SC2006

PROJECT SKYSCAPE

TEAMWON

Li Yaohao

Lim Zhi Yong

Kieran Liow Jiun Kiat

Jin Ying

Kiersten Yeo Shu Xian

Yoon Sang Won

Table of Contents

1. INTRODUCTION

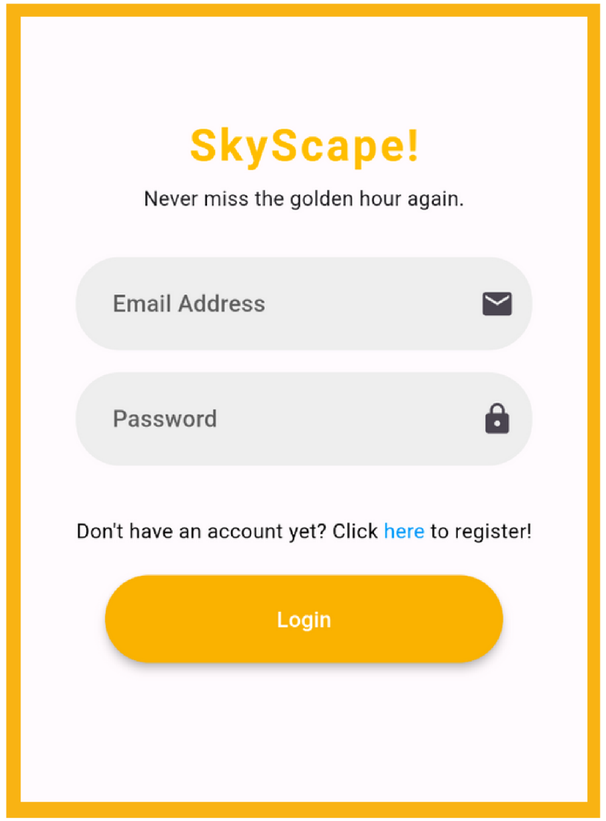
NEVER MISS THE GOLDEN HOUR AGAIN.

About Skyscape

Introduction

Purpose

To predict the quality of sunsets on any given day and bring everyone closer to nature’s spectacle.



Target Audience

For casual users who occasionally enjoy sunsets, and also photography enthusiasts and nature lovers.

About Skyscape

Introduction

Motivation

- Skyscape is modelled after social media applications whereby users are able to interact with other users.

- Drawing inspiration from activity-tracking apps like "Strava," Skyscape places a strong emphasis on community involvement.

- This communal aspect is instrumental in retaining users.

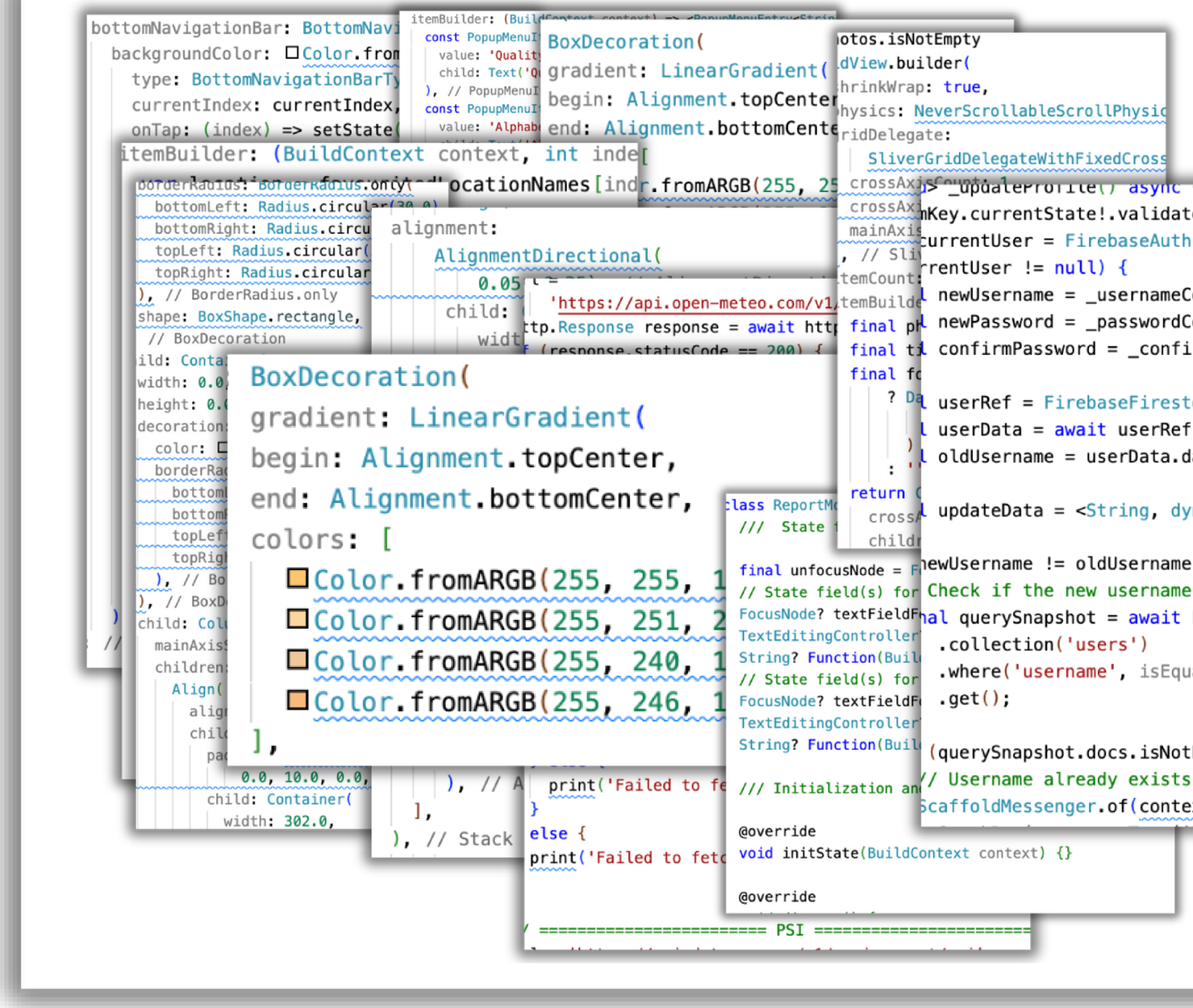
Community Stories

STRAVA

About Skyscape

Flutter

- Extensive widget library and reactive framework streamline UI development
- Hot reload feature for faster development
- Declarative UI approach



2. USE CASE & FUNCTIONALITIES

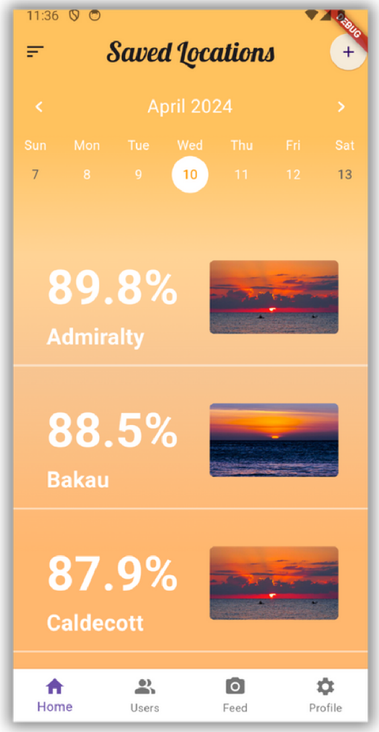
NEVER MISS THE GOLDEN HOUR AGAIN.

Skyscape

Brief Overview

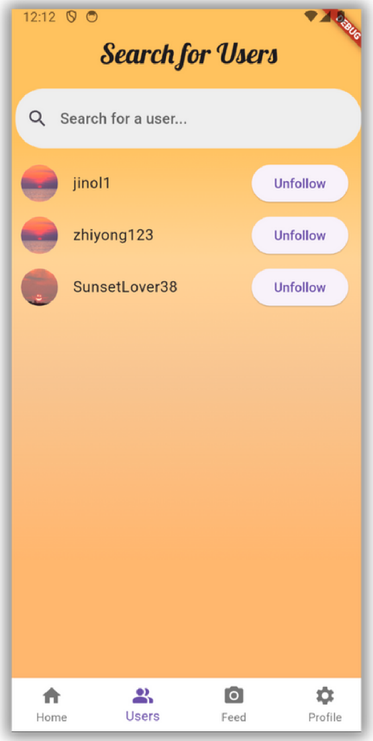
Quality of sunsets

Users can view the predicted quality of sunsets of any location on a chosen date on the home page



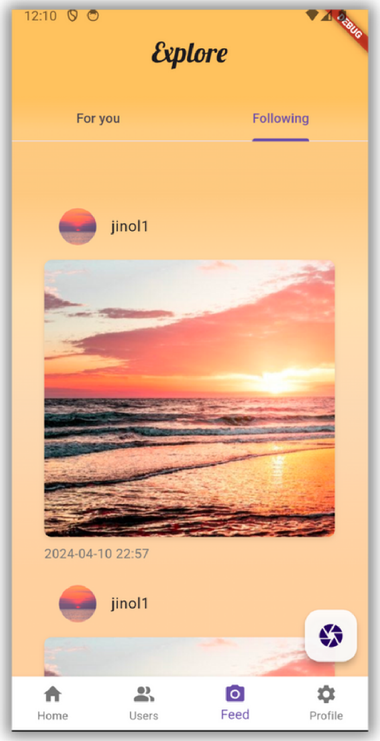
Follow Users

Users can follow other users to view the sunset images that they have uploaded



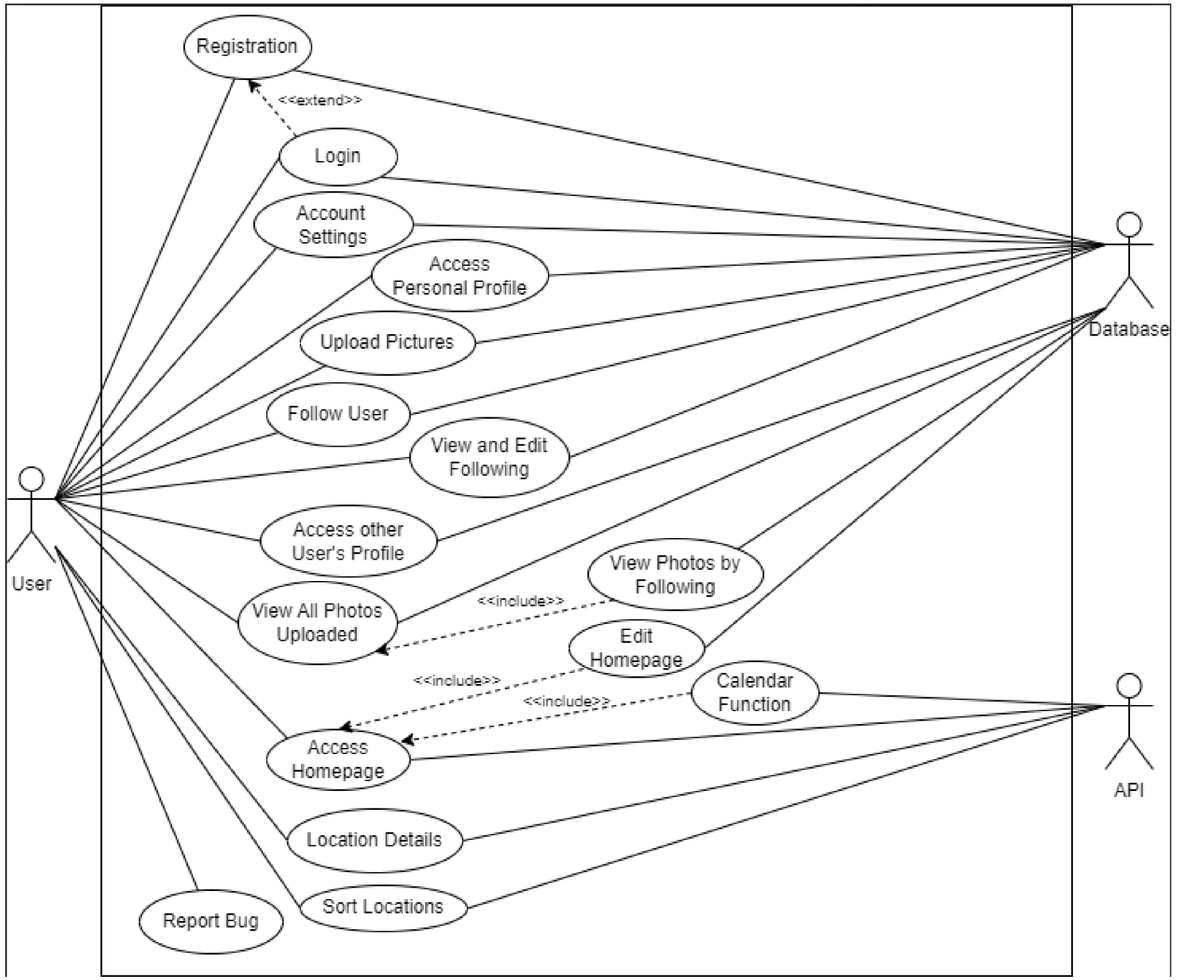
Upload Pictures

Users can upload pictures of sunsets that they have taken for memory and to share with other users



Skyscape

Complete Use Case Diagram



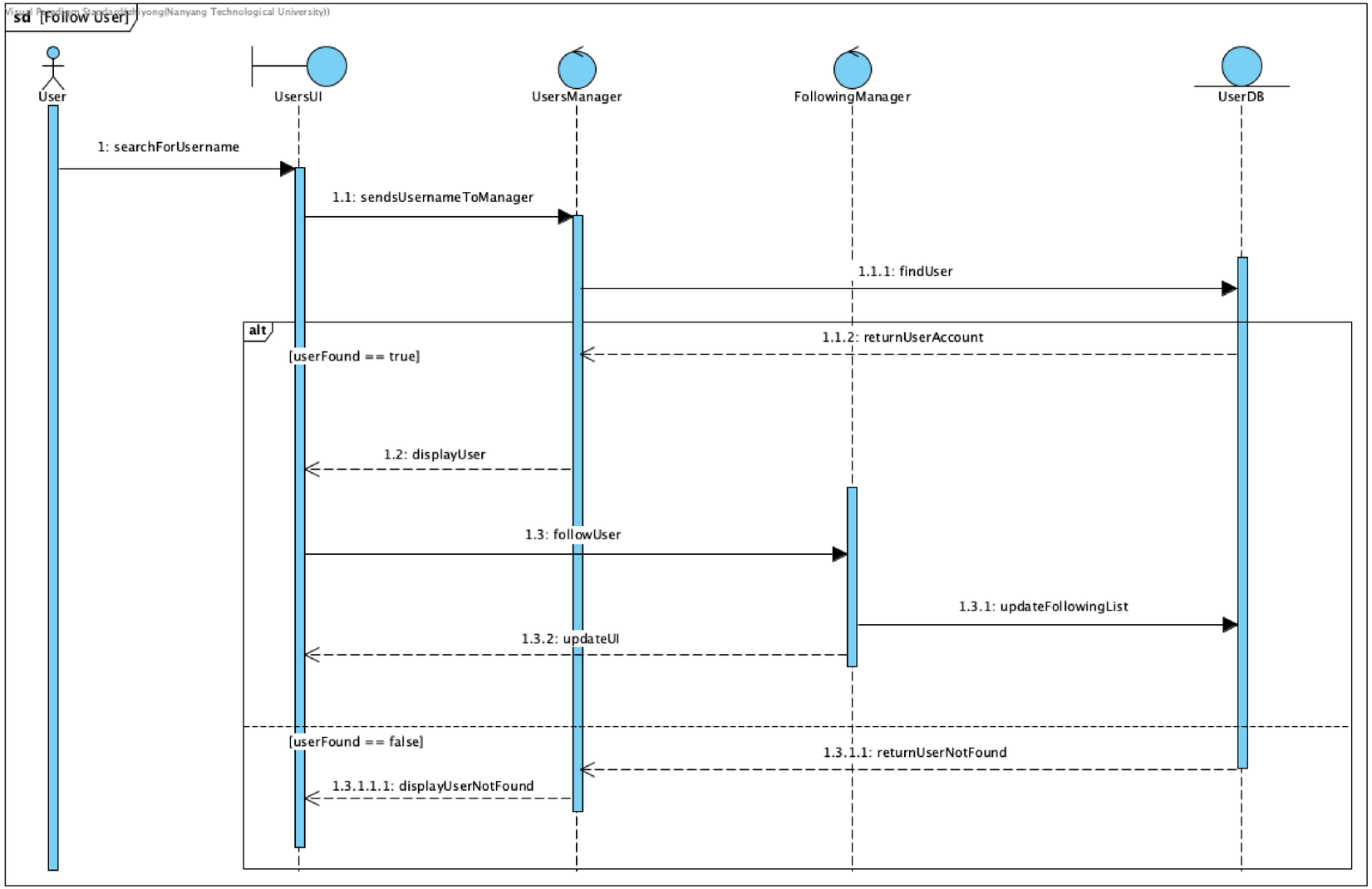
Use Case Diagram

Main Functionalities

- Login/Register
- View Homepage
- View Location Details
- View Feed
- Upload Pictures
- Follow User
- View Other’s Profile
- Report Bugs

Skyscape

Sequence Diagram

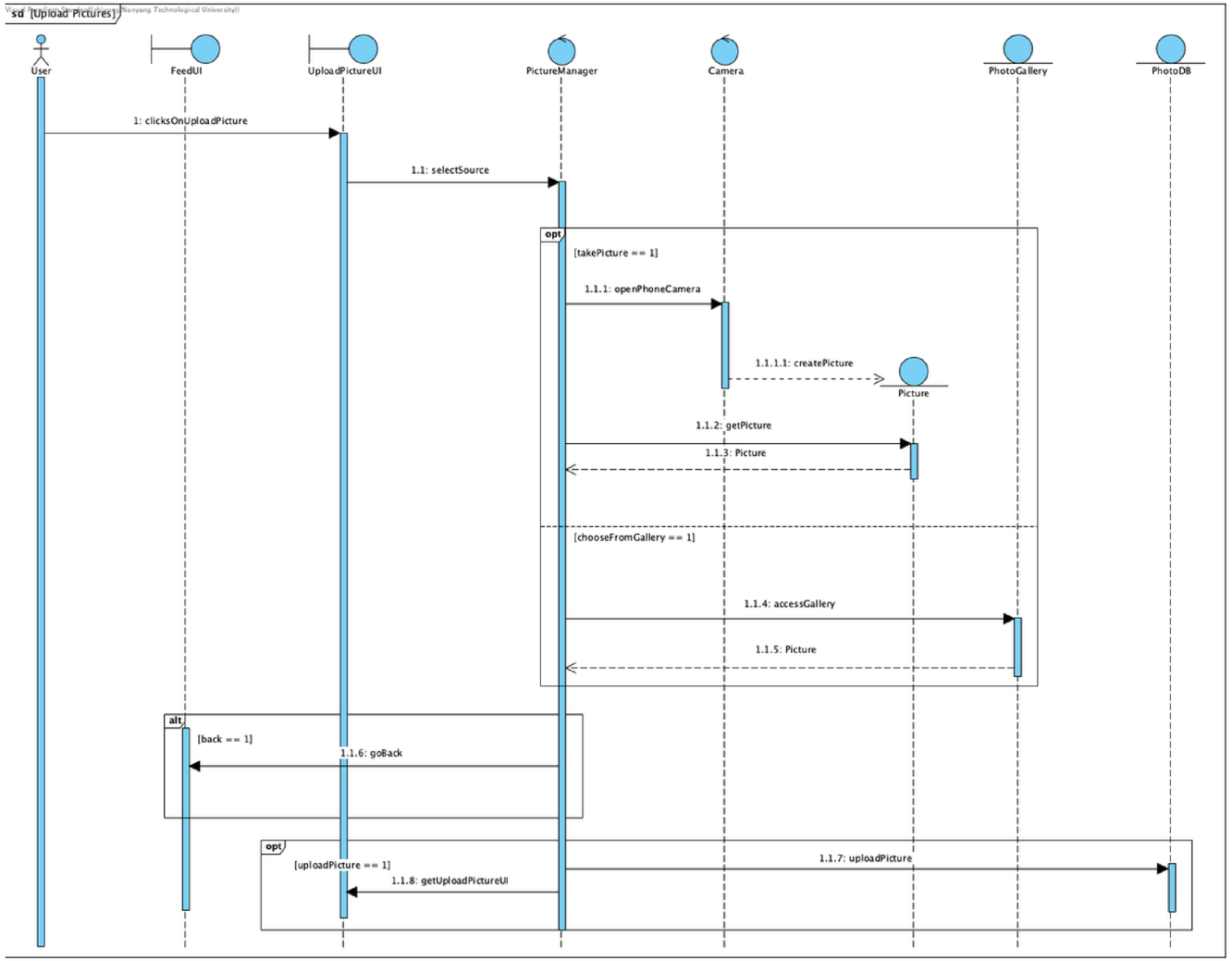


Follow User Functionality

User is able to search the username of other users he wishes to find and follow that user

Skyscape

Sequence Diagram

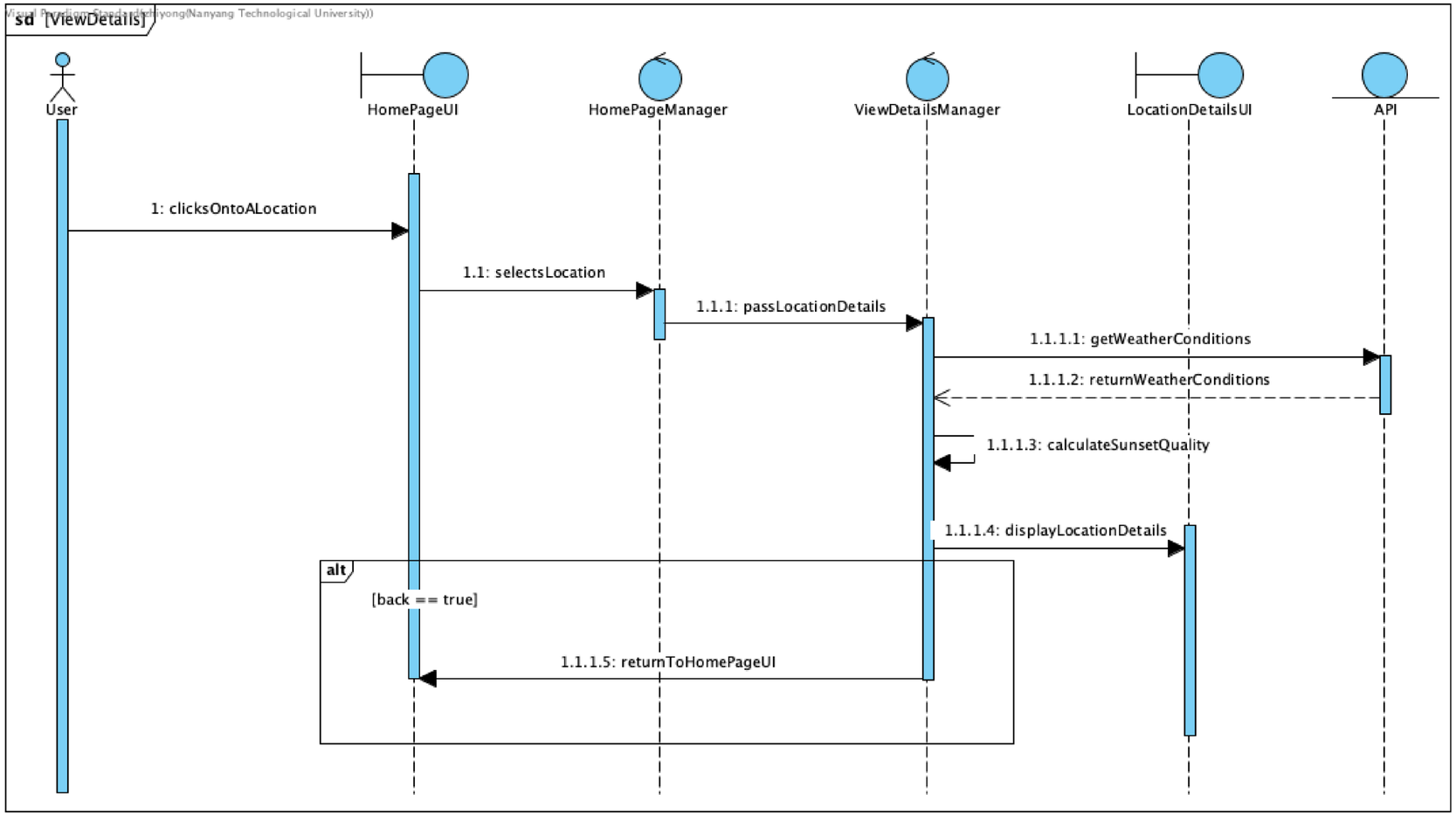


Upload Pictures Functionality

User is able to select a photo from his gallery or take a photo using the phone camera and upload the photos to the feed for others to see

Skyscape

Sequence Diagram



View Detail Functionality

User is able to view the weather details of the location he chooses as well as the calculated sunset quality with the information obtained from API

3. GOOD SE PRACTICES

NEVER MISS THE GOLDEN HOUR AGAIN.

Good SE Practices

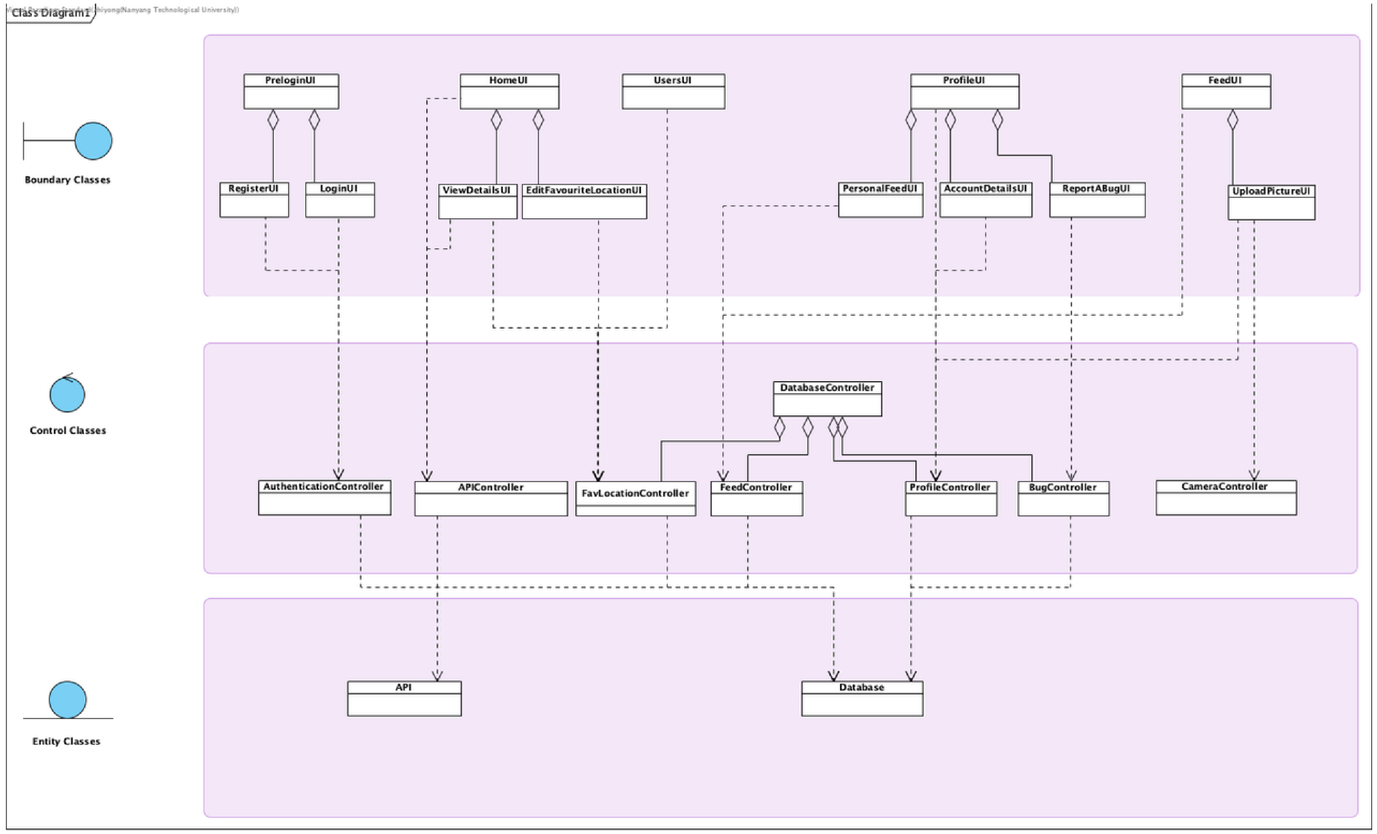
- 3 Layered Architecture
- DRY - Don’t Repeat Yourself
- SRP - Single Responsibility Principle
- Black box Testing

3 Layered Architecture

A well-established software application architecture that organizes applications into three logical and physical computing tiers.

Benefits

1. Accelerated Development
2. Improved Scalability
3. Loose Coupling



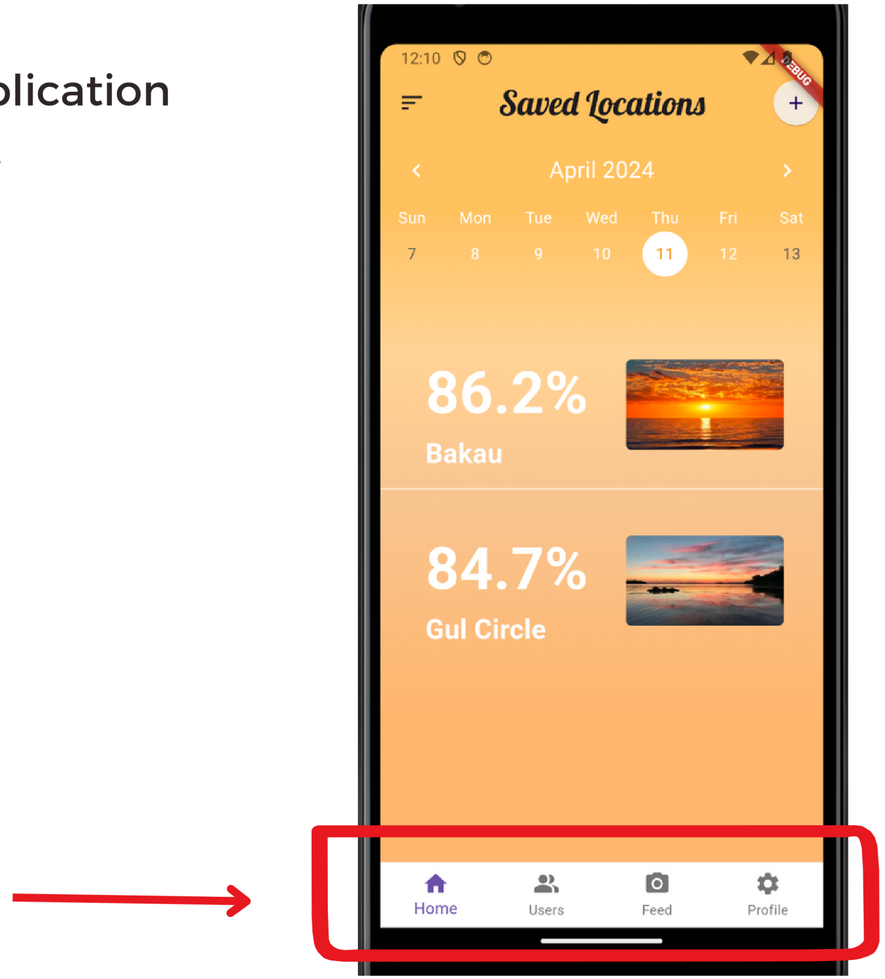
DRY - Don’t Repeat Yourself

DRY

A principle of software development that aims at **reducing the repetition** of patterns and code duplication in favor of abstractions and avoiding redundancy. (Muldrow, 2020)

Benefits

1. Improved Code Readability
2. Better Performance
3. Better Maintainability



Skyscape

DRY - Don't Repeat Yourself

Build the body around the fixed template rather than include the template on individual bodies.

```
        index: currentIndex,
        children: [
          buildHomeScreen(),
          SearchUsers(),
          FeedPage(),
          ProfileMainWidget(),
        ],
      ), // IndexedStack
    ), // Container
  // ),
  bottomNavigationBar: BottomNavigationBar(
    backgroundColor: Color.fromARGB(255, 255, 255, 255),
      type: BottomNavigationBarType.fixed,
      currentIndex: currentIndex,
      onTap: (index) => setState(() => currentIndex = index),
      items: const [
        BottomNavigationBarItem(
          icon: Icon(Icons.home),
          label: 'Home',
        ), // BottomNavigationBarItem
        BottomNavigationBarItem(
          icon: Icon(Icons.people_alt_rounded),
          label: 'Users',
        ), // BottomNavigationBarItem
        BottomNavigationBarItem(
          icon: Icon(Icons.camera_alt),
          label: 'Feed',
        ), // BottomNavigationBarItem
        BottomNavigationBarItem(
          icon: Icon(Icons.settings),
          label: 'Profile',
        ), // BottomNavigationBarItem
      ],
    ), // BottomNavigationBar
```

Implementation for the Bottom Navigation Bar

Skyscape

SRP - Single Responsibility Principle

SRP

A class should have only **one reason to change,** in other words, a class should have a single responsibility or concern. (Kant, 2023)

Benefits

1. Simplifies Debugging
2. Improved Code Ownership

```
Future<List<String>> getFavouritedLocations() async {
  DocumentSnapshot snapshot = await userCollection.doc(uid).get();
  if (snapshot.exists) {
    List<dynamic> locations = snapshot.get('favouritedLocations');
    return locations.map((location) => location.toString()).toList();
  }
  return [];
}

Future<void> sortFavouritedLocations(List<String> locationList) async {
  return await userCollection.doc(uid).update({
    'favouritedLocations': locationList,
  });
}

Future<String> findUsernameFromUID(String userid) async {
  DocumentSnapshot snapshot = await userCollection.doc(userid).get();
  if (snapshot.exists) {
    String username = snapshot.get('username');
    return username;
  }
  return "No user found.";
}

Future<String> findUIDFromUsername(String username) async {
  QuerySnapshot snapshot = await userCollection.where('username', isEqualTo: username).get();
  if (snapshot.docs.isNotEmpty) {
    String userid = snapshot.docs.first.id;
    return userid;
  }
  return "No user found";
}
```

Skyscape

Equivalence class testing

Black box Testing

A good software engineering practice is to test the software using equivalence class testing to **find bugs**

Benefits

1. Directs tester to a very small subset of test inputs that can highly likely find bugs

Login Functionalities

Test ID	Test Case Description	Input: Email	Input: Password	Expected Output	Actual Output	Pass/Fail
EC1	Verify successful login with valid email and password	sunsetlover72 @gmail.com	SUNSETlover72	User is logged in successfully	User is logged in successfully	Pass
EC2	Verify error message when email is invalid	sunsetlover72	SUNSETlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC3	Verify error message when password is invalid	sunsetlover72 @gmail.com	sunsetlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC4	Verify error message when email is unregistered	sunsetlover73 @gmail.com	SUNSETlover73	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC5	Verify error message when email is empty		SUNSETlover72	Error message “Enter an email”	Error message “Enter an email”	Pass
EC6	Verify error message when password is empty	sunsetlover72 @gmail.com		Error message “Enter a valid password”	Error message “Enter a valid password”	Pass

Skyscape

Equivalence class testing

Login Functionalities

Test ID	Test Case Description	Input: Email	Input: Password	Expected Output	Actual Output	Pass/Fail
EC1	Verify successful login with valid email and password	sunsetlover72 @gmail.com	SUNSETlover72	User is logged in successfully	User is logged in successfully	Pass
EC2	Verify error message when email is invalid	sunsetlover72	SUNSETlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC3	Verify error message when password is invalid	sunsetlover72 @gmail.com	sunsetlover72	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC4	Verify error message when email is unregistered	sunsetlover73 @gmail.com	SUNSETlover73	Error message “Invalid email or password”	Error message “Invalid email or password”	Pass
EC5	Verify error message when email is empty		SUNSETlover72	Error message “Enter an email”	Error message “Enter an email”	Pass
EC6	Verify error message when password is empty	sunsetlover72 @gmail.com		Error message “Enter a valid password”	Error message “Enter a valid password”	Pass

Testing only one invalid input for each test case, allowing us to learn the error handling or behavior for each type of invalid input

4. GOOD TRACEABILITY PRACTICES

NEVER MISS THE GOLDEN HOUR AGAIN.

Skyscape

Traceability Practices

Version traceability

Usage of Github/Git to help in maintaining a history of changes to code files.

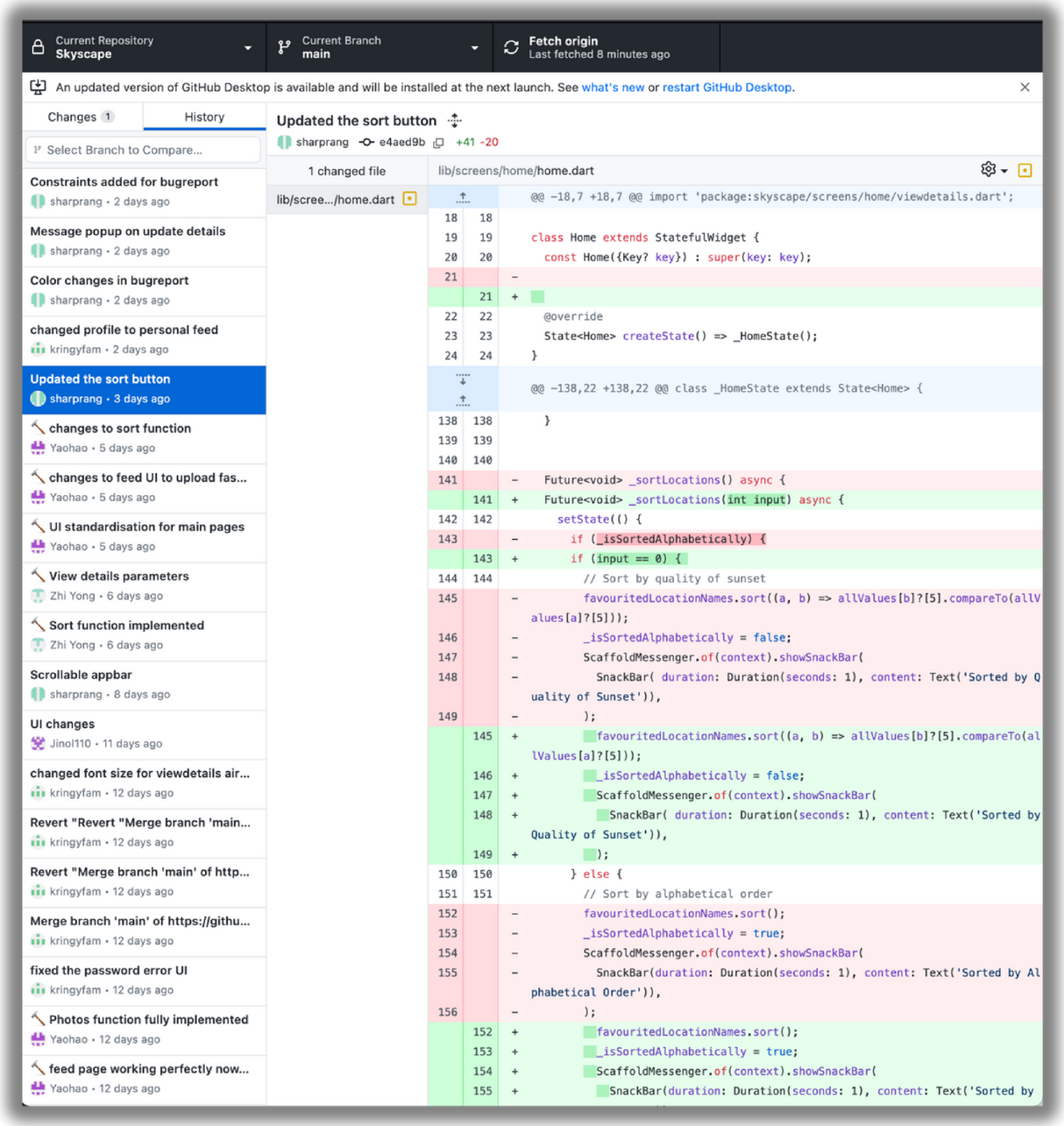
Design traceability

Code followed UI mockups and documentations done previously

Requirements traceability

Code followed closely to Use case descriptions, as well as functional and non functional requirements

Skyscape



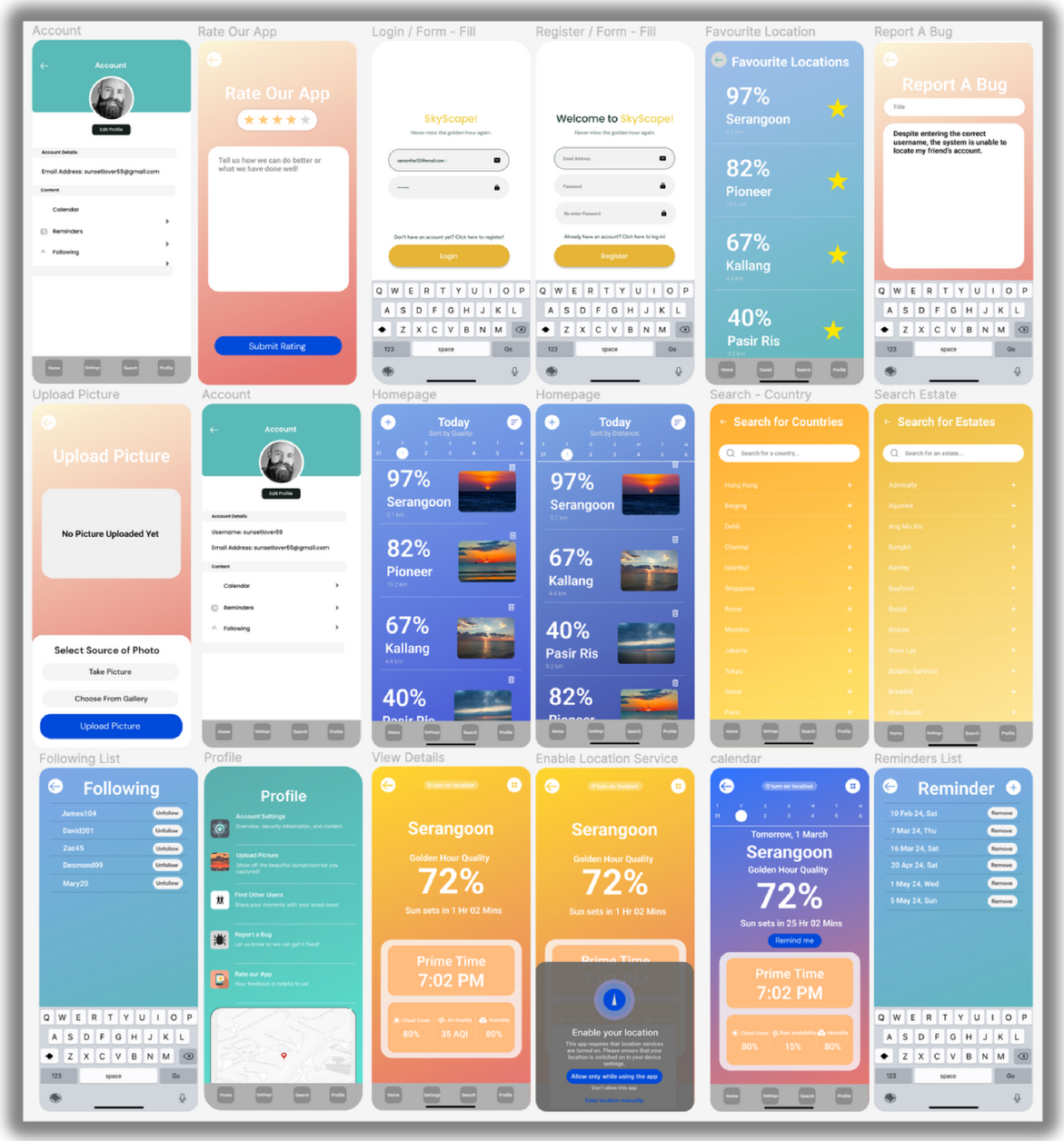
Traceability

Version Traceability

GitHub/Git helps with version control in traceability by maintaining a history of changes to code and related artifacts.

This enabled our team to track the changes in design, functions and implementations over time.

Skyscape



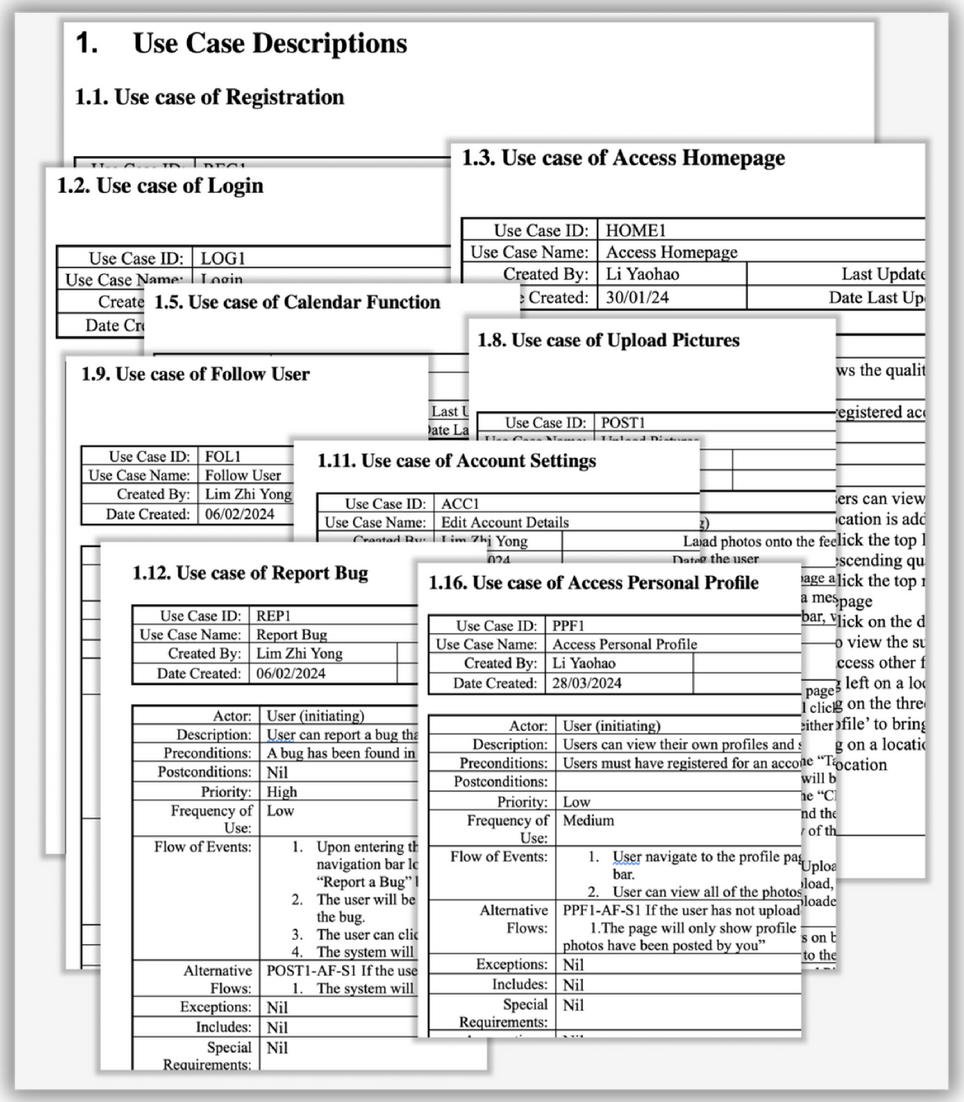
Traceability

Design Traceability

Following UI mockups during coding helps with design traceability by ensuring that the implemented code accurately reflects the intended user interface and design decisions.

Also facilitates alignment between the design and implementation stages of software development.

Skyscape



1. Use Case Descriptions

1.1. Use case of Registration

1.2. Use case of Login

Use Case ID:	LOG1
Use Case Name:	Login

1.3. Use case of Access Homepage

Use Case ID:	HOME1
Use Case Name:	Access Homepage
Created By:	Li Yaohao
Created:	30/01/24

1.5. Use case of Calendar Function

1.8. Use case of Upload Pictures

Use Case ID:	POST1

1.9. Use case of Follow User

Use Case ID:	FOL1
Use Case Name:	Follow User
Created By:	Lim Zhi Yong
Date Created:	06/02/2024

1.11. Use case of Account Settings

Use Case ID:	ACC1
Use Case Name:	Edit Account Details

1.12. Use case of Report Bug

Use Case ID:	REP1
Use Case Name:	Report Bug
Created By:	Lim Zhi Yong
Date Created:	06/02/2024

Actor:	User (initiating)
Description:	User can report a bug tha
Preconditions:	A bug has been found in
Postconditions:	Nil
Priority:	High
Frequency of Use:	Low
Flow of Events:	1. Upon entering th navigation bar lc "Report a Bug" 2. The user will be the bug. 3. The user can clic 4. The system will
Alternative Flows:	POST1-AF-S1 If the use 1. The system will
Exceptions:	Nil
Includes:	Nil
Special Requirements:	Nil

1.16. Use case of Access Personal Profile

Use Case ID:	PPF1
Use Case Name:	Access Personal Profile
Created By:	Li Yaohao
Date Created:	28/03/2024

Actor:	User (initiating)
Description:	Users can view their own profiles and
Preconditions:	Users must have registered for an acco
Postconditions:	
Priority:	Low
Frequency of Use:	Medium
Flow of Events:	1. User navigate to the profile pa bar. 2. User can view all of the photos
Alternative Flows:	PPF1-AF-S1 If the user has not upload 1.The page will only show profile photos have been posted by you"
Exceptions:	Nil
Includes:	Nil
Special Requirements:	Nil

Non-Functional Requirements:

Usability	- The user must be able to cl (English, Chinese, Malay, T - The user interface must be to users from different level - The system should be able mode to accommodate use
Reliability	- After a system reboot, the f within 2 minutes.

Functional requirements

1. The system must have a register fu
   - 1.1. The registration form m information.
     - 1.1.1. The form must consi
     - 1.1.2. The form must consi
     - 1.1.3. The form must consi
     - 1.1.4. The form must consi
   - 1.2. The system must verify
   - 1.3. The system must verify
     - 1.3.1. The username must
     - 1.3.2. The email must not
     - 1.3.3. The password must and have at least 8 c

Traceability

Requirements Traceability

Following use case descriptions and functional and non-functional requirements during coding helps with requirement traceability by ensuring that the implemented code directly addresses and fulfills the specified functionalities.

This establishes a clear link between the software's capabilities and its intended purpose.

5. DEMONSTRATION

NEVER MISS THE GOLDEN HOUR AGAIN.