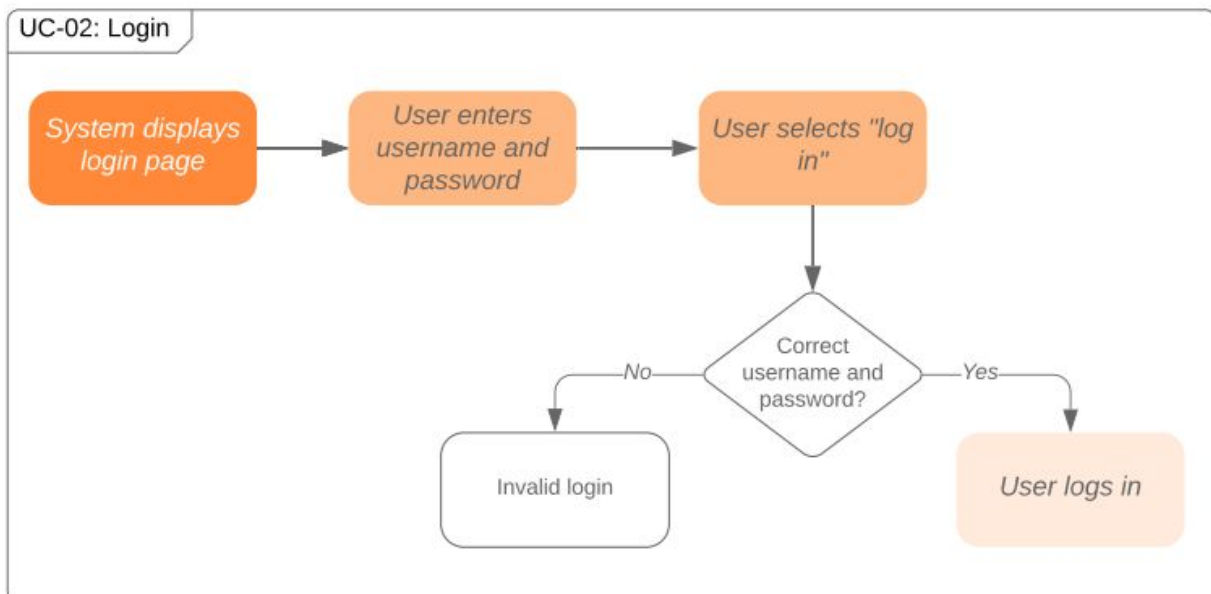
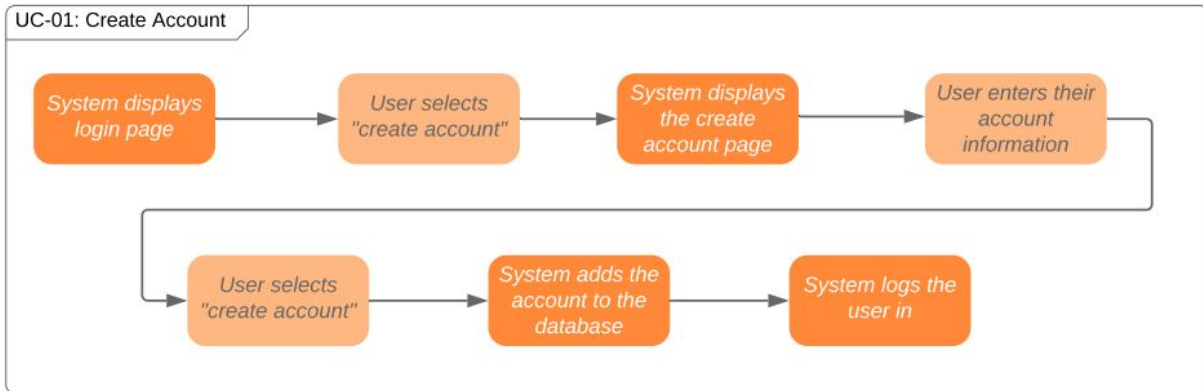
1. The information is saved into the account database for events

3 Sequence Diagrams for Use-Cases

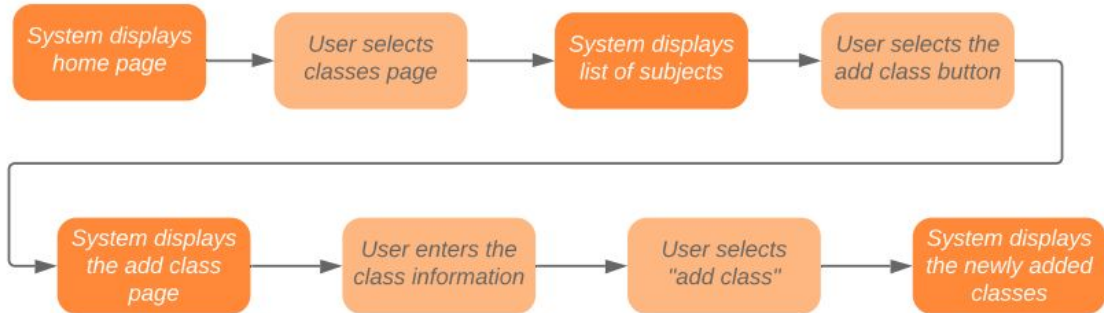
Sequence Diagram
PLANiT



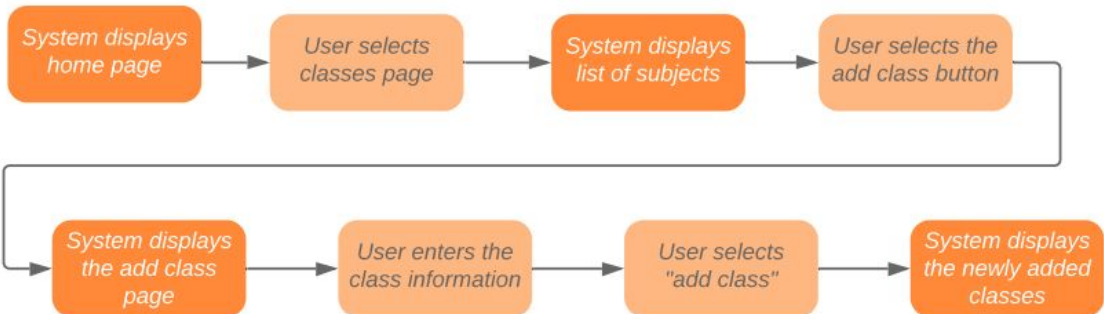
Activity Diagrams



UC-05: Add class subjects



UC-10: View Calendar



Pre-game Planning

5 User - Stories (individual) - Total 25

We each come up with 5 user stories, and below are the reasons why we choose those functions:

Xuanru He: The reason I chose my five functions is that I want to customize my calendar based on the priority, and remind me of important events. Also if possible, I'd like to share it with my friends so that we can arrange the meeting more easily.

Lidel Mendoza: Creating an account and being able to login are the most important tasks for this application in order for me, the user, to be able to see my schedule. After, being able to add, remove events, and upload schedule seemed important because it describes the essence of what the app can do.

Anne Poso: Organization is important for keeping track of both my work and school schedules, so it is essential to have an app with a to-do list, personal customization, and the ability to share my availability with friends so we can coordinate group projects/study groups together.

Brianna Sharpe: I chose "add events" because it is the primary focus of the app. "Reminders" are for the user to be aware of the events/tasks without having to open the app. "Upload class schedule" is a feature to make it easier for student to translate their schedules into our app with a screenshot of their class schedules. "Class colors" should be implemented to indicate the different classes in daily/weekly/monthly views for the user to recognize and distinguish the classes at one glance. "Upload files" would be for important class files to be available in the app instead of going through the school website to access files such as the syllabus or assignment schedule.

Sam Flinkfelt: The main reason I choose my user stories, is because I want to "add events" be able to look at my schedule when I'm asked if I'm free during that time. Schedule important dates especially deadlines, like homework/projects and "be notified". "Add a schedule of my classes", because they change every semester, and every day classes are different during the week. "Add Alarms" to remind me of the little things I usually forget, like submitting attendance at a certain time everyday, street cleaning to move my car and even waking me up. Lastly, be able to "view" all my scheduled events and calendar so I can get an idea of what's going on.

Staging/Grooming

For the 1st sprint we focused on the foundation of the application, which starts with creating an account and logging in. Then we have tasks; tasks allow the user to keep track of events/assignments as in checking off tasks when completed, deleting irrelevant tasks, and being able to view tasks/events. Lastly, features include calendar view, uploading class schedule, writing notes under events, and setting reminders.

1st Sprint: 5 User-Stories

We chose the following because they are fundamental to our application:

- Create account
- Login
- Add tasks
- Check off task
- Add class subjects

We prioritized the main features of a digital planner, but also chose simpler features to implement because it is our first time using the programming language dart.

Poker Game:

We use poker game to estimate and distribute work. For each user story, we discussed the estimates and high and low estimates are explained. We repeated the process until estimates converge. Below are the records of how we did the poker game:

- 1.) Create account
 $3+3+2+5+5 = 3.6\text{hrs or } 4 \text{ hours}$
- 2.) Login
 $1+1/2+2+3+3 = 1.9\text{hr or } 2 \text{ hours}$
- 3.) Add task
 $5+5+3+5+8 = 5.2 \text{ or } 5 \text{ hours}$
- 4.) Check off tasks
 $3+2+5+3+2 = 3 \text{ hours} \sim \text{rethought to take } 5 \text{ hours}$
- 5.) Add class subjects
 $13+10+8+10+8 = 50 \text{ hours}$

Development Process

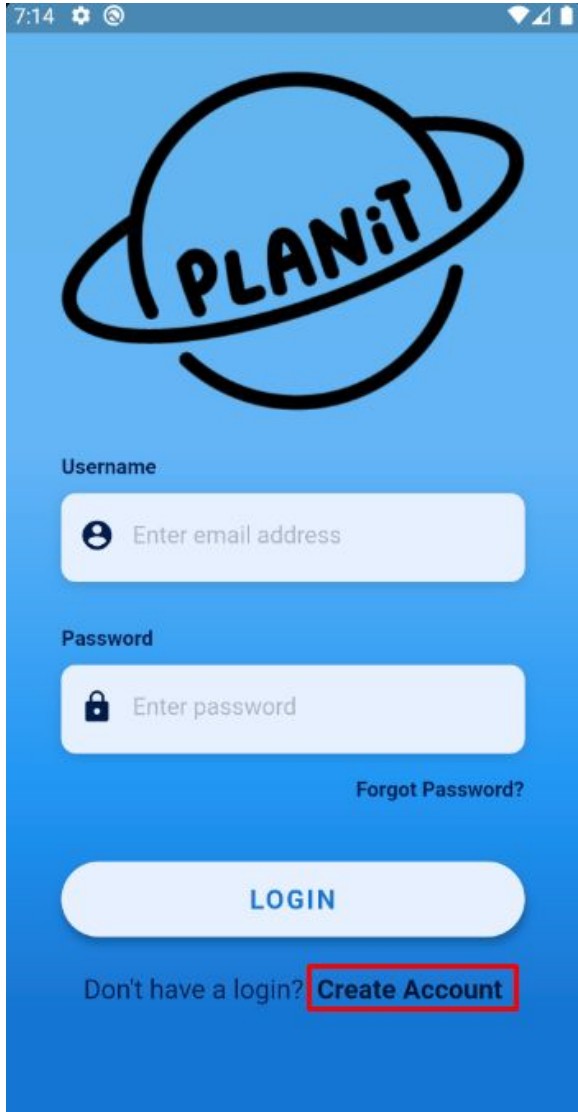
We are using Android Studio as our integrated development environment (IDE) with the Flutter software development kit (SDK). The programming language that we are using is Dart, a language developed by Google. For our project's backend services, we use Firebase, a mobile and web application development platform. These services include a cloud-hosted database and user authentication methods.

We use the Firebase Authentication service for our register, login, and forgot password functionalities in order to track the identity of each user and save their data in the cloud to provide a personalized experience for each user of the app. We also use Firebase's Cloud Firestore database to store user data on the mobile app.

User Manual

Skip steps 1 & 2 if the user already has an existing account.

1. To create an account, press “Create Account”

A screenshot of the PLANiT mobile application's login screen. The background is blue. At the top, there is a logo consisting of a black circle with a ring around it, and the text "PLANiT" in a bold, black, sans-serif font. Below the logo, there are two input fields: "Username" with a placeholder "Enter email address" and a person icon, and "Password" with a placeholder "Enter password" and a lock icon. To the right of the password field is a link that says "Forgot Password?". At the bottom, there is a large white button with the text "LOGIN". Below the login button, there is a link that says "Don't have a login?" followed by a white button with the text "Create Account". The "Create Account" button is highlighted with a red rectangle.

7:14

PLANiT

Username

Enter email address

Password

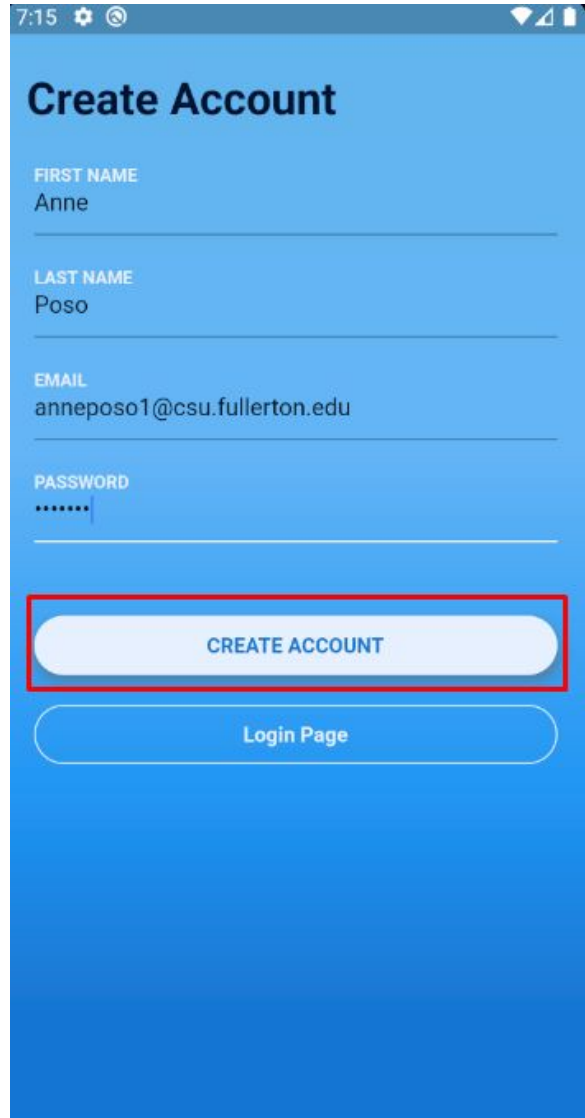
Enter password

Forgot Password?

LOGIN

Don't have a login? **Create Account**

2. Enter first name, last name, email, and password

A screenshot of the PLANiT mobile application's "Create Account" screen. The background is blue. At the top, there is a title "Create Account" in a bold, black, sans-serif font. Below the title, there are four input fields: "FIRST NAME" with the text "Anne", "LAST NAME" with the text "Poso", "EMAIL" with the text "anneposo1@csu.fullerton.edu", and "PASSWORD" with a placeholder "*****". Below the password field, there is a large white button with the text "CREATE ACCOUNT". Below the "CREATE ACCOUNT" button, there is a white button with the text "Login Page". The "CREATE ACCOUNT" button is highlighted with a red rectangle.

7:15

Create Account

FIRST NAME

Anne

LAST NAME

Poso

EMAIL

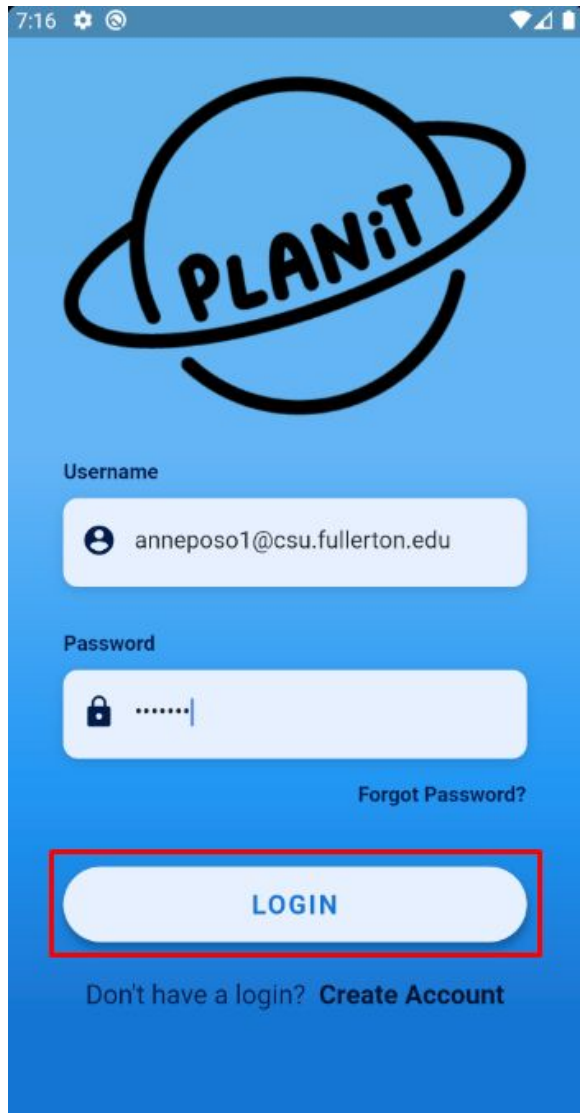
anneposo1@csu.fullerton.edu

PASSWORD

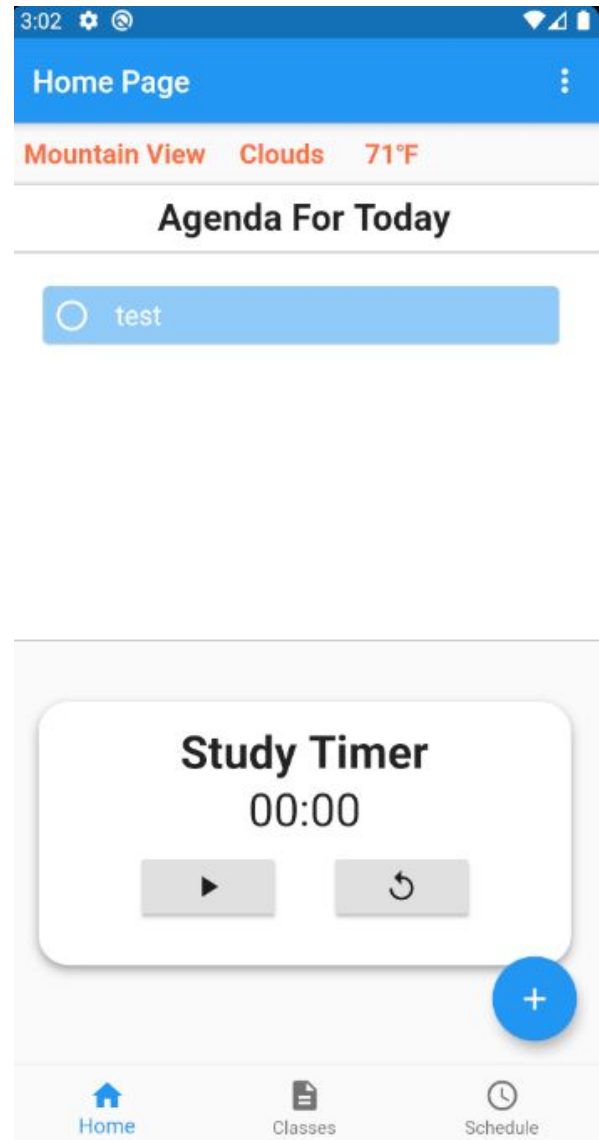
CREATE ACCOUNT

Login Page

3. Enter email address and password, then press “Login”

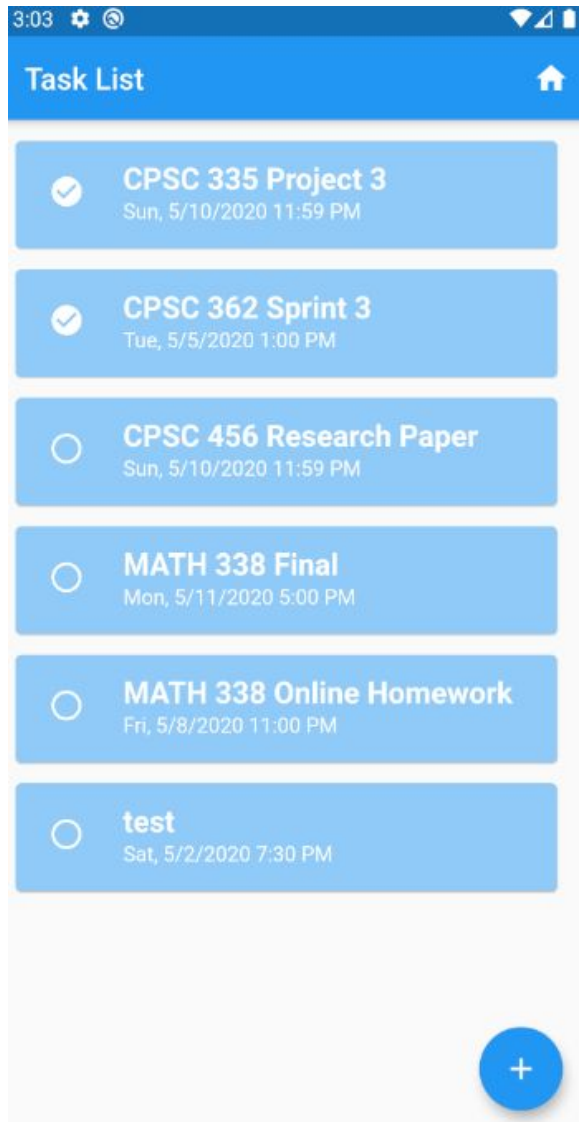


4. System redirects user to app's homepage

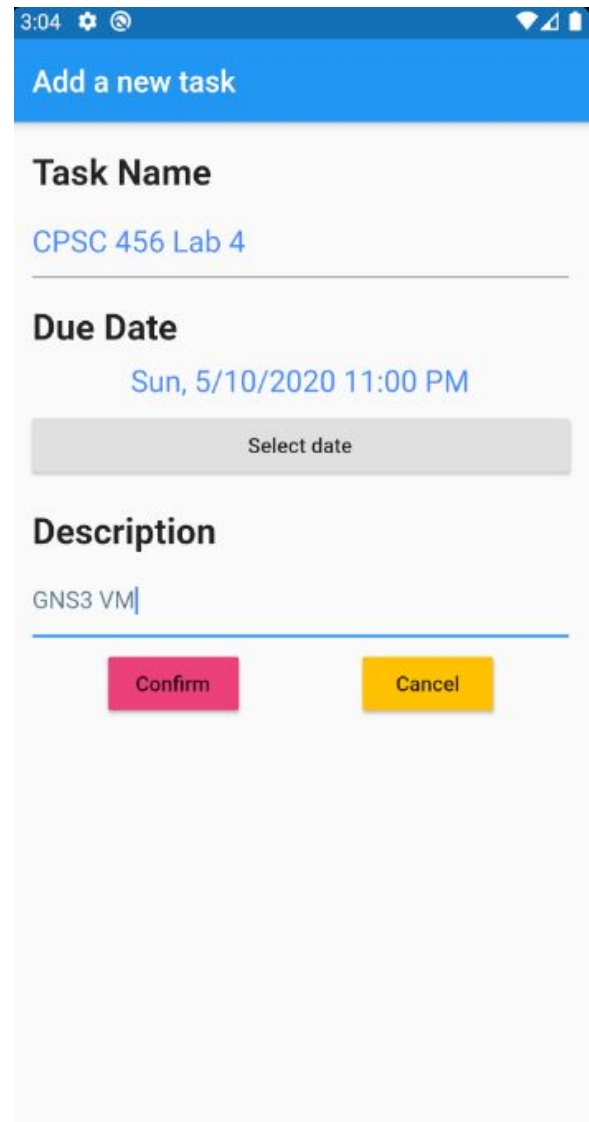


How To Add/Check Off A Task

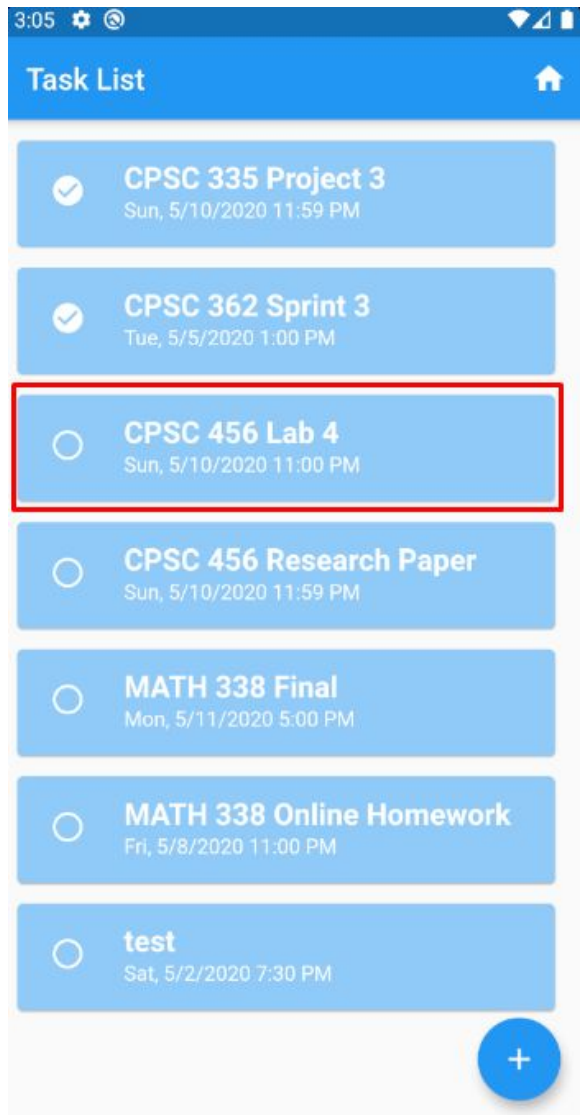
1. Press the “+” button to add tasks



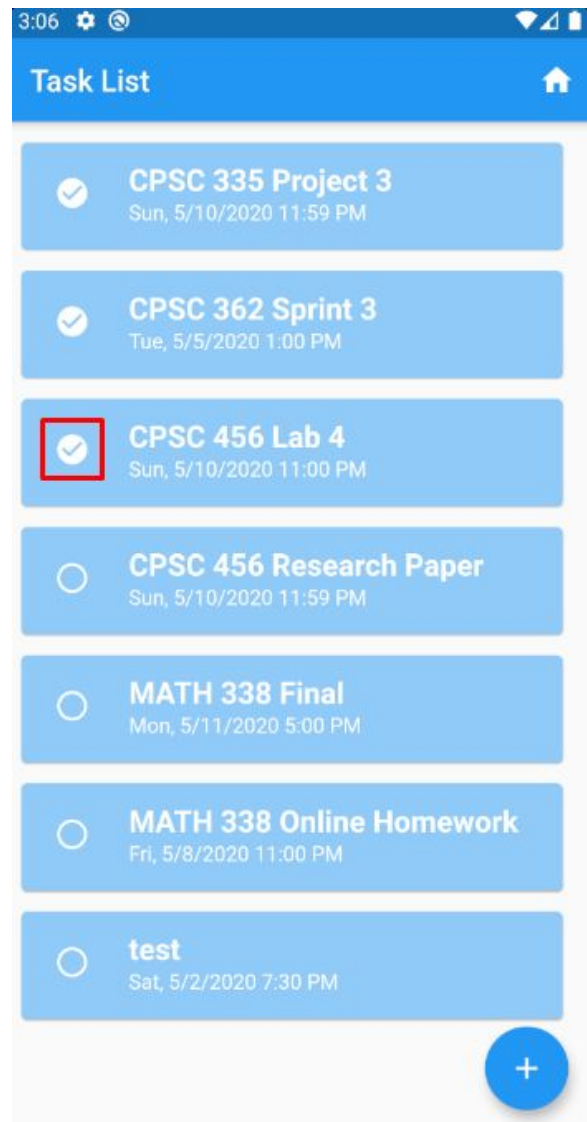
2. Input task name and description, select due date, and then click 'Confirm' to submit



3. System displays new task in list view

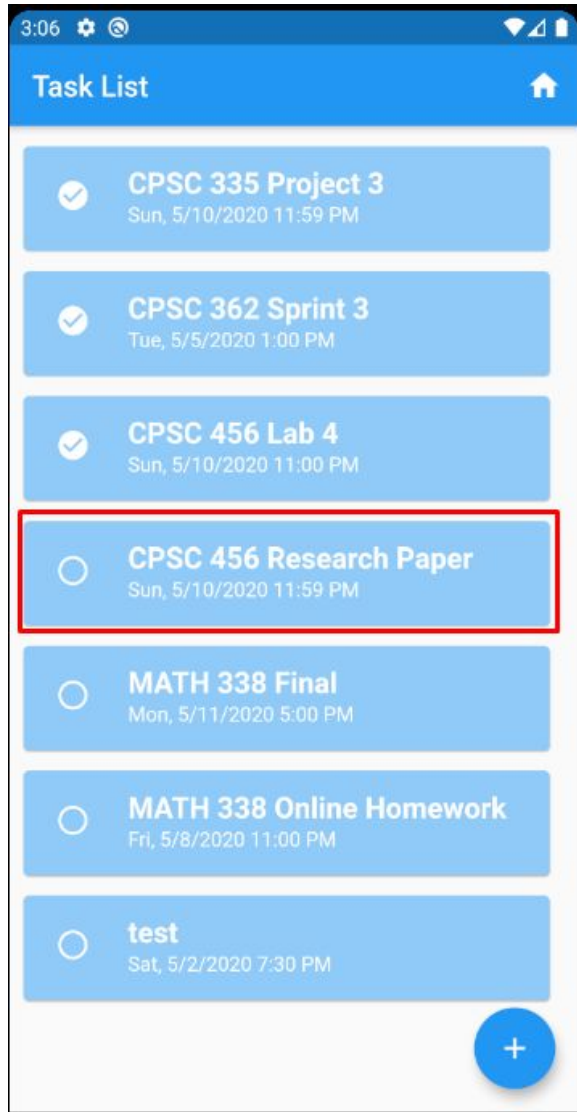


4. Tap the circle to next to the task name to mark as completed

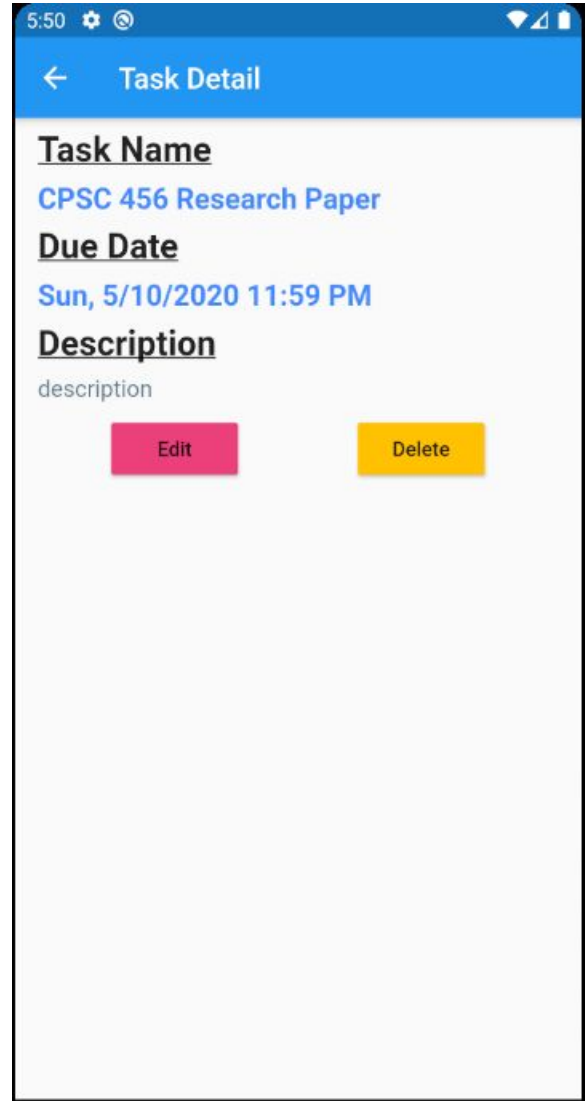


How To View/Edit Task

1. Tap the task name to view task details



2. Systems displays task details in a new page, and then press 'Edit' to edit your task, or press 'Delete' to delete the task from task list



3. Change your due date and add description here, then click 'Confirm' to update or 'Cancel' to go back

6:28

← Edit task

123

Due Date

2020-04-06

Select date

Description

123

Confirm Cancel

>

1 2 3 4 5 6 7 8 9 0

q w e r t y u i o p

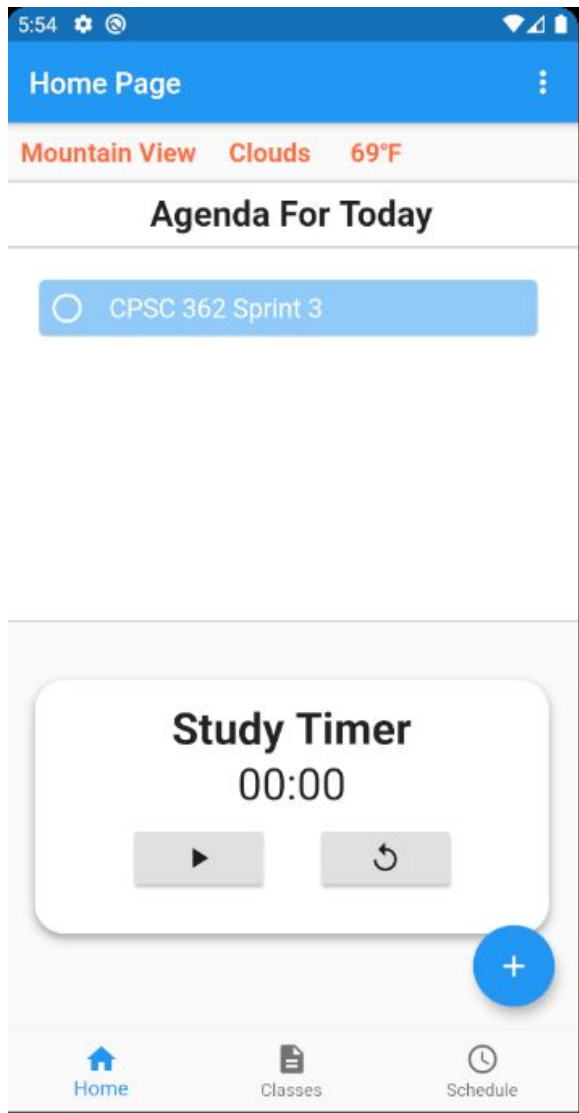
a s d f g h j k l

⬆ z x c v b n m ⬆

?123 , 😊 . ✓

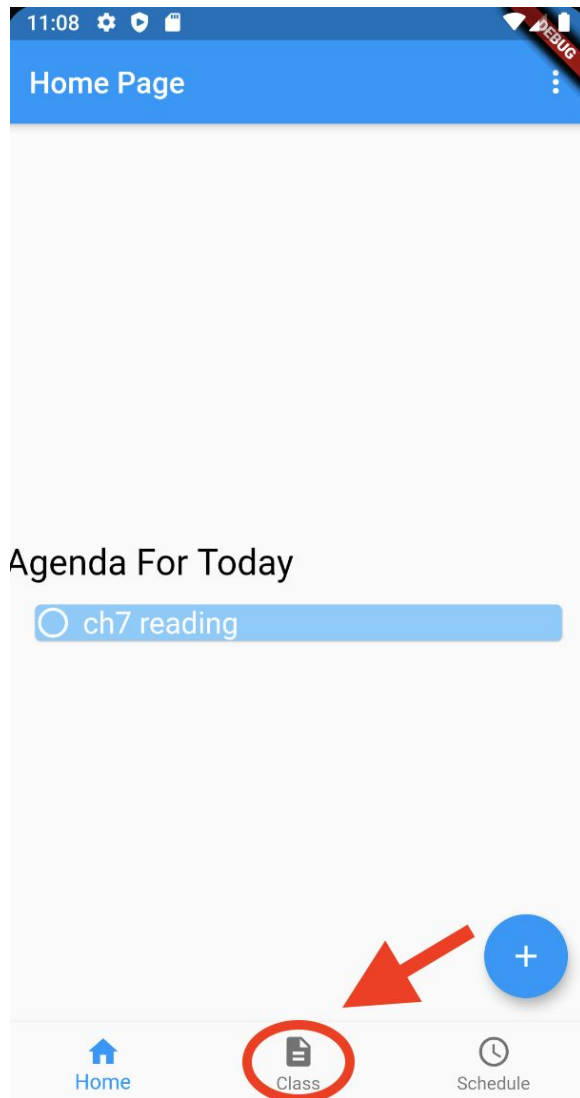
How To View Agenda

Tap task name to view task detail and you will be directed to the 'Task Detail' page

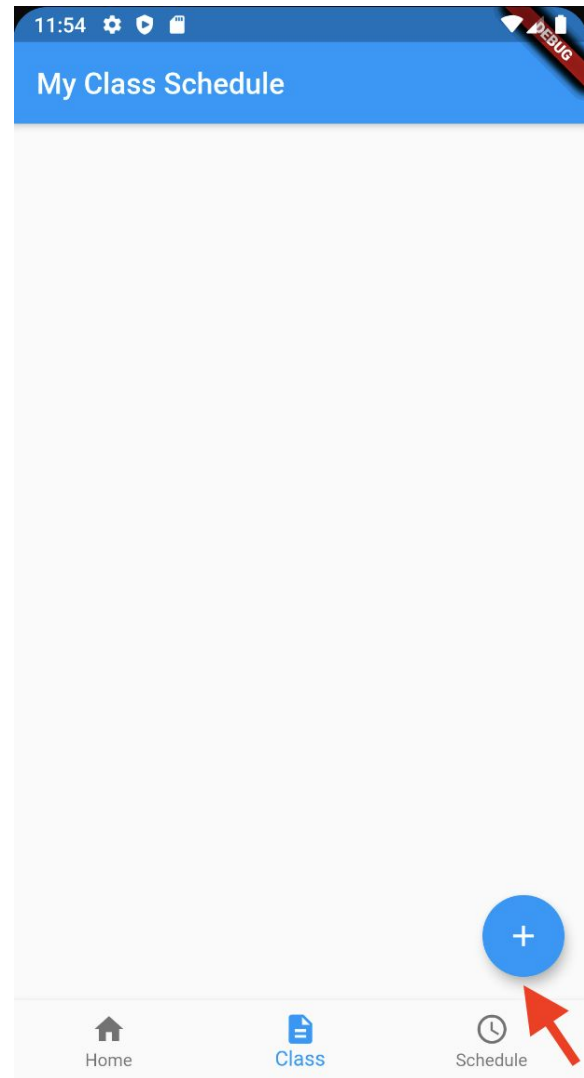


Add A Class Subject

1. Select 'Classes' from the bottom bar.



2. Click on the “plus” icon



3. Fill in the class information fields.

11:06

← Add a new class

Course Name
Enter Course Name...

Description
ex: 2 midterms and no final...

Start Date
2020-04-03
Select date

Class Begins:
ex: 11am or 1pm...

thanks | | we

q w e r t y u i o p

a s d f g h j k l

↑ z x c v b n m

?123 , 😊 . ✓

3. Then, click on the “confirm” button.

11:07

← Add a new class

Description
1midterm 1 final and 1 main project

Start Date
2020-04-03
Select date

Class Begins:
11am

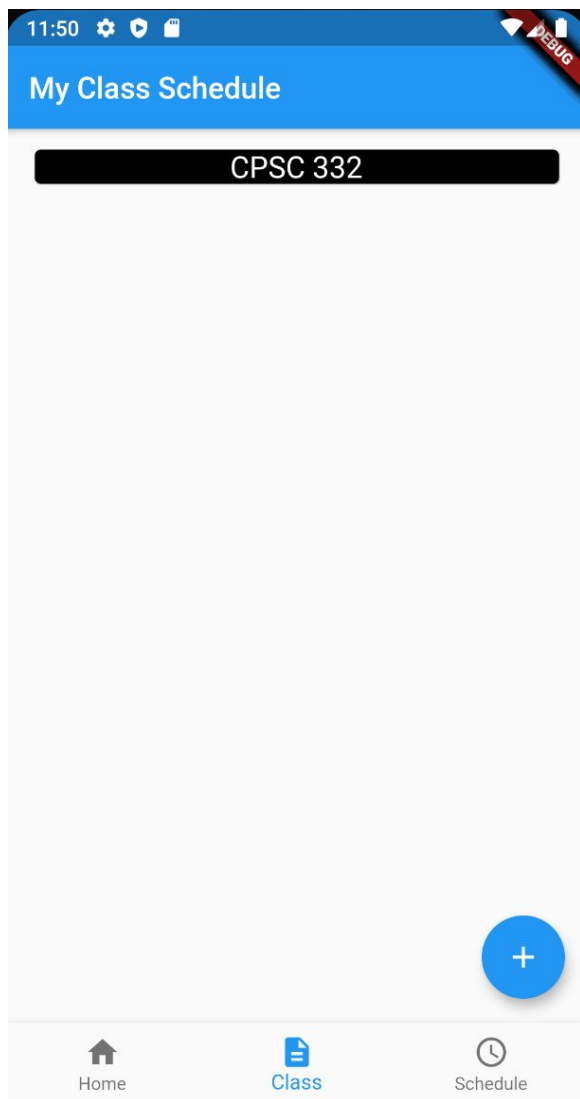
Class Ends:
12pm

Room number:
CPSC 109A

Instructor:
Mr. Xang

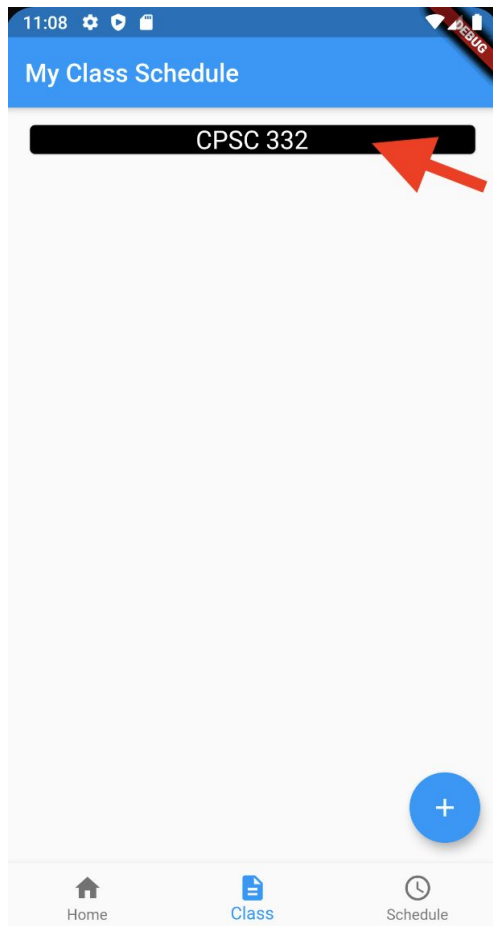
Confirm Cancel

4. Your class subject will then be added to “My Class Schedule” page

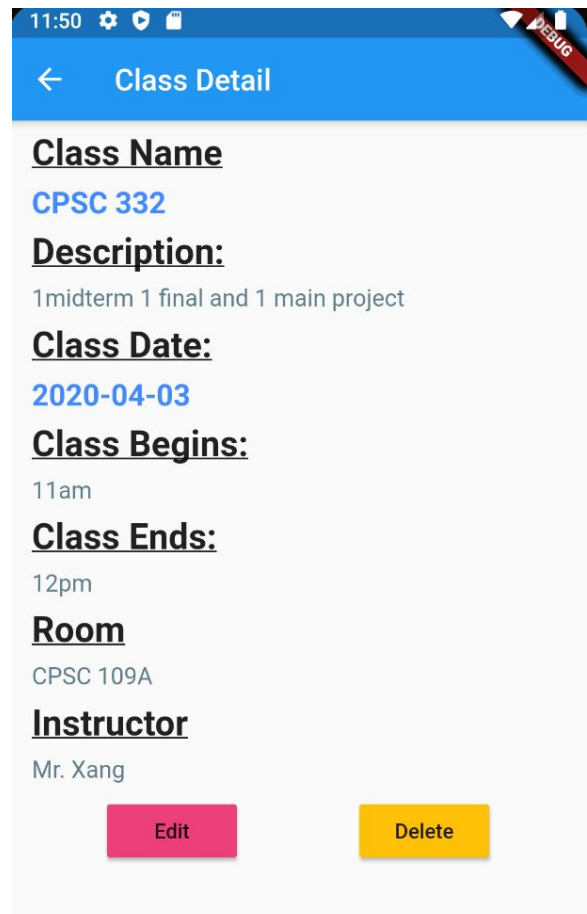


View your class information

1. Click on your subject tab

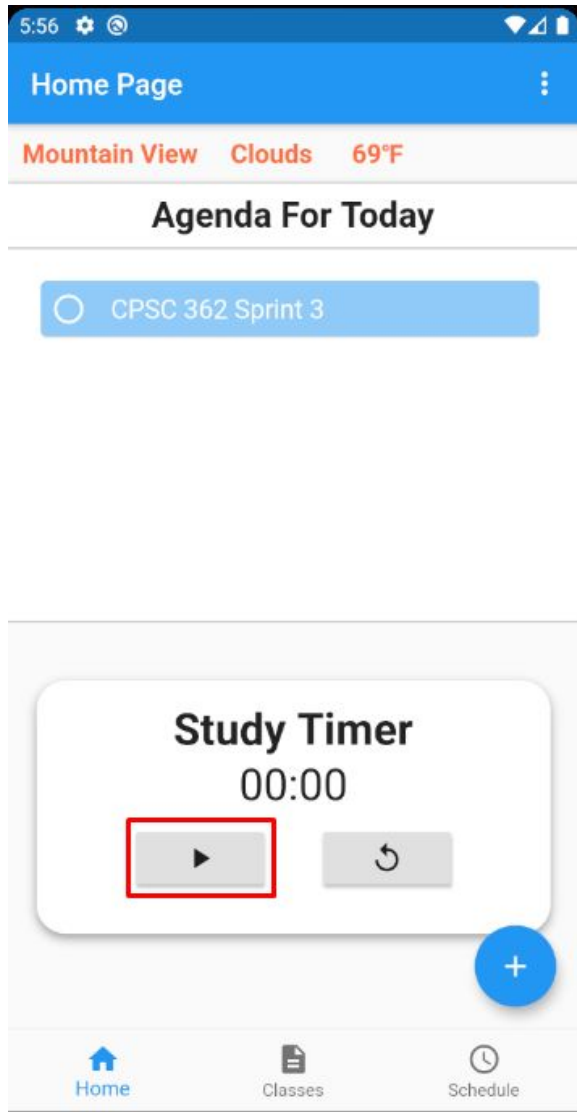


3. Displays your class detail

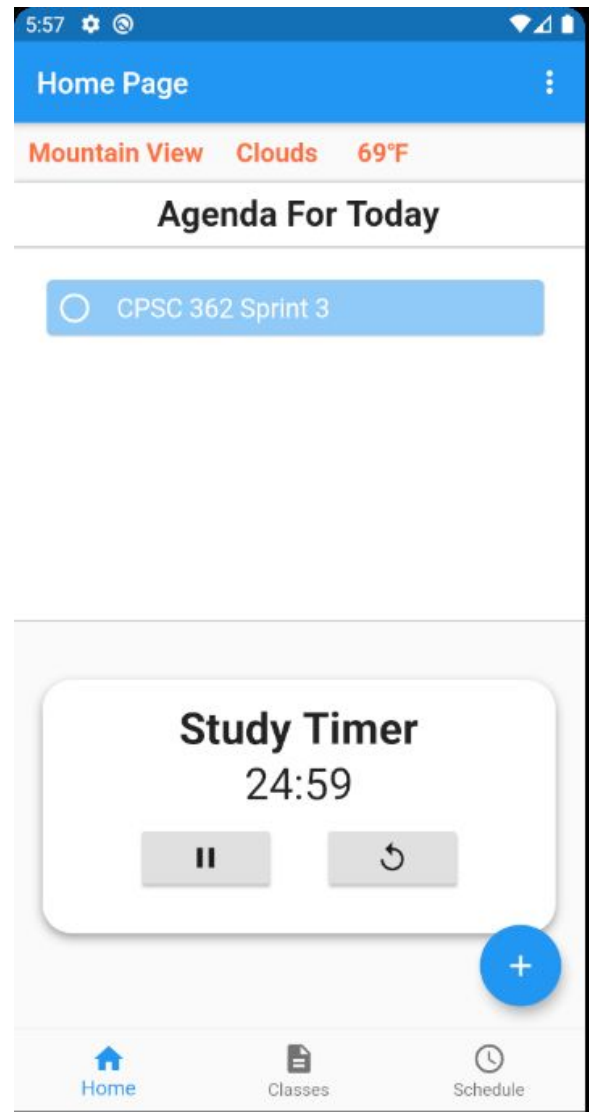


Study Timer

1. Press play button to start timer countdown



2. Study timer should begin counting down from 25:00 minutes



References

N/A