SC2006 Software Engineering

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Lab#3 Deliverables

Lab Group: SCEC E3

Group Name: Mission: Passable

Group members:

CHUA QIN DI
NICHOLAS CHANG CHIA KUAN
REDDIPALLI SAI DHARSHAK SATHAVAHANA
SHARMA HIMANSHU
VONA TANISHA
WU YIQING

Table of Content

1. Use Case Descriptions

Refer to SC2006 UseCaseDiagram.pdf for the Use Case Diagram

Use Case ID:	UC-DB-1		
Use Case Name:	Register		
Created By:	Chua Qin Di	Last Updated By:	Chua Qin Di
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Register** allows new users to create an account in our system database. To create an account, users must input Username, Email, Password
Preconditions:	1. Username must not be taken 2. Email must not be linked to existing account 3. Passwords must meet requirements: must be at least 8 characters long, containing 1 capital letter and 1 number.
Postconditions:	
Priority:	High
Frequency of Use:	Once
Flow of Events:	1. User inputs required information: Username, Email, Password. 2. System validates the information. 3. System creates the account. 4. System updates account information in database. 5. System directs the user to the homepage.
Alternative Flows:	Username taken 1. System displays “User account already exists” 2. System prompts user to input a different username 3. Continue from main flow step 1 Email linked to existing account 1. System displays “Email linked to existing account” 2. System prompts user to log in to existing account or register account 3. If log in, System directs the user to the Login page 4. If register account, continue from main flow step 1 Password does not meet requirements 1. System displays “Password does not meet requirements” 2. System prompts user to re-enter their password. 3. Continue from main flow Step 1.
Exceptions:	

Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-DB-2		
Use Case Name:	Login		
Created By:	Chua Qin Di	Last Updated By:	Chua Qin Di
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Login** allows users to enter the app and use the features intended for registered users of the app. To login to the system, the user must input their username and the corresponding password.
Preconditions:	User must have an existing account in the database
Postconditions:	System displays the main homepage
Priority:	High
Frequency of Use:	Occasional
Flow of Events:	1. User enters username and password on the login page. 2. System validates the user's username and password. 3. System shows the main homepage.
Alternative Flows:	Invalid Username 1. System cannot find the username in the database. 2. System displays an error message. 3. System prompts the user to input username and password or register an account. 4 Use case resumes at main flow step 1 or Register account page Invalid Password 1. Password does not match for the username. 2. System displays an error message. 3. System prompts user to input username and password. 4. Use case resumes at main flow step 1.
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-PF-1		
Use Case Name:	Manage Profile		
Created By:	Wu Yiqing	Last Updated By:	Wu Yiqing
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Manage Profile** allows users to view and update personal information.
Preconditions:	1. User must have an existing account in the database. 2. The user must have logged into the app.
Postconditions:	
Priority:	High
Frequency of Use:	Occasional
Flow of Events:	1. User navigates to profile settings. 2. User selects the option to update profile. 3. User inputs updated information. (e.g., name, password and email) 4. User submits updated information. 5. System validates the updated information. 6. System updates the new profile into database. 7. System displays new profile for user.
Alternative Flows:	
Exceptions:	Invalid Data Entry 1. User enters an invalid information format. 2. System displays error message and rejects update profile request. 3. User must rectify the invalid format before proceeding. User cancels update profile request. 1. User chooses to exit the update profile page. 2. App returns to home page.
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-FR-1		
Use Case Name:	Manage Friends		
Created By:	Vona Tanisha	Last Updated By:	Vona Tanisha
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Manage Friends** allows users to add existing users as friends or allows users to accept another user's Friend Request or remove friends from their friend list.
Preconditions:	Friend must be an existing user in the database
Postconditions:	Request will be sent to friend, pending his approval Friend shows up on Friend List
Priority:	Medium
Frequency of Use:	Occasional
Flow of Events:	1. User inputs another existing user's username to add as friend 2. System displays "Friend request sent" 3. User chooses to press Accept or Reject Friend Request 4. User removes friend from Friend List
Alternative Flows:	Cannot add yourself as Friend 1. System displays error message and prompt user to input another username Username does not exist 1. System displays error message and prompt user to input another username Already your friend 1. System displays error message and prompt user to input another username
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-CO-1		
Use Case Name:	Manage Course		
Created By:	Chua Qin Di	Last Updated By:	Chua Qin Di
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Manage Course** allows users to add, remove or view their current courses into the database.
Preconditions:	
Postconditions:	
Priority:	Medium
Frequency of Use:	Occasional
Flow of Events:	1. User selects Add Course, View Course or Remove Course a. Add Course allows users to add their courses into their course list by inputting course ID b. View Course allows users to view the information of the selected course c. Remove Course allows users to remove the course from their course list
Alternative Flows:	
Exceptions:	
Includes:	Add Course, View Course, View Friend's Course, Remove Course
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-TS-1		
Use Case Name:	Manage Task		
Created By:	Chua Qin Di	Last Updated By:	Chua Qin Di
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**Manage Task** allows users to add, view or remove their tasks on the database
Preconditions:	
Postconditions:	
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	1. User selects Add Task, View Task or Remove Task a. Add Task allows users to add tasks to complete onto the their task list by inputting the description of the task and the deadline b. View Task allows users to view the information of the selected task c. Remove Task allows users to remove the task from the task list
Alternative Flows:	1. User can mark task as completed
Exceptions:	
Includes:	Add Task, View Task, View Friend's Task, Remove Task
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-TB-1		
Use Case Name:	View Timetable		
Created By:	Vona Tanisha	Last Updated By:	Vona Tanisha
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**View Timetable** allows users to view their personal timetable and their Friends timetables
Preconditions:	Must be friends to view other users timetable
Postconditions:	
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	1. User can Choose to view personal timetable 2. User can choose a Friends timetable to view
Alternative Flows:	1. If the user is not connected to the internet a. The system notifies the user that they will not be able to view a friend's timetable till internet connection is restored.
Exceptions:	
Includes:	View personal timetable, View Friends Timetable
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-NT-1		
Use Case Name:	Notifications		
Created By:	Wu Yiqing	Last Updated By:	Wu Yiqing
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User, Database
Description:	**Notifications** are alerts received by users when the system detects upcoming deadlines for tasks, and messages sent to the user from other users.
Preconditions:	1. Upcoming deadlines for task 2. Incoming friend requests 3. Friend notifies user
Postconditions:	
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	1. System detects that a message is sent to the user 2. The app displays an alert on screen for the user with the information
Alternative Flows:	E1. User is not connected to internet 1. System delays notification until user is connected to the internet
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

Use Case ID:	UC-PG-1		
Use Case Name:	ProgressBar		
Created By:	Chua Qin Di	Last Updated By:	Chua Qin Di
Date Created:	3 Feb 2025	Date Last Updated:	25 March 2025

Actor:	User
Description:	**ProgressBar** is a graphical UI to show users the completion rate of their current tasks
Preconditions:	1. Have at least one current task
Postconditions:	
Priority:	High
Frequency of Use:	Frequent
Flow of Events:	1. The app displays a graphical progress bar to show the percentage of tasks to be completed
Alternative Flows:	No ongoing tasks 1. The app will display a message indicating that there are no tasks to be completed at the moment
Exceptions:	
Includes:	
Special Requirements:	
Assumptions:	
Notes and Issues:	

2. Class diagram of entity classes

Refer to SC2006 ClassDiagram.pdf on github repository: softwarelab3/2006-SCEC-E3/lab3

3. Sequence diagrams

Refer to SC2006 SequenceDiagram.pdf on github repository: softwarelab3/2006-SCEC-E3/lab3

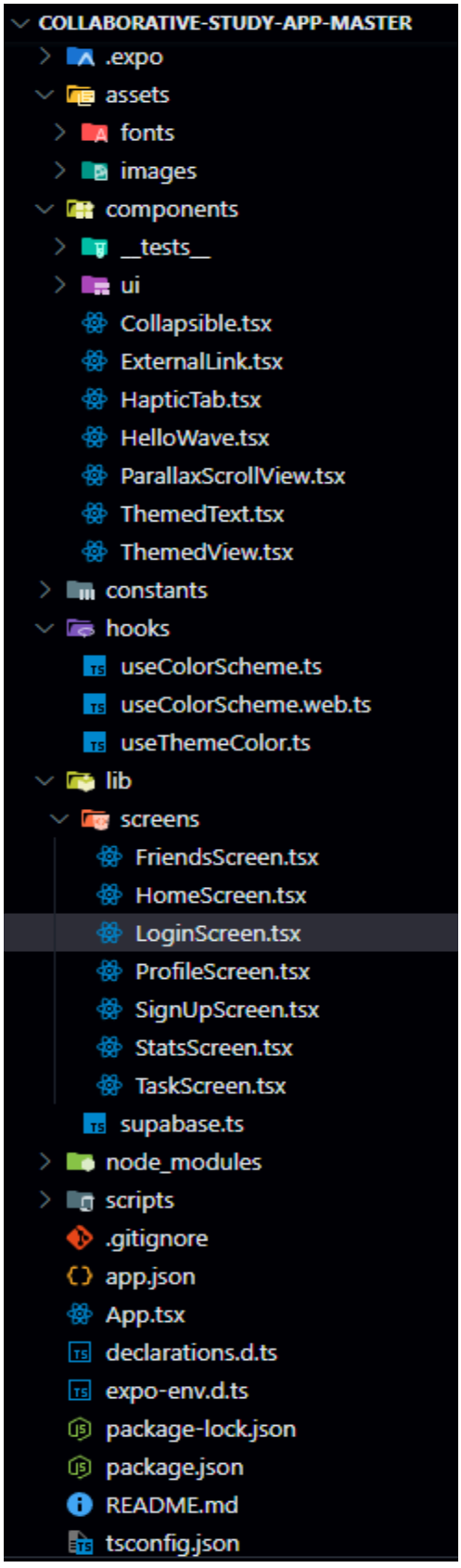
4. Dialog map

Refer to SC2006 DialogMap.pdf on github repository: softwarelab3/2006-SCEC-E3/lab3

5. System Architecture

Refer to SC2006 System Architecture.pdf on github repository: softwarelab3/2006-SCEC-E3/lab3

6. Application Skeleton



.expo:
Expo is the developer tool our group is using to run our mobile app.
This folder contains Expo-specific configuration files for running and building the app.

assets:
This folder stores static files used in the app such as custom fonts for UI styling and images such as logos & background images.

components:
This folder houses reusable UI components used across different application parts, avoiding repetition and improving code modularity.

constants:
This folder contains app-wide constant values.such as theme color.

hooks:
This folder contains custom React hooks that are stored here to encapsulate reusable logic, enhancing the modularity of the codebase.

libs:
This folder contains the core app logic and services. **screens/** contains all the main pages of the app seen by the user.
Supabase is the database used for our app.
supabase.ts initializes the Supabase client with our API key and project URL.

node_modules:
This folder holds all dependencies installed via npm, a package manager for Node.js

scripts:
This folder stores utility scripts for automation such as environment setup

app.json:
This file contains the main configuration for the mobile app which includes the app name, description, permissions and deep linking

App.tsx
This file is the main entry point of the React Native app.

declarations.d.ts & **expo-env.d.ts**
These declaration files add type definitions for non-code imports to prevent TypeScript errors when importing assets.

package-lock.json & **package.json:**
These files Lists dependencies, scripts, and metadata for the project.

README.md
This file contains the project documentation for contributors and users. It includes the setup instructions and the description of our app.

tsconfig.json:
This file configures TypeScript behavior such as setting compiler options and defining paths for imports