

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

SC2006 – Software Engineering

Lab 1: Requirements Elicitation

Lab Group	SCMB
Team	SPLIIT
Members	Goh Jun Keat
	Koh Ze Kai Leo
	Madhumita Thiruppathi
	Siah Yee Long
	Teo Liang Wei, Ryan
	Zhang Yichi

Table of Contents

Problem Statement.....	4
Functional Requirements	4
Landing page.....	4
User registration and authentication.....	4
Trip selection	5
Logging expenses	6
Expenses dashboard.....	6
Transaction logs	7
Trip information.....	7
Edit profile	7
Non-functional Requirements.....	9
Usability	9
Reliability	9
Performance	9
Supportability.....	9
Data dictionary.....	10
Initial Use Case Model	13
Use Case diagram.....	13
Use Case descriptions	14
UI Mockups.....	39
Landing Page.....	39
RegisterPage	40
LoginPage.....	41
UserStartPage	42
CreateNewTripPage	43
SuccessfullyCreatedTripPage	44
LogTransactionsPage	45
Dashboard Page.....	46
SeeTransactionsPage.....	47
AltineraryPage.....	48
TripInformationPage	49
EditProfilePage	50

Problem Statement

When travelling in groups, it is common to split expenses among one another, especially for large expenses like flight tickets and accommodation. At the end of the trip, the group will settle and pay what they owe to one another. However, it can get confusing when the number of transactions is large, or when the tracking is not done accurately.

To solve this, we have SPLIIT – an expense tracker for groups that simplifies tracking expenses and settling debts.

SPLIIT can be used by groups who are travelling together or if they just want a simplified expense tracking app that simplifies debt settlement.

Functional Requirements

Landing page

1. The SPLIIT landing page shall detail information about the system where users can get an impression of the system and understand how the system can benefit them
 - 1.1. Information about the system shall include:
 - 1.1.1. Why users should choose to use SPLIIT as their preferred group expense tracker
 - 1.1.2. Who can use SPLIIT
 - 1.1.3. What functionality SPLIIT provides including product snapshots
 - 1.1.4. How the debt settlement function works in SPLIIT

User registration and authentication

2. The SPLIIT landing page shall allow users to register themselves with the system
 - 2.1. During registration, new users will have to enter their:
 - 2.1.1. Email: must be a valid email
 - 2.1.2. Password: must contain at least 8 characters with 1 special character
 - 2.1.3. Username: must be unique within the scope of SPLIIT's **users database**
 - 2.1.4. Display name: need not be unique
 - 2.1.5. Favourite colour: must be within the selected options only
 - 2.2. Users can also register using Google's OAuth but they will still have to pick a username, display name, and favourite colour thereafter
 - 2.3. The registration form shall ensure data validity by checking for the abovementioned requirements
 - 2.3.1. If the requirements are not met, users will be notified to change the field(s) before submitting the registration form
 - 2.3.2. If the requirements are not met, the registration form submission will be unsuccessful

- 2.4. Upon successful registration of a user, the system will encrypt the user's password and store the registered user's information in a secure **users database**
3. The SPLIIT landing page shall allow users to log in to the system
 - 3.1. A user will be prompted to enter their:
 - 3.1.1. Username or email
 - 3.1.2. Password
 - 3.2. Upon submitting the login form, the system will check for the user's corresponding username/email with password
 - 3.2.1. If the login credentials do not match the records in the **users database**, the login request shall be denied
 - 3.2.2. If the login credentials correctly match a record in the users database, the login request shall be accepted, and the system will set the user as the current session's user. The user shall then be redirected to the start page
 - 3.3. Users shall also be able to select a "Forgot password" option, where they will be asked to reset their password upon an email verification of identity

Trip selection

4. Once a user is logged in, they shall be able to select a trip that they have previously created or create a new trip
 - 4.1. When the user clicks on a previously-created trip, the system will set the trip as the current session's trip. They will then be redirected to the expense logging page
5. When a new trip is created, the user shall be prompted to fill in details about the trip such as:
 - 5.1. Trip name: in a text field
 - 5.2. Trip description: in a text field
 - 5.3. Trip image (optional): via file upload
 - 5.4. Foreign currency to track: in a dropdown box
 - 5.5. Local currency to track: in a dropdown box
 - 5.6. I have a Trip ID: text field
 - 5.7. Cities/States visiting: multi-input text field (optional)
 - 5.8. Budget per person: in a numerical field (optional)
 - 5.9. From and To date: via a date selector (optional)
6. Upon creation of a new trip, SPLIIT shall save the new trip's data into SPLIIT's **trips database**, and update the user's trips in the **users database**
 - 6.1. A trip ID will be generated and appended to the new trip's record in the **trips database**
 - 6.2. The username of the trip's creator will also be appended to the new trip's record in the **trips database**
 - 6.3. The corresponding trip ID will be appended to the corresponding user's trips attribute in the **users database**

7. Upon successful creation, SPLIIT will display that the trip creation is successful, and display information for users to proceed forth:
 - 7.1. Trip ID: users should be able to share their trip ID with their group to get them to join the same trip within SPLIIT
 - 7.2. If a user has a trip ID shared with them by the organiser of the trip, they shall be able to key that in in the trip creation form
 - 7.2.1. The user shall then be prompted if they want to confirm joining the trip
 - If the user accepts the join request, then the trip's users in the **trips database** shall be updated accordingly
 - If the user accepts the join request, then the user's trips in the **users database** shall be updated accordingly
- 7.3. A "start logging expenses" button to begin using SPLIIT's main functionality at the expense logging page

Logging expenses

8. Once a logged in user has selected a trip, they shall be navigated to the main page of SPLIIT, where they are prompted to log their expenses
 - 8.1. Expense logging shall be presented as a form, where users can key in, for each expense, the:
 - 8.1.1. Price
 - 8.1.2. Currency of amount paid (either local or foreign)
 - 8.1.3. Recipients
 - 8.1.4. Category
 - 8.1.5. Description
 - 8.1.6. Payer
 - 8.2. The form's fields shall all be validated before submission
 - 8.3. After the form's submission, the transaction data shall be saved in SPLIIT's **transactions database**
 - 8.3.1. Upon submission, the following fields shall also be appended to each transaction record:
 - Timestamp
 - Trip ID
 - Current exchange rate from the trip's foreign to local currency
 - Geographical location information

Expenses dashboard

9. The expenses dashboard page in SPLIIT shall provide insights to the trip's participants' expenses that have been logged. It shall show:
 - 9.1. Every participant within the trip
 - 9.2. A pie chart for each participant with breakdown of their expenses by category
 - 9.3. Each participant's total spending
 - 9.4. A breakdown of how much each participant owes other participants

10. The breakdown of each participant's debt to one another shall be computed by SPLIIT's **debt settlement algorithm**
 - 10.1. The algorithm shall be able to reduce and simplify debts among all participants, and output a **reduced debt matrix** and a **simplified debt matrix** based on the transactions made within the trip
 - 10.1.1. The reduced debt matrix will be displayed as requirement 9.4.
 - 10.1.2. The simplified debt matrix will be displayed if the user selects the "Simplify debts" button in the dashboard's page

Transaction logs

11. SPLIIT shall allow users to view the trip's transaction logs
 - 11.1. The user shall be able to click into each transaction to see its details
 - 11.2. The user shall be able to delete a transaction
 - 11.3. The expenses shall be displayed in the original currency logged by default
 - 11.4. The transactions shall be exportable into a CSV format if the user chooses to extract the transactions data

Trip information

12. SPLIIT shall allow users to view the current trip's information such as:
 - 12.1. The information as created during the trip creation stage
 - 12.2. The current users who have joined the current trip
13. SPLIIT shall also allow users to:
 - 13.1. Edit the Cities/States visiting of the trip
 - 13.2. Edit the start and end date of the trip
 - 13.3. Edit the budget of the trip
 - 13.4. Delete the trip
 - 13.4.1. Users will be prompted to key in the trip's name in order to fully confirm that they want to delete the entire trip
 - 13.4.2. Upon successful deletion:
 - The participants' trips in the **users database** will be adjusted to reflect accordingly
 - The corresponding trip record will be deleted in the **trips database**
 - The corresponding transactions containing the trip's ID will be deleted in the **transactions database**

Edit profile

14. SPLIIT shall allow users to edit their profile to:
 - 14.1. Change their display name
 - 14.2. Change their favourite colour
 - 14.3. Delete a trip

14.3.1. Users shall be prompted which trip they would like to delete, and be redirected to the same functionality as requirement 13.4

Non-functional Requirements

Usability

1. The SPLIIT website should have a simple and intuitive user interface
 - 1.1. Users should be able to comfortably find the information they need within 2 minutes of entering the site
2. The SPLIIT website should be mobile responsive
 - 2.1. Users should be able to see 100% of the page content with at most 20% of redundant information (like navigation tabs) hidden for a more pleasant user experience
 - 2.2. The website should remain usable regardless if it is viewed on a PC, laptop, mobile tablet or mobile phone
3. All error messages should be clear and provide actionable feedback.
4. Form validation should be immediate and clear.

Reliability

1. Upon page reload, every page of the SPLIIT website should be viewable within 5s on a Chrome browser using a standard 2.4G Wi-Fi connection on a mid-tier laptop or an equivalent.
2. The system must ensure data integrity, preventing data corruption or loss
3. Can handle simultaneous user access without crashes or inconsistencies
4. Login and authentication must be robust and ensure no unauthorized access

Performance

1. Currency conversion should update at least weekly.
2. The SPLIIT website should process expense submissions within 5 seconds.

Supportability

1. System should support automated updates without disrupting active users
2. The system must be deployable on any cloud infrastructure with minimal configuration changes, ensuring portability and scalability.
3. System logs should be centralised and searchable.
4. System architecture should follow microservices principles for easy maintenance.

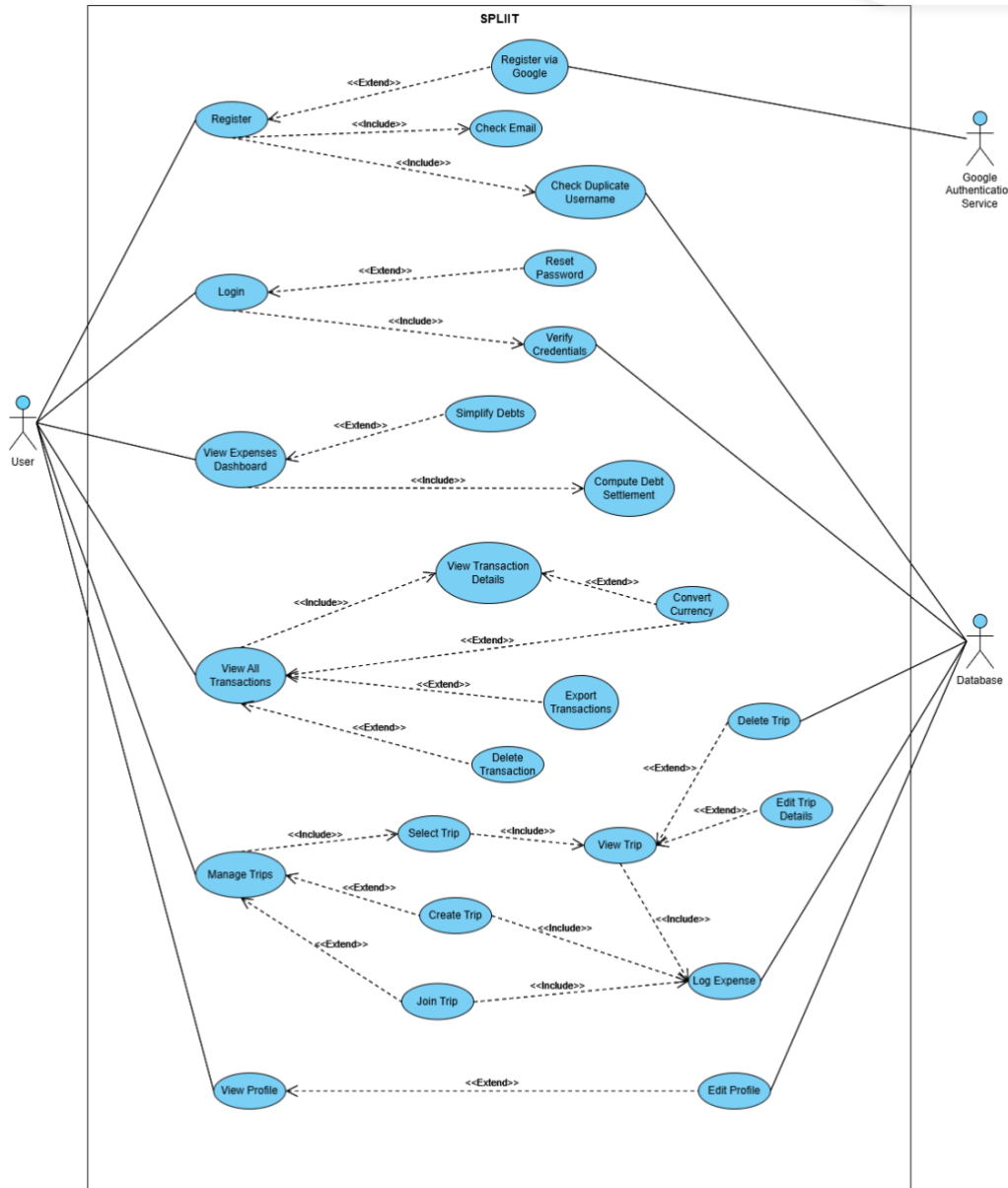
Data dictionary

Term	Definition
SPLIIT	The name of the web application. Can refer to the name of the system itself too
Trip	A session where the group of users log expenses in. The expenses logged will be in the context of the trip. Can be named for identification purposes like "Our Disney Cruise Trip" or "Japan 2025"
Trip ID	A unique ID generated for each trip to identify the reference to a trip
Expense	<p>A record of expenditure. It should contain values for:</p> <ul style="list-style-type: none">- Recipients- Amount- Currency- Description- Payer <p>Automatically-appended values for:</p> <ul style="list-style-type: none">- Timestamp- Trip ID- Current exchange rate from the trip's foreign to local currency- Geographical location information <p>Will also be logged in each expense transaction</p>
Currency	The currency that was transacted in the expense. Either the foreign or local currency. This field is non-nullable
Recipients	One or many users who are the beneficiaries or recipients of an expense logged. The amount paid will be split equally among all recipients. E.g. there are 4 recipients for a \$12 expense. Each recipient will be deemed to have spent \$3 on that expense. This field is non-nullable
Payer	A single user who paid for the expense (whether it be on behalf of others or not). This field is non-nullable
Description	Text describing the expense for the user's reference. Not important in the functionality of SPLIIT
Debt	The amount that a user owes another, based on the home currency. E.g. if user A owes user B MYR3.30, then user A is in debt to user B by ~SGD1
Debt Matrix	An n by n matrix, where n is the number of users in the trip. The matrix represents the debt between each user. E.g. if $\text{debt_matrix}[x][y] = 4.13$, it means user x owes user y SGD4.13

Reduced Debt Matrix	<p>Reducing transactions will reduce unnecessary back-and-forth transactions.</p> <p>E.g. if A owes B \$10, but B owes A \$5, then A effectively only owes B \$5. The Reduced Debt Matrix reduces these 2 transactions down to 1.</p>
Simplified Debt Matrix	<p>The Simplified Debt Matrix minimises the number of transactions among users in a trip to settle debts.</p> <p>E.g. if A owes B \$10, but B owes C \$10, but C owes A \$10, then instead of making 3 transactions among them, simplifying the transactions will ensure no transactions are made in this case.</p> <p>At the end of simplifying debts, those who are owed will receive what they are expected to get in return, nothing more, nothing less.</p>
Foreign currency	The selected currency foreign to the user in the destination currency. Selected when creating a trip
Local currency	The selected currency familiar to the user in their home currency. Selected when creating a trip
Login credentials	Username or email, with corresponding password
Participant	A user who is in a particular trip
Owner	The creator of a trip
Users database	<p>The database (or table) that stores information about users. It should contain information such as:</p> <ul style="list-style-type: none"> - Username (unique) - Display name - Favourite colour - Trips
Trips database	<p>The database (or table) that stores information about trips. It should contain information such as:</p> <ul style="list-style-type: none"> - Trip ID (unique) - Trip name - Trip description - Trip image (optional) - Foreign currency - Local currency - Cities / states (optional) - Budget per person (optional) - From and to date (optional) - Participants
Transactions database	The database (or table) that stores information about the expenses.

Initial Use Case Model

Use Case diagram



Use Case descriptions

1.1 Register

Use Case ID:	UC-01-1		
Use Case Name:	Register		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to register an account on SPLIIT
Preconditions:	1. User has a valid and unique email address
Postconditions:	1. New user account is created and stored in database. 2. A verification email is sent to provided email address.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	1. The user navigates to the registration page. 2. The user enters their email, password, username, display name and favorite color. 3. The system checks email uniqueness using the use case CheckEmail. 4. The system checks username uniqueness using the use case CheckDuplicateUsername. 5. The user reviews and accepts the terms, conditions, privacy concerns and disclaimers. 6. The user clicks the "Create Account" button. 7. The system creates an account for the user and stores their data in a secure database.
Alternative Flows:	AF-S3: If the email provided is already in use 1. The system prompts user to provide another email. 2. The system returns to step 2. AF-S4: If the username provided is already in use 1. The system prompts user to enter another username. 2. The system returns to step 2.
Exceptions:	User is notified if verification email is unable to be sent.
Includes:	1. CheckEmail 2. CheckDuplicateUsername
Special Requirements:	1. Must adhere to terms and conditions. 2. All data must be encrypted.
Assumptions:	The user has entered accurate and complete information.
Notes and Issues:	None

1.2 Register via Google

Use Case ID:	UC-01-2		
Use Case Name:	RegisterViaGoogle		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User (Initiating Actor), Google Authentication Service
Description:	Allows users to register on SPLIIT using their Google account.
Preconditions:	1. User has an active Google account.
Postconditions:	1. New user account is created with data obtained via Google and stored in database. 2. A verification email is sent to provided email address.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects the option to register via Google OAuth. 2. The system redirects the user to the Google OAuth authentication page. 3. The user authenticates using their Google account. 4. If authentication is successful, Google returns basic user information to the system. 5. The user enters their username, display name and favourite colour. 6. The user reviews and accepts the terms, conditions, privacy concerns and disclaimers. 7. The user clicks the "Create Account" button. 8. The system creates an account for the user and stores their data in a secure database.
Alternative Flows:	AF-S4: If Google OAuth authentication fails <ol style="list-style-type: none"> 1. The system displays "Authentication failed. Please try again." 2. The system returns to step 1.
Exceptions:	The registration process is aborted if user cancels the Google Oath process.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. The integration must comply with Google OAuth security standards. 2. Must adhere to terms and conditions. 3. All data must be encrypted.
Assumptions:	Google OAuth service is available and functioning correctly.
Notes and Issues:	None

1.3 Check Email

Use Case ID:	UC-01-3		
Use Case Name:	CheckEmail		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	System
Description:	Check the uniqueness of the email provided.
Preconditions:	None
Postconditions:	1. The uniqueness of the email provided is determined.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user enters an email address. 2. The system checks the email provided against the database. 3. If there is no matching email in the database, the email provided is deemed to be unique.
Alternative Flows:	AF-S3: If there is a match in the database <ol style="list-style-type: none"> 1. The system displays "Email already in use!" 2. The system returns to step 1.
Exceptions:	None
Includes:	None
Special Requirements:	1. All data must be encrypted.
Assumptions:	The user has entered a valid email address.
Notes and Issues:	None

1.4 Check Duplicate Username

Use Case ID:	UC-01-4		
Use Case Name:	CheckDuplicateUsername		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	11/2/2025	Date Last Updated:	11/2/2025

Actor:	System, Database
Description:	Check the uniqueness of the username provided.
Preconditions:	None
Postconditions:	1. The uniqueness of the username provided is determined.
Priority:	High
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user enters a username. 2. The system checks the username provided against the database. 3. If there is no matching username in the database, the username provided is deemed to be unique.
Alternative Flows:	AF-S3: If there is a match in the database <ol style="list-style-type: none"> 1. The system displays "Username already in use!" 2. The system returns to step 1.
Exceptions:	None
Includes:	None
Special Requirements:	All data must be encrypted.
Assumptions:	None
Notes and Issues:	None

2.1 Login

Use Case ID:	UC-02-1
--------------	---------

Use Case Name:	Login		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allow users to log in to their SPLIIT account.
Preconditions:	1. The user must have a registered account
Postconditions:	1. The user is navigated to their dashboard on the home page.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to the login page. 2. The user enters their email or username along with their password. 3. The user clicks the "Login" button. 4. The system uses the use case VerifyCredentials to check the credentials provided against the database. 5. If the credentials are verified, the user is redirected to their dashboard on the home page.
Alternative Flows:	AF-S5: If credential validation fails <ol style="list-style-type: none"> 1. The system prompts user to re-enter their email or username and password. 2. The system returns to step 1.
Exceptions:	If any information is missing, an error message is displayed.
Includes:	1. VerifyCredentials
Special Requirements:	The password entered must be encrypted and secured.
Assumptions:	User has an existing and active SPLIIT account.
Notes and Issues:	Consider logging all login attempts for security purposes.

2.2 Verify Credentials

Use Case ID:	UC-02-2
--------------	---------

Use Case Name:	VerifyCredentials		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	System, Database
Description:	Verifies whether the email or username and password provided match the database
Preconditions:	None
Postconditions:	1. The system successfully determines whether the email or username and password provided match the database.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user inputs their email or username along with their password. 2. The system checks whether the password provided matches the password stored for the email or username provided. 3. If the email-password or username-password pair match, the inputs are successfully validated.
Alternative Flows:	AF-S3: If the email-password pair or username-password pair do not match <ol style="list-style-type: none"> 1. The system displays "Incorrect email/username/password!" 2. The system returns to step 1.
Exceptions:	None
Includes:	None
Special Requirements:	1. All data, especially the password must be encrypted.
Assumptions:	The user has entered an email address or username linked to an active SPLIIT account.
Notes and Issues:	None

2.3 Reset Password

Use Case ID:	UC-02-3
--------------	---------

Use Case Name:	ResetPassword		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to reset their password if they forgot it.
Preconditions:	1. The user must have an active SPLIIT account.
Postconditions:	1. The user password is updated, encrypted and stored in database
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to login page. 2. The user clicks the “Forgot Password” button. 3. The system prompts the user to verify their identity via email. 4. A verification email is sent to the provided email address. 5. Upon successful email verification, the system prompts the user to enter a new password. 6. The system updates user password by storing the new password in database.
Alternative Flows:	<p>AF-S4: If verification email is not received by user</p> <ol style="list-style-type: none"> 1. The user can request the system to resend the email after 1 minute. <p>AF-S5: If email verification fails</p> <ol style="list-style-type: none"> 1. The system aborts the reset password process and returns to step 1.
Exceptions:	User is notified if verification email is unable to be sent.
Includes:	None
Special Requirements:	The new password entered must be encrypted and secured.
Assumptions:	The user has entered a valid and registered email address for verification.
Notes and Issues:	None

3.1 View Expenses Dashboard

Use Case ID:	UC-03-1
--------------	---------

Use Case Name:	ViewExpensesDashboard		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Displays insights into the expenses of every user for a trip
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The user has selected an existing trip.
Postconditions:	<ol style="list-style-type: none"> 1. Every participant of the selected trip is displayed. 2. Displays a pie chart for each participant showing a breakdown of their expenses by category. 3. Each participant's total spending is displayed. 4. A breakdown of how much each participant owes other participants is displayed.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to the dashboard. 2. The user selects an existing trip. 3. The system retrieves and displays all expenses logged. 4. The system automatically uses the use case ComputeDebtSettlement and produces an inter-participant debt matrix. 5. The computed debt matrix is displayed.
Alternative Flows:	<p>AF-S2: If user does not select an existing trip</p> <ol style="list-style-type: none"> 1. The system displays "Trip not found!" 2. The system returns to step 1. <p>AF-S3: If the trip selected has no expenses logged</p> <ol style="list-style-type: none"> 1. The system displays "No transactions found."
Exceptions:	If system encounters error while retrieving data, user is notified and dashboard does not load expense details.
Includes:	<ol style="list-style-type: none"> 1. ComputeDebtSettlement
Special Requirements:	<ol style="list-style-type: none"> 1. Dashboard must load quickly and be accessible on multiple devices. 2. All data and information must be displayed clearly. 3. Explanations provided must be clear and concise.
Assumptions:	The data used for analysis is reliable and up to date.
Notes and Issues:	None

3.2 Compute Debt Settlement

Use Case ID:	UC-03-2
--------------	---------

Use Case Name:	ComputeDebtSettlement		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	System
Description:	Describes how the system calculates debt settlement among trip participants using the debt settlement algorithm
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The user has selected an existing trip with logged expenses
Postconditions:	<ol style="list-style-type: none"> 1. A debt settlement matrix is successfully computed and displayed on the user's dashboard
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The system retrieves all logged expenses for the selected trip from database. 2. The system analyses the data and summarises each participant's expenses. 3. The system calculates each participant's net balance by comparing the total amount paid against what they owe. 4. The debt settlement algorithm processes the data and produces an inter-participant debt matrix. 5. The computed matrix is passed to dashboard for display.
Alternative Flows:	None
Exceptions:	If output debt matrix is unable to be produced or displayed, user is notified and an error message is displayed.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. The computation must complete within a short time. 2. The algorithm must accurately calculate results.
Assumptions:	All expense transactions have been accurately recorded and stored in system database.
Notes and Issues:	Can cache the computed results for better performance on next retrieval.

3.3 Simplify Debts

Use Case ID:	UC-03-3
--------------	---------

Use Case Name:	SimplifyDebts		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to remove key skills from their profile
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The system must have successfully computed an inter-participant debt matrix.
Postconditions:	<ol style="list-style-type: none"> 1. A simplified debt matrix is successfully computed and displayed on the user's dashboard
Priority:	Medium
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user clicks the "Simplify Debts" button. 2. The system uses the debt settlement algorithm to produce a simplified debt matrix. 3. The simplified debt matrix is passed to dashboard for display.
Alternative Flows:	None
Exceptions:	If simplified debt matrix is unable to be produced, user is notified and the original inter-participant matrix is displayed.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. The computation must complete within a short time. 2. The algorithm must accurately calculate results.
Assumptions:	All expense transactions have been accurately recorded and stored in system database.
Notes and Issues:	None

4.1 View All Transactions

Use Case ID:	UC-04-1
--------------	---------

Use Case Name:	ViewAllTransactions		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to view all the transaction logs of a trip.
Preconditions:	1. The user must be logged in.
Postconditions:	1. All transaction logs of the trip are displayed.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects an existing trip. 2. The system retrieves all transaction logs of the selected trip from database. 3. The transaction logs are displayed to the user.
Alternative Flows:	AF-S2: If the selected trip has no recorded transaction logs. <ol style="list-style-type: none"> 1. The system displays “No transactions recorded.”
Exceptions:	If system encounters error while retrieving data, user is notified and transaction logs are not displayed.
Includes:	1. ViewTransactionDetails
Special Requirements:	<ol style="list-style-type: none"> 1. Transaction logs must load quickly and be accessible on multiple devices. 2. All information must be displayed clearly in a list.
Assumptions:	All expense transactions have been accurately recorded and stored in system database.
Notes and Issues:	None

4.2 View Transaction Details

Use Case ID:	UC-04-2
--------------	---------

Use Case Name:	ViewTransactionDetails		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows user to view details of a selected transaction log.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. All transaction logs must be successfully displayed.
Postconditions:	<ol style="list-style-type: none"> 1. The details of the specified transaction are displayed.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects a transaction log from the list of all transaction logs. 2. The system retrieves all details of the selected transaction. 3. The details are displayed to the user.
Alternative Flows:	None
Exceptions:	If system encounters error while retrieving data, user is notified and transaction details are not displayed.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. Transaction details must load quickly and be accessible on multiple devices. 2. All information must be displayed clearly.
Assumptions:	The details of the transaction are complete and accurate.
Notes and Issues:	None

4.3 Convert Currency

Use Case ID:	UC-04-3
--------------	---------

Use Case Name:	ConvertCurrency		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User, System
Description:	Converts transaction amount(s) to local currency.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. All transaction logs must be successfully displayed.
Postconditions:	<ol style="list-style-type: none"> 1. The selected transaction log amount(s) are converted to the user's local currency.
Priority:	Medium
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects an existing trip. 2. The user clicks the "Convert Currency" button. 3. The system uses the most recent exchange rate to convert all the transaction amounts to the user's local currency. 4. The converted amounts are displayed, replacing the original amount shown.
Alternative Flows:	<p>AF-S1: If user selects a specific transaction</p> <ol style="list-style-type: none"> 1. The user clicks the "Convert Currency" button. 2. The system uses the most recent exchange rate to convert the transaction amount to the user's local currency. 3. The converted amount is displayed, replacing the original amount shown.
Exceptions:	If the conversion fails, the user is notified and amount is not converted.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. Converted transaction amounts must load quickly. 2. All information must be displayed clearly.
Assumptions:	<ol style="list-style-type: none"> 1. The conversion rates used are the most recent rates. 2. There are no errors in the conversions.
Notes and Issues:	None

4.4 Export Transactions

Use Case ID:	UC-04-4
--------------	---------

Use Case Name:	ExportTransactions		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to export all transactions as CSV file.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. All transaction logs must be successfully displayed.
Postconditions:	<ol style="list-style-type: none"> 1. All transaction logs are exported as CSV file.
Priority:	Low
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects the "Export to CSV" button. 2. The system converts the transaction logs into a CSV file. 3. The user can choose to download the CSV file.
Alternative Flows:	None
Exceptions:	The user is notified if CSV file is unable to be generated.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. The CSV file must be generated within a short time.
Assumptions:	All transactions to be exported have been accurately recorded and stored in system database beforehand.
Notes and Issues:	None

4.5 Delete Transaction

Use Case ID:	UC-04-5
--------------	---------

Use Case Name:	DeleteTransaction		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows user to delete a transaction.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The transaction selected to be deleted must exist.
Postconditions:	<ol style="list-style-type: none"> 1. The selected transaction is deleted and database is updated.
Priority:	Low
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects a transaction. 2. The user clicks the "Delete transaction" button. 3. The system prompts the user to confirm their actions. 4. Upon confirmation, the system deletes the transaction and updates database.
Alternative Flows:	AF-S4: If user cancels their actions <ol style="list-style-type: none"> 1. The delete process is aborted.
Exceptions:	If transaction is unable to be deleted, the user is notified and transaction is not deleted.
Includes:	None
Special Requirements:	The database must be updated securely.
Assumptions:	The user is aware of the implications of their actions.
Notes and Issues:	Deleted transactions cannot be restored.

5.1 Manage Trips

Use Case ID:	UC-05-1
--------------	---------

Use Case Name:	ManageTrips		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to manage all their trips.
Preconditions:	1. The user must be logged in.
Postconditions:	1. All the user's existing trips are displayed.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to trips page. 2. The system retrieves all the user's existing trips from database. 3. All the trips are displayed on the trips page. 4. If the user selects "Create Trip", the system uses the use case CreateTrip to create a new trip. 5. If the user selects "Join Trip", the system uses the use case JoinTrip to allow user to join an existing trip.
Alternative Flows:	AF-S3: User has no existing trips in database 1. The system displays "No Trips Found."
Exceptions:	If system encounters error while retrieving data, user is notified and trips are not displayed.
Includes:	1. SelectTrip
Special Requirements:	<ol style="list-style-type: none"> 1. All trips must load quickly and be accessible on multiple devices. 2. All information must be displayed clearly.
Assumptions:	All trips have been accurately recorded and stored in system database.
Notes and Issues:	None

5.2 Select Trip

Use Case ID:	UC-05-2
--------------	---------

Use Case Name:	SelectTrip		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows user to select an existing trip.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. All the user's existing trips are displayed.
Postconditions:	None
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects a trip from list of existing trips. 2. The system uses the use case "ViewTrip" to display all the selected trip's details.
Alternative Flows:	None
Exceptions:	If trip is unable to be selected, the user can choose to reload the page.
Includes:	<ol style="list-style-type: none"> 1. ViewTrip
Special Requirements:	None
Assumptions:	The selected trip is the user's existing trip recorded in the database.
Notes and Issues:	User only can select trips in which they are a participant.

6.3 View Trip

Use Case ID:	UC-06-3
--------------	---------

Use Case Name:	ViewTrip		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	System
Description:	Allows user to view all details of a selected trip.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The user has selected an existing trip.
Postconditions:	<ol style="list-style-type: none"> 1. All the details related to the selected trip are displayed.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The system retrieves all data related to the selected trip from database. 2. The data is displayed to the user.
Alternative Flows:	None
Exceptions:	If system encounters error while retrieving data, user is notified and trip is not selected.
Includes:	<ol style="list-style-type: none"> 1. LogExpense
Special Requirements:	<ol style="list-style-type: none"> 1. All data from the selected trip must load quickly and be accessible on multiple devices. 2. All information must be displayed clearly.
Assumptions:	All trip details have been accurately recorded and stored in system database beforehand.
Notes and Issues:	None

6.4 Delete Trip

Use Case ID:	UC-06-4
--------------	---------

Use Case Name:	DeleteTrip		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User, Database
Description:	Allows user to delete a trip
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The trip selected to be deleted must exist.
Postconditions:	<ol style="list-style-type: none"> 1. The selected trip is deleted and database is updated.
Priority:	Low
Frequency of Use:	Low
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects a trip 2. The user clicks the "Delete trip" button. 3. The system prompts the user to confirm their actions. 4. Upon confirmation, the system deletes the trip and updates database.
Alternative Flows:	AF-S4: If user cancels their actions <ol style="list-style-type: none"> 1. The delete process is aborted.
Exceptions:	If trip is unable to be deleted, the user is notified and trip is not deleted.
Includes:	None
Special Requirements:	The database must be updated securely.
Assumptions:	The user is aware of the implications of their actions.
Notes and Issues:	Deleted trips cannot be restored.

6.5 Edit Trip Details

Use Case ID:	UC-06-5
--------------	---------

Use Case Name:	EditTripDetails		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows user to edit trip details, such as cities/state visited, start date, end date and budget
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The trip selected to be deleted must exist.
Postconditions:	<ol style="list-style-type: none"> 1. The trips details and database are updated.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects a trip. 2. The user clicks the "Edit Trip" button. 3. The user updates the cities/states visited, start date, end date or budget of the trip. 4. The user clicks the "Save" button. 5. The system updates the database and changes are displayed.
Alternative Flows:	AF-S3: The user cancels the edit process <ol style="list-style-type: none"> 1. The edit process is aborted and nothing is changed.
Exceptions:	If there is a database error, an error message is displayed and changes are not saved.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. All updates must be encrypted and saved securely. 2. All changes should be reflected immediately upon saving.
Assumptions:	The user has entered accurate data.
Notes and Issues:	None

6.6 Create Trip

Use Case ID:	UC-06-6
--------------	---------

Use Case Name:	CreateTrip		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows user to create a new trip
Preconditions:	1. The user must be logged in.
Postconditions:	1. A new trip is created with a unique Trip ID
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects the "Create Trip" button. 2. The user enters the trip name, trip description, foreign currency to track and local currency to track. 3. The user can optionally enter a trip image, destination, custom Trip ID, cities/states to be visited, estimated budget per person, start and end dates. 4. The system creates a new trip with the input data and saved it in the database. 5. A unique Trip ID will be generated and appended to the new trip's records in database alongside the user's username.
Alternative Flows:	AF-S2: If there is missing information <ol style="list-style-type: none"> 1. The user is prompted to fill in the missing information. 2. When inputs are complete, the system resumes flow.
Exceptions:	If new trip cannot be created, the user is notified and no trip is created.
Includes:	1. LogExpense
Special Requirements:	<ol style="list-style-type: none"> 1. All data must be encrypted. 2. Only the trip owner can share the unique Trip ID with other users.
Assumptions:	The user has entered accurate and complete information.
Notes and Issues:	None

6.7 Join Trip

Use Case ID:	UC-06-7
--------------	---------

Use Case Name:	JoinTrip		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to join an existing trip using unique Trip ID.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The trip to be joined must exist.
Postconditions:	<ol style="list-style-type: none"> 1. The user successfully joins the trip with the Trip ID.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects the "Join Trip" button. 2. The system prompts the user to enter the unique Trip ID of the trip they want to join. 3. The system checks the database and finds a trip with the matching Trip ID. 4. The system prompts the user to confirm if they want to join the trip. 5. Upon confirmation, the user joins the trip and the database is updated.
Alternative Flows:	<p>AF-S3: Trip not found</p> <ol style="list-style-type: none"> 1. The system displays "Trip not found!" 2. The system returns to step 2. <p>AF-S5: The user cancels the join process.</p> <ol style="list-style-type: none"> 1. The user is not added to the trip. 2. The system returns to trips page.
Exceptions:	If the trip joining process fails, the user is notified and user does not join the trip.
Includes:	<ol style="list-style-type: none"> 1. LogExpense
Special Requirements:	<ol style="list-style-type: none"> 1. Unique Trip ID must be encrypted. 2. Database must be updated securely.
Assumptions:	The user has entered the Trip ID of an existing trip.
Notes and Issues:	None

6.8 Log Expense

Use Case ID:	UC-06-8
--------------	---------

Use Case Name:	LogExpense		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User, Database
Description:	Allows users to log a transaction.
Preconditions:	<ol style="list-style-type: none"> 1. The user must be logged in. 2. The user must be a participant in an existing trip.
Postconditions:	<ol style="list-style-type: none"> 1. The transaction is successfully logged.
Priority:	High
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user selects option to add a transaction. 2. The user enters the amount paid, currency of amount paid, recipients, category of payment, payment description and payer. 3. The user presses the “Log Transaction” button. 4. The system validates the inputs. 5. Upon successful validation, the system saves the transaction data in database. 6. A timestamp, the Trip ID, the current exchange rate and geographical location information are also appended to the saved transaction record.
Alternative Flows:	AF-S5: If there is missing information <ol style="list-style-type: none"> 1. The user is prompted to fill in the missing information. 2. The system returns to step 2.
Exceptions:	If transaction cannot be added, the user is notified and transaction is not added.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. All data must be encrypted and saved securely.
Assumptions:	The user has entered accurate and complete information.
Notes and Issues:	None

7.1 View Profile

Use Case ID:	UC-07-1
--------------	---------

Use Case Name:	ViewProfile		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User
Description:	Allows users to view their profile.
Preconditions:	1. The user must be logged in.
Postconditions:	1. The user's profile is displayed.
Priority:	Medium
Frequency of Use:	High
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to the profile page. 2. The system retrieves the user's profile information from database. 3. The information is displayed on the profile page.
Alternative Flows:	None
Exceptions:	If there is a database error, an error message is displayed and profile is not shown.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. User profile must load quickly and be accessible on multiple devices. 2. All information must be displayed clearly. 3. All data must be encrypted in database.
Assumptions:	The information displayed is accurate and complete
Notes and Issues:	None

7.2 Edit Profile

Use Case ID:	UC-07-2
--------------	---------

Use Case Name:	EditProfile		
Created By:	Goh Jun Keat	Last Updated By:	Goh Jun Keat
Date Created:	10/2/2025	Date Last Updated:	10/2/2025

Actor:	User, Database
Description:	Allows user to edit their profile information
Preconditions:	1. The user must be logged in.
Postconditions:	1. The profile information and database are updated.
Priority:	Medium
Frequency of Use:	Medium
Flow of Events:	<ol style="list-style-type: none"> 1. The user navigates to profile page 2. The user clicks the "Edit Profile" button. 3. The user updates their display name and favourite colour. 4. The user clicks the "Save" button. 5. The system updates the database and changes are displayed.
Alternative Flows:	AF-S3: The user cancels the edit process 2. The edit process is aborted and nothing is changed.
Exceptions:	If there is a database error, an error message is displayed and changes are not saved.
Includes:	None
Special Requirements:	<ol style="list-style-type: none"> 1. All updates must be encrypted and saved securely. 2. All changes should be reflected immediately upon saving.
Assumptions:	The user has entered accurate data.
Notes and Issues:	None

UI Mockups

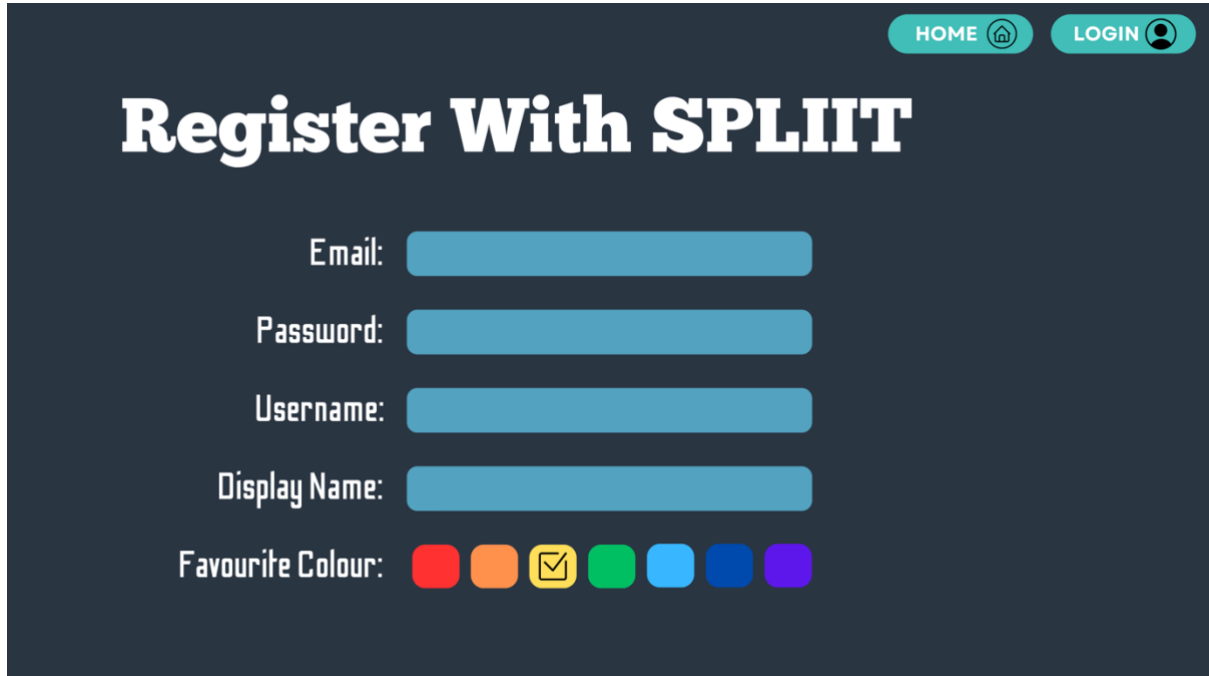
Landing Page

The landing page will display information about the SPLIIT app for users to get a better idea of what SPLIIT is about and how to use it. It should be in a layout of a generic blog page.



RegisterPage

The Register Page will be used for new users to register themselves with SPLIIT. It should be in a layout of a form.

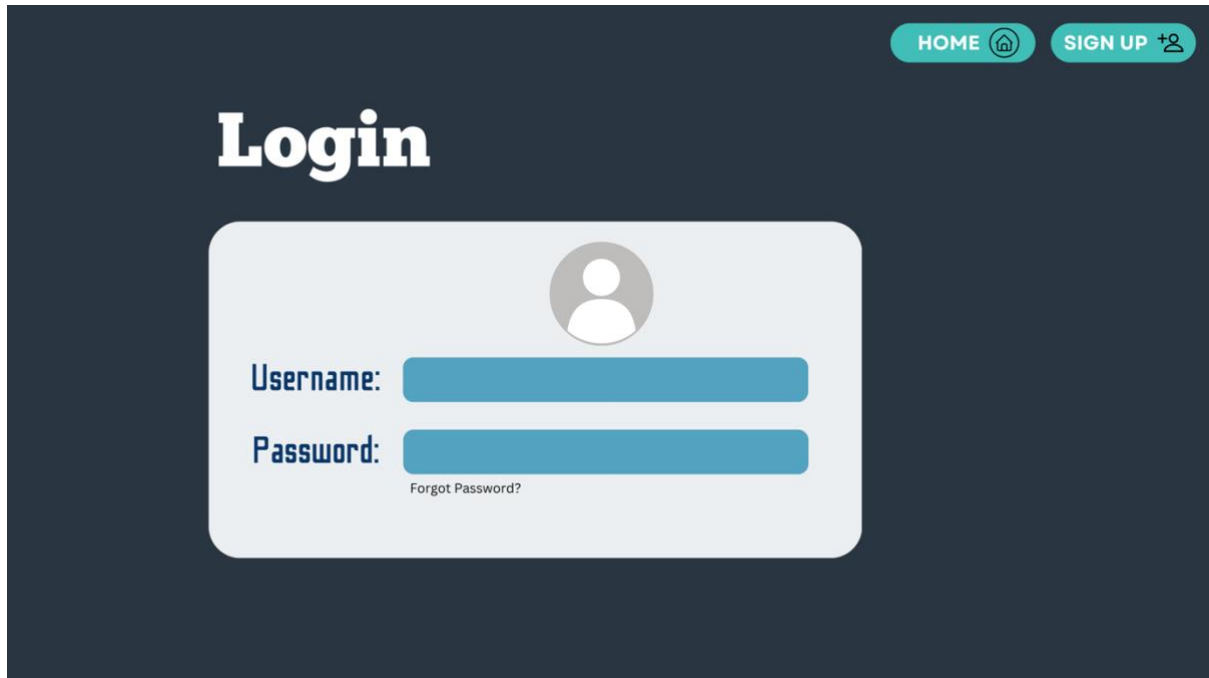


The image shows a registration form titled "Register With SPLIIT" on a dark blue background. At the top right, there are two buttons: "HOME" with a house icon and "LOGIN" with a user icon. The form fields are as follows:


- Email:
- Password:
- Username:
- Display Name:
- Favourite Colour: A row of seven colored squares (red, orange, yellow, green, light blue, dark blue, purple) with a yellow square containing a checkmark icon selected.


LoginPage

The Login Page will be used for existing users to log in to SPLIIT. It should be in a layout of a form.




The image shows a login page with a dark blue background. In the top right corner, there are two teal buttons: "HOME" with a house icon and "SIGN UP" with a plus and person icon. The word "Login" is displayed in large white text. Below it is a light gray rounded rectangle containing a user profile icon, a "Username:" label with a teal input field, a "Password:" label with a teal input field, and a "Forgot Password?" link.

HOME 

SIGN UP 

Login



Username:

Password:

[Forgot Password?](#)

UserStartPage

The UserStartPage will be the first page a user sees after logging in. Users are prompted to activate an existing trip or start a new trip by creating one. Trips are listed in large cards as a “call to action”.



CreateNewTripPage

The CreateNewTripPage will be used to create a new trip based on a user's desired configuration. It should be in a layout of a form.

Create New Trip

I already have a trip ID:

Trip Name:

Trip Currency: USD ▼

Home Currency: SGD ▼

+ CREATE

-Optional-

Cities/States: ▼

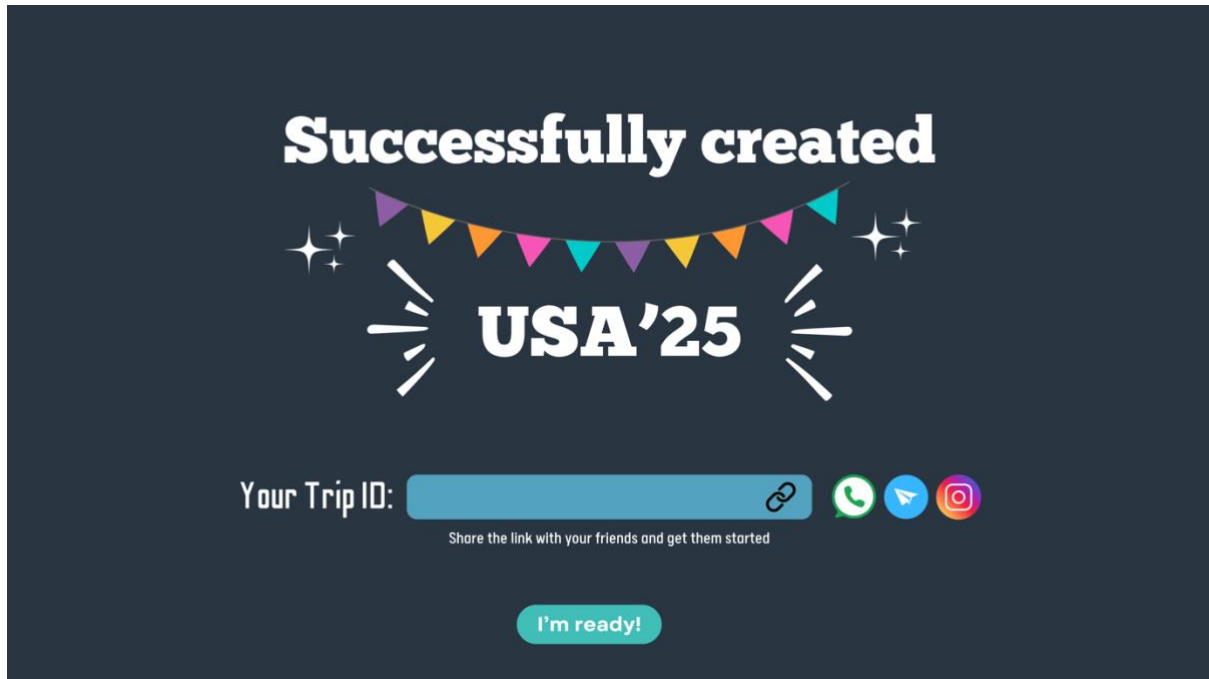
☐ Florida ☐ Ohio

Budget:

From-To:
(dd/mm/yy)

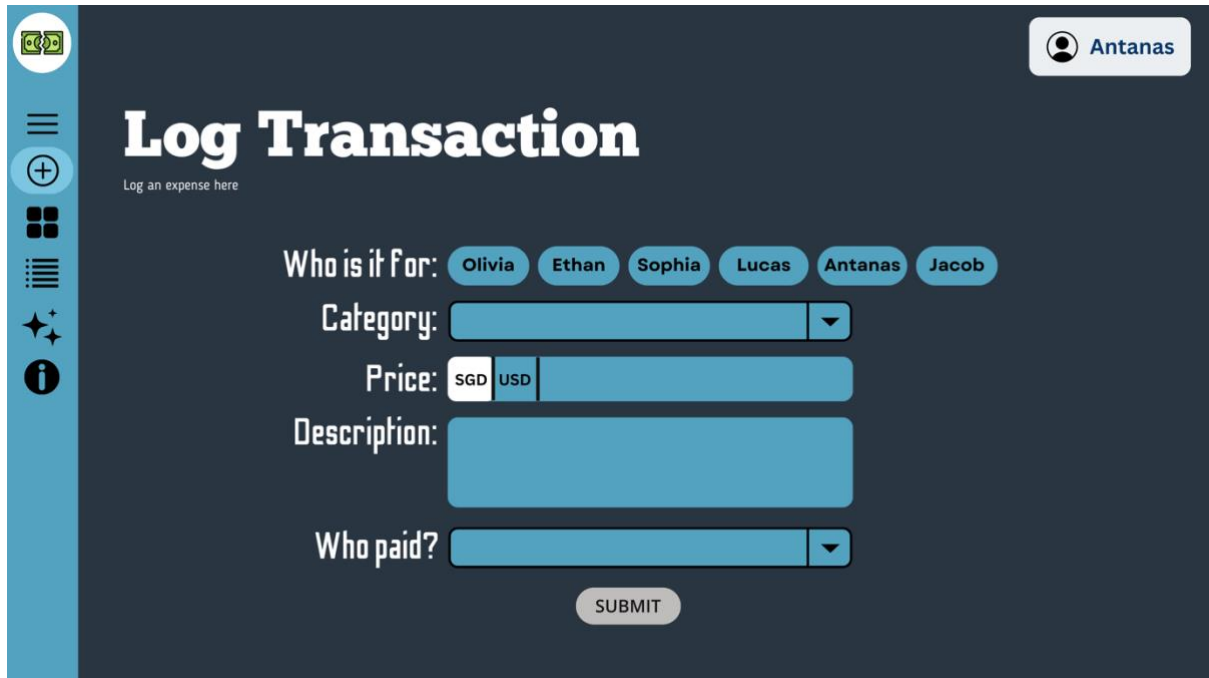
SuccessfullyCreatedTripPage

The SuccessfullyCreatedTripPage is shown upon successful creation of a trip. It will contain interactive elements for users to share the trip ID, either via a link or a copy to clipboard button. It then brings the user to the next step which is to enter the main part of SPLIIT.











LogTransactionsPage

The LogTransactionsPage will be the first page of the main part of SPLIIT. Users are to log every transaction made in this page. It should be in a layout of a form.



The screenshot shows a web application interface for logging transactions. On the left is a dark blue sidebar with a vertical stack of icons: a wallet icon, a hamburger menu, a plus sign, a grid, a list, a star, and an information icon. The main content area has a dark blue header with a user profile icon and the name 'Antanas' in the top right. Below the header, the title 'Log Transaction' is displayed in large white font, with the subtitle 'Log an expense here' in smaller white font. The form consists of several fields: 'Who is it for:' with a horizontal list of name buttons (Olivia, Ethan, Sophia, Lucas, Antanas, Jacob); 'Category:' with a text input and a dropdown arrow; 'Price:' with a currency selector (SGD, USD) and a text input; 'Description:' with a large text area; and 'Who paid?' with a text input and a dropdown arrow. A 'SUBMIT' button is located at the bottom center of the form.



 Antanas

Log Transaction

Log an expense here

Who is it for: Olivia Ethan Sophia Lucas Antanas Jacob

Category: ▼

Price: SGD USD

Description:

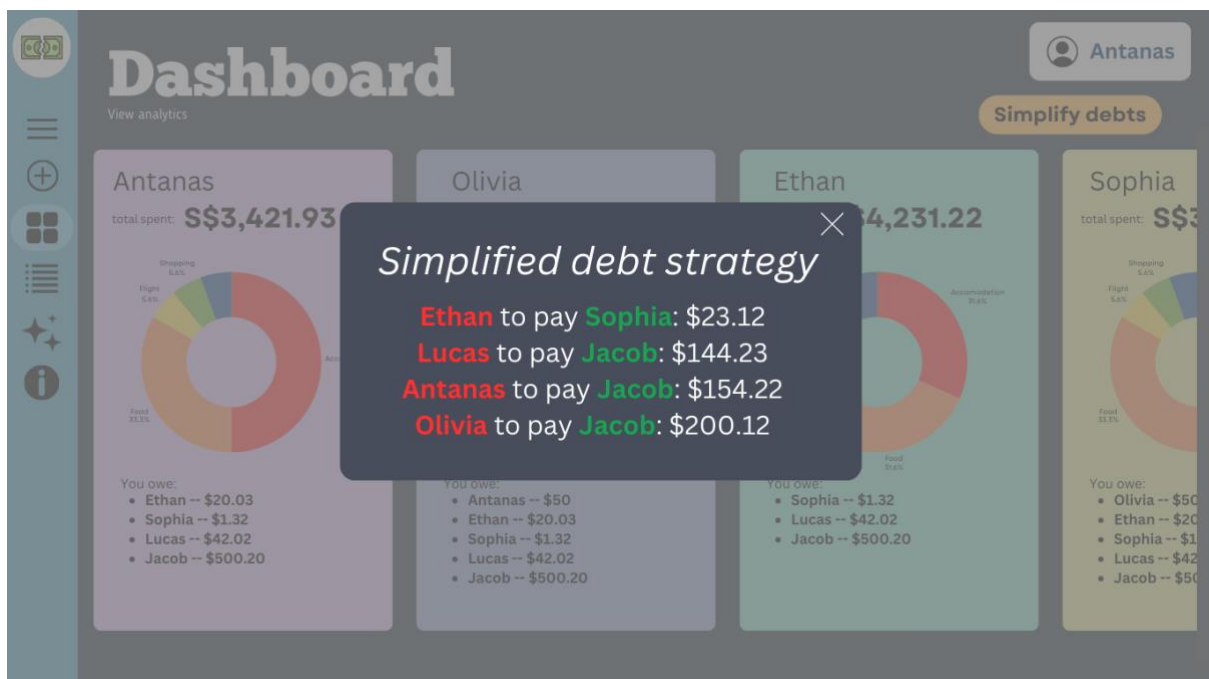
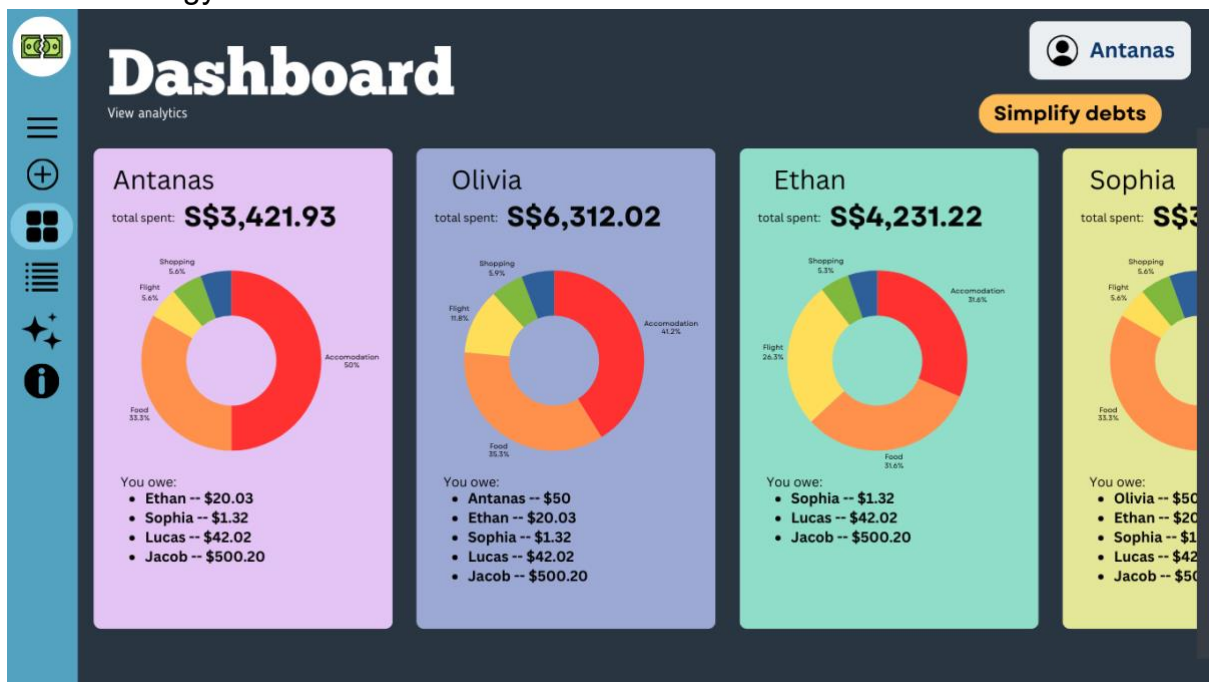
Who paid? ▼

SUBMIT

Dashboard Page

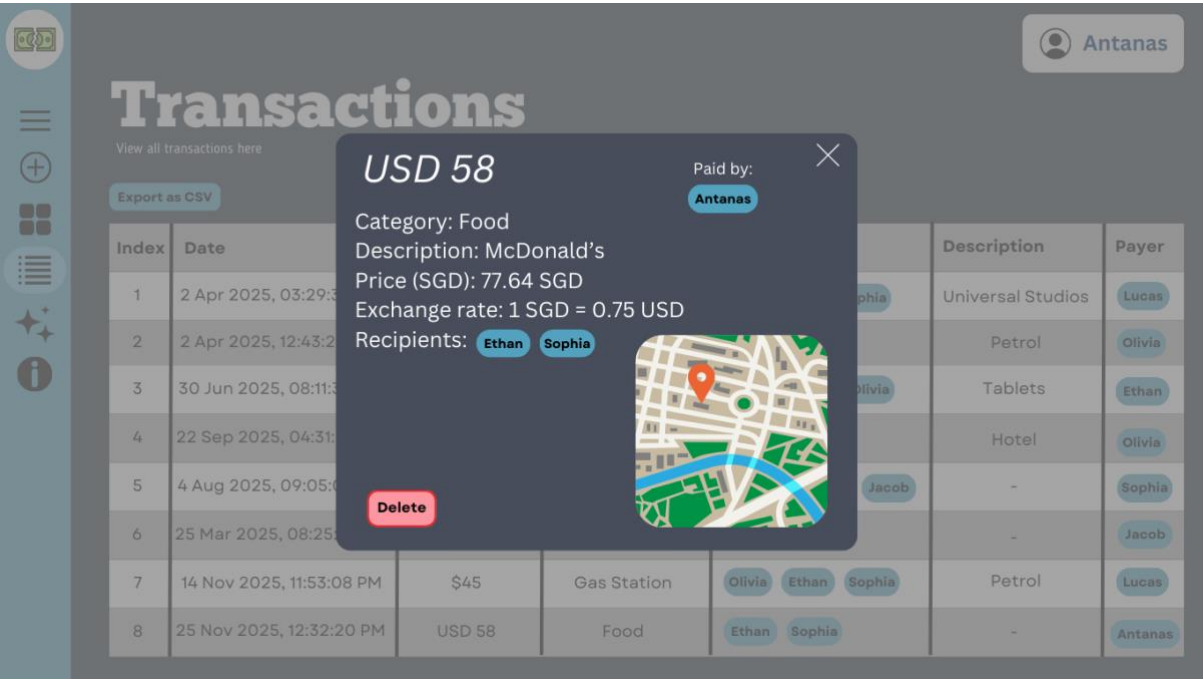
The Dashboard Page will depict analytics information about each trip participants' spending. Each participant's analytics information will be summarised in their individual cards. In each card, it will contain:

1. A pie chart breakdown of spending by category
2. Total that an individual has spent (in the local currency)
3. How much this participant owes other participants based on the reduced debt matrix
4. A Simplify debts button to generate and show the simplified debt settlement strategy



SeeTransactionsPage

The SeeTransactionsPage will depict all transactions logged in the trip. Users can click into each transaction to view it in a more concise format. They can delete the transaction too.



AltineraryPage

The AltineraryPage will display the summarised recommended itinerary planned by generative AI based on the trip's budget and locality.

The screenshot displays the 'Altinerary' app interface. On the left is a vertical sidebar with icons for currency, menu, add, grid, list, stars, and info. The main header features the 'Altinerary' logo, a subtitle 'Tailored Itinerary based on your budget', a user profile 'Antanas', and a disclaimer: '[Disclaimer] This Itinerary was generated by AI'. The itinerary content is as follows:

Trip Overview

- Duration: 14 days
- Destinations: Florida (Orlando, Miami, Key West), Ohio (Cleveland, Columbus, Cincinnati)
- Budget per person: SGD 5000 (~USD 3700)
- Group Size: 7 people

Day 1-2: Travel & Arrival in Florida

- Flight: Singapore → Orlando (~24 hours with layovers)
- Check-in: Airbnb/Hotel in Orlando
- Activity: Light exploration around International Drive, dinner at a local American diner

Day 3-5: Orlando (Theme Parks & Attractions)

- Day 3: Walt Disney World – Magic Kingdom
- Day 4: Universal Studios & Islands of Adventure (Harry Potter World)
- Day 5: Kennedy Space Center (NASA) + Cocoa Beach

Day 6-8: Miami & Key West (Beaches & Nightlife)

- Day 6: Road trip to Miami (~4 hours) → Visit South Beach & Ocean Drive
-

Day 7: Day trip to Key West (Snorkeling, Southernmost Point, Sunset Celebration at Mallory

TripInformationPage

The TripInformationPage will display information about the current trip. Users can see the trip ID here or edit the photo of the display picture.



Trip Information

USA 2025

Trip ID: AFORSC2006 

Current members:

- Antanas
- Olivia
- Lucas
- Ethan
- Sophia
- Jacob

Cities / States:

☒ Florida

☐ Ohio

[+ Add more](#)

From: 1 Jan 2025

To: 15 Jan 2025 

 Antanas



 Edit trip photo

Foreign currency: USD

Local currency: SGD

Trip budget: S\$5,000 / person

EditProfilePage



The image shows a mobile application interface for editing a user profile. On the left is a vertical sidebar with icons for profile, home, add, grid, list, stars, and info. The main area has a dark blue background. At the top, there's a header with a profile picture and the handle '@antanas'. Below this, the 'Display Name' is set to 'Antanas' in a light blue input field with an edit icon. The 'Favourite Colour' section shows a row of seven color swatches: red, orange, yellow (selected with a checkmark), green, light blue, dark blue, and purple. The 'Your Trips' section displays four trip cards with flags and years: USA'25, Japan'23, South Korea'22, and Germany'20. Each card has a red 'X' icon in the top right corner. A fifth card with a plus sign is labeled 'New Trip'.

 @antanas

Display Name: 

Favourite Colour:       

Your Trips:


USA'25


Japan'23


South Korea'22


Germany'20


New Trip