Problem definition

Planning for a trip with friends can be such obstacles. Starting from voting for destinations, planning schedules, managing expenses, and bringing everyone together on the appointed date. Many new group chats are created every time a new trip comes up since there can be different numbers of participants for each trip with a different management system. It is also hard to know the current location of your friends, so when there is a meetup, members can become missing or not show up on time. These problems inspire us to develop a new mobile application called “Vera with Friends” which translates to “time with friends”.

Objectives

Our objective is to make trip arrangements more convenient. After brainstorming, we see that most people come across similar difficulties when planning and going on a trip with their friends. We aim to eliminate those common problems and make your travel experience with your friends seamless. Therefore, our application can simply provide features that will assist users with scheduling and planning trips becoming an easy task.

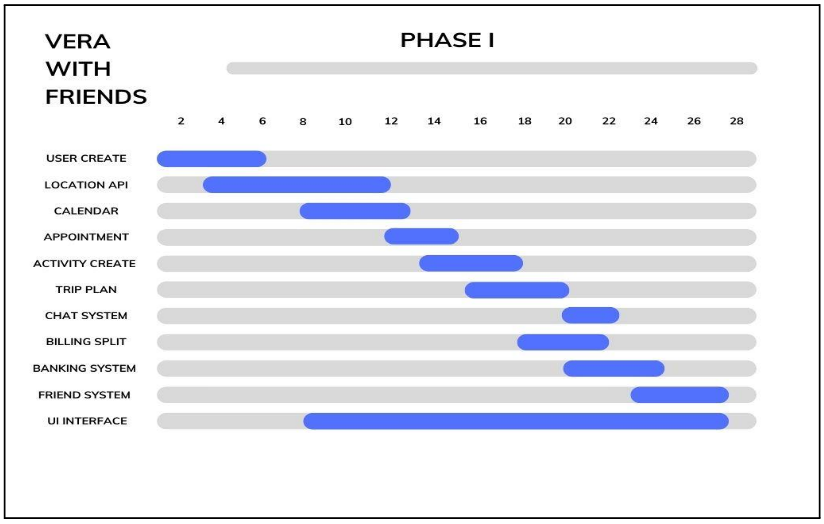
Project Plan

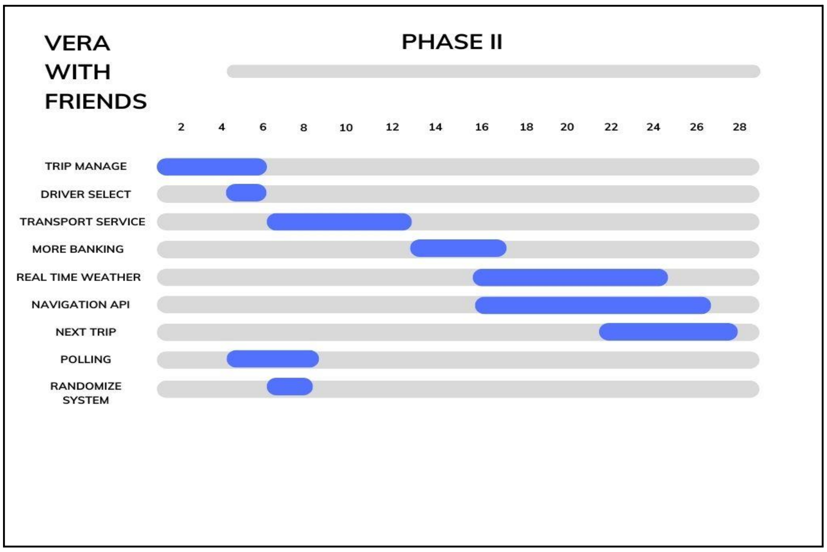
For our project, we began from brainstorming any problems in any activities in our daily life that we individually encounter and . So the idea of the name ‘Vera with Friends’ popped up as an inside joke among us, so if we thought that if we really just came up with a unique name then we have to make something unique too. We ended up designing a whole new system. Starting from the beginning, we found out that we all have a common interest among each other which is traveling as a group, and we came to an agreement that group traveling is really difficult. Especially when traveling with many people. So our model towards the problem is described as “The higher the number of people in the group, then the more difficult the management is going to be”.

This results in a stressful and unenjoyable trip that can make everyone dislike group traveling. Therefore, we listed the number of possibilities of problems that we can encounter while traveling as a group, and we tried to find solutions that our software can solve. For our milestones, we are aiming to follow the incremental model for our software. The first step is to achieve our functional goals of the software in the first phase, which is to provide all the solutions to the problems that we mentioned above. However, the extra features that are not necessary for the software to operate such as “Driver selection” or “Polling System”. Once the first phase of our prototype is fully developed by the backend developers, the next step is to pass our software to the visual team where we have UI designers and artists to make our software more appealing to the public users. Once this part is completed, the next objective is to find the appealing factor which cannot be anyone, but from the users themselves. Therefore, the beta stage is required for our milestone to make sure that our software is suitable for its purpose and attracts the majority of targeted users.

When the first prototype of our software is ready for a public release, we call this the beta stage. The beta stage allows the full deployment of our software mainly towards the public users. The public is non-restricted to the target group of the testers, so any experience or backgrounds are all welcome to see what our software is capable of, which they must accept the risk of performance issues and errors they might encounter during the use of the software. During this phase, we follow the incremental model that we start developing the second phase in the background. There is some time for the feedback to be received. After the second phase is completed, this phase will be deployed and the feedback will be enough for the developers to understand the further requirement to improve the software according to the feedback given from them. Eventually, once the second stage is deployed and receives the feedback from the users, then the incremental model continues and the third phase can be implemented for any new features that will be added to the software continuously.

Gantt Chart:





First Phase -> Beta Deployment -> Feedback Improvement -> First Phase Full Deployment-> Feedback Improvement -> Second Phase Development -> Second Phase Deployment -> Feedback Improvement -> Bug Fixing and Performance Tweak

User Requirements

Application user requirements

Using mobile application interface

1. **Plan/schedule events**

   Users can plan trips together with their friends on the application. Destination and booking can be noted in the schedule.

2. **User profile**

   User can input their personal details such as user identification, profile picture, contact information, allergy, account number/QR code (optional), emergency contact and whether they have a driving license or not

3. **Friends management**

   Users can add new friends by user ID. Users can also block friends that they no longer keep contact with. Facebook sign-in allows the user to connect to their friends who also use this app and sign up through Facebook authentication. Direct messaging allows one to one messaging between their friends just like a personal conversation. Status updates can make users express their thoughts to their friends in the application. This is for the case if their friends share similar thoughts on their wishlist of where to go, this can be a fun feature to travel with friends easily. Accepting and declining invitation is to avoid users mistakenly selected to a trip without their consent.

4. **Expense management**

   Expenses for the trip can be recorded in the application which will later be split among the members. The application will calculate and keep track of who has or hasn't paid yet. Users can choose to divide the expense among everyone or just some selective members.

5. **Driver random selection**

   This feature can be used to select the driver. The driver will be randomly selected among the members of the group.

6. **Receive notification**

   Activities or updates regarding the trip that the user joined will be sent as notifications to users.

7. **Group conversation**

   Users will be able to communicate with friends through this application. The application will provide services such as chat, voice calls and video calls. The group conversation will be deleted a week after the trip ends by default unless the user decides to keep the chat history.

8. **Poll**

   Polls are the feature for voting. The vote can be anonymous and can be for any purpose.

9. **Location tracking**

   The application will track the user's location from their GPS while they are on a trip. Users can choose to enable or disable this feature. This is typically the case for meeting up purposes.

System Requirements

Functional requirements

1. **Plan/schedule events**

   Actors Involved: User

   Description:

   - Users will be able to set an event with dates, locations, pictures, notes and people who will participate.
   - Once the event is added, it will show on the calendar.
   - Calendars can be viewed daily, weekly and monthly.
   - Events can be edited later by any member.
   - If the user has a schedule overlap then a warning will pop up.

2. **User profile**

   Actors Involved: Users

   Description:

   - Users will have a profile page where they can input their name, profile picture, user ID, account number/QR code (optional) and whether they have a driving license.

3. **Friend management**

   Actors Involved: User

   Description:

   - On the friend page, there will be an option to add friends, users will have to input friends' user ID.
   - Users can choose to block friends that they no longer keep in contact with.
   - Friends using this application can be imported from third party service such as Facebook (suggested friends you may know)
   - Status update allows users to express their own thoughts about wishlist or upcoming trips, so others may have a similar thought and promote group traveling.
   - Members can choose to accept or decline when invited to an upcoming trip, this avoids an automatically added system where members are automatically added to a trip they did not want to participate in.

4. **Expense management**

   Actors Involved: Users

   Description:

   - Users can input the costs that are used on the trip and choose options for other payback. If the user allows bank transfer then as default the system will take the account number/QR code input in the profile. If they didn't input in the profile or what to change then they must upload a QR code or input their account number.
   - By default, the application will split the bill among everyone in the group but the user can also choose to divide for just some selective members. In the case where the cost shouldn't be divided equally, the user can manually input the number for each member.
   - There will be a debt profile for each user, there will be a list of who the user owed, with the number, who to pay back to and payment options; bank transfer and cash. If users choose to pay back by bank transfer, they must upload a slip as evidence. If choose to pay back by cash then the application will notify the lender to confirm the payment.

- There will be a lender profile for each user, there will be a list of who owes the user, by how much and status of whether they pay back yet.

5. **Driver selection**

   Actors Involved: user

   Description

   - For this function, the application will randomly select members in the group that have a driving license. The number selected will be determined by the input of the user.

6. **Receive notification**

   Actors Involved: User

   Description:

   - Notifications can be sent by other users in the chatroom
   - The system sends reminders or updates to users

7. **Group conversation**

   Actors Involved: User

   Description:

   - There will be a chat room where all members can communicate. Users can choose to have a group voice call or a group video call.
   - The chatroom will only be available for a week after the trip/event ends by default. When it reaches the initial expiration date of the chat then they can choose to extend it but it can’t be extended for more than a month.

8. **Poll**

   Actors Involved: User

   Description:

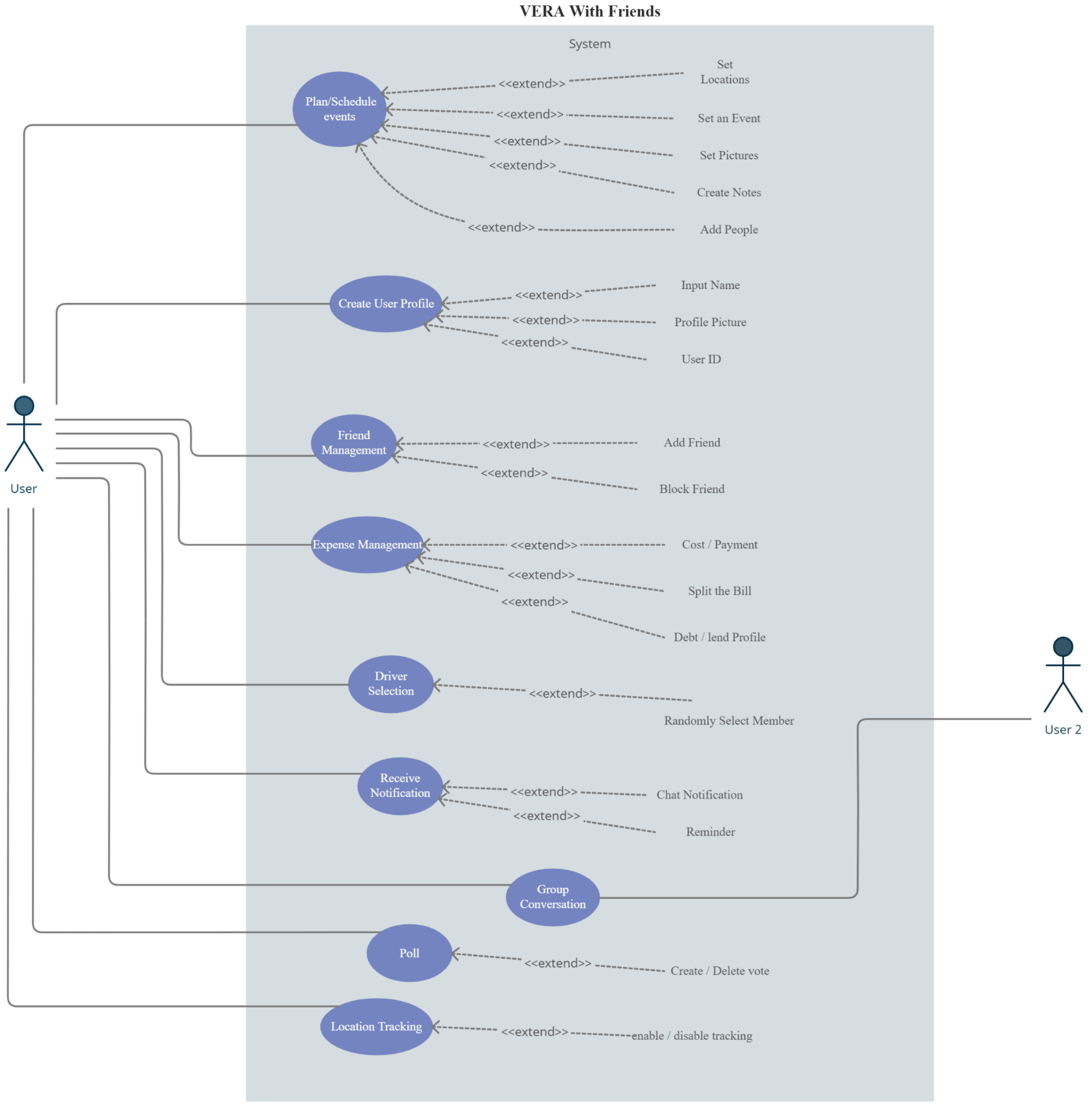
   - Users can choose to use this feature if they want to vote
   - Voting can be used known or anonymously depending on the user that creates the poll

9. **Location tracking**

   Actors Involved: User

   Description:

   - Users can choose whether to enable or disable this feature
   - The application will start to track the user's location once the trip/event start
   - The user profile picture will be used as an avatar on the map to show the user's current location
   - If the user moves at a speed of more than 40 km then it will show the icon of a car.



Non-functional requirements

1. ***Performance Requirements***
   - The maximum time for a reload most updated data: 3 seconds
   - Average time delay for 1 notification to be sent to the user by the system: 2 seconds
   - Average time for instant chat message: 1 second
2. ***Safety requirements***
   - Password for the account must be a strong password. Must contain at least 8 characters with at least 1 capital letter, 1 number and 1 special character.
3. ***Security Requirements***
   - The user's current location should be protected and not be used for other purposes.

- User personal information must be kept confidential
- To enter the system, the user must log in with a username and password

4. ***Software qualities***
   - Calculate in the expense management must be fully accurate
   - Data should be shown the same across all the devices

System constraints

Hardware requirements

- Group video calls require a camera and microphone.
- The application requires at least 300MB of storage space.

Software requirements

- For IOS, our program only supports IOS version 16 or above.
- For Android, our program only supports Android 12 or above.

Domain requirements

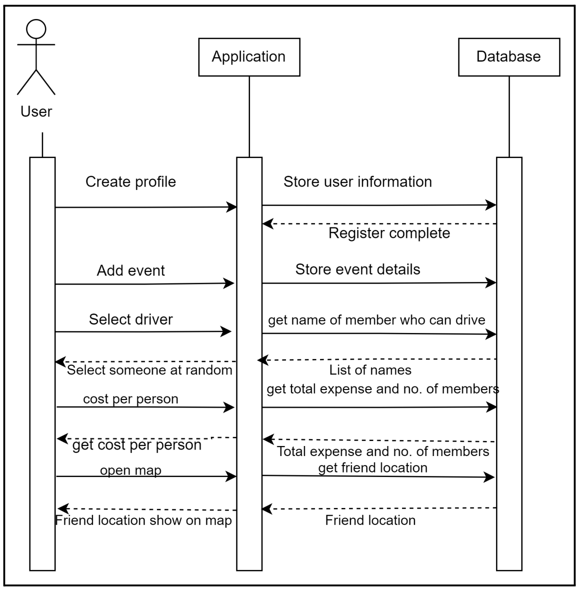
1. Absolute safety of users' private data must be guaranteed.
2. Expense calculation must be calculated to two decimal places

Design specification document

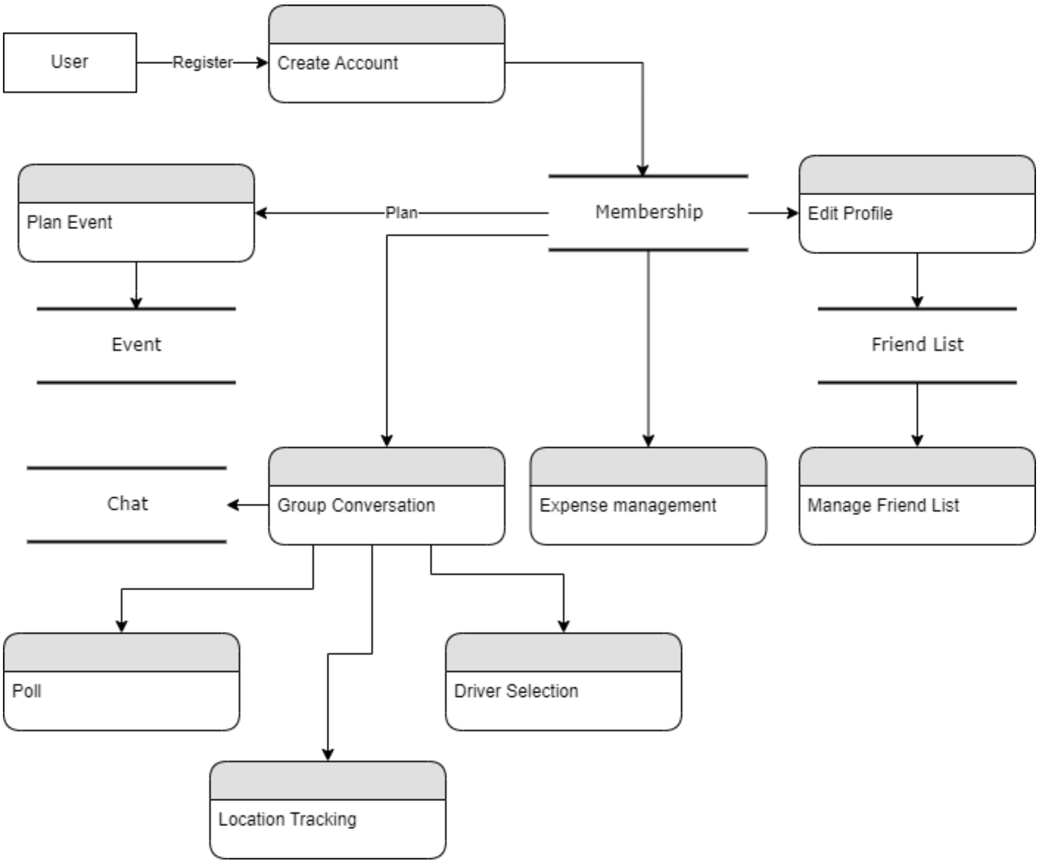
Software architecture

We will use an Incremental process model to develop our software since we can use the early increment to check whether the requirement is enough or not, and we can add more requirements later. Moreover, we can gather the comments of the user to improve our software in future delivery of increments while developing the next increment, so our project has a low risk of failing. The incremental model also allows us to test functionality of the system and enables us to deploy the software earlier than other models.

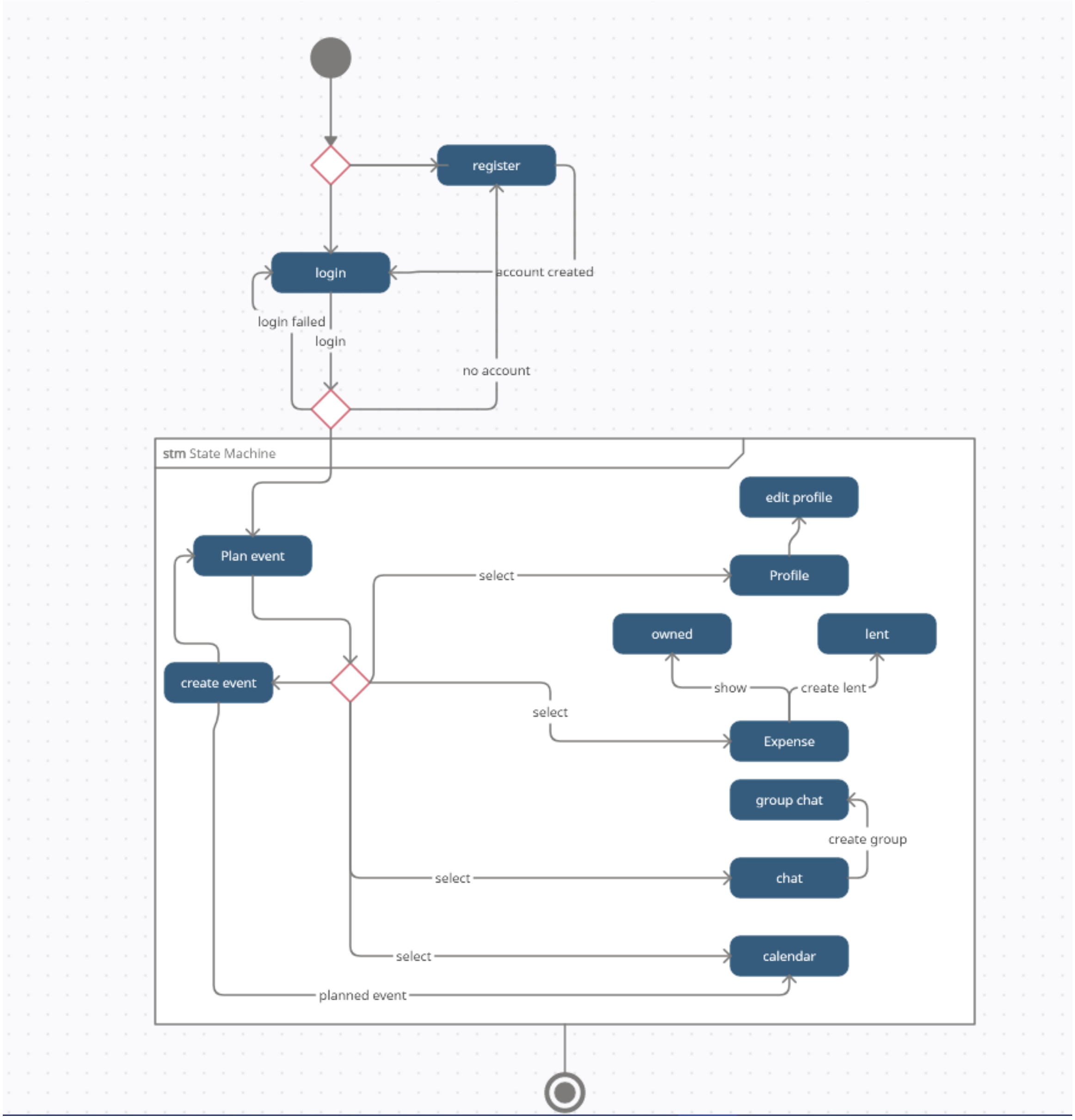
Sequence diagram



DFD diagram


User
Register
Create Account
Plan Event
Plan
Membership
Edit Profile
Event
Friend List
Chat
Group Conversation
Expense management
Manage Friend List
Poll
Driver Selection
Location Tracking

State machine diagram



Technical documentation

Programming languages:

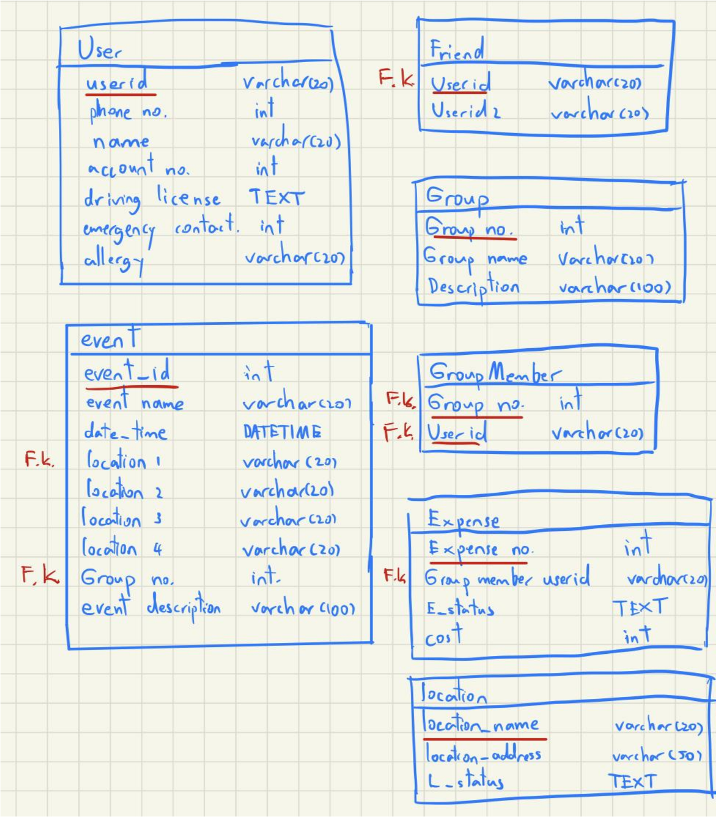
HTML5 for front-end programming : HTML5 enables programmers to produce more organized and consistent code and it can be used across multiple platforms which means that you can apply the same code to platforms such as App store and Play store. Moreover, it contains a variety of functions that improve the performance of the application in terms of speed, less load, geolocation services (help track the location of the user), etc.

Java Node.Js for back-end programming : Java Node.js mostly utilizes JavaScript as the programming language. As a result, corporations save a lot of money by cutting down on development time. Developers may easily transfer data across the server-side and client-side thanks to Node.js.

Tools and environments

- Visual Studio Code for coding
- Uizard for UI design
- MySQL for Database management
- Github for version control and access control

Database system:



	User	
	userid	varchar(20)
	phone no.	int
	name	varchar(20)
	account no.	int
	driving license	TEXT
	emergency contact.	int
	allergy	varchar(20)

	event	
	event_id	int
	event name	varchar(20)
	date_time	DATETIME
F.k.	location 1	varchar (20)
	location 2	varchar(20)
	location 3	varchar (20)
	location 4	varchar (20)
F.K	Group no.	int.
	event description	varchar (100)

	Friend	
F.k	Userid	varchar(20)
	Userid 2	varchar (20)

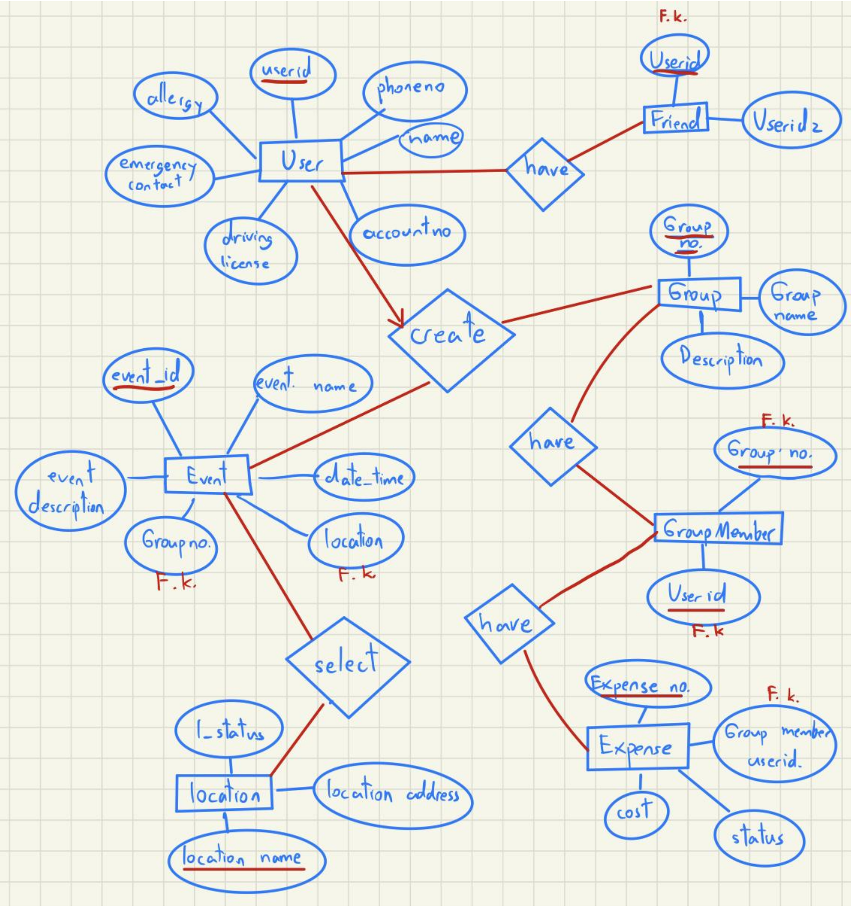
	Group	
	Group no.	int
	Group name	Varchar(20)
	Description	varchar (100)

	Group Member	
F.k.	Group no.	int
F.k	User id	varchar(20)

	Expense	
	Expense no.	int
F.k	Group member userid	varchar(20)
	E_status	TEXT
	cost	int

	location	
	location_name	varchar (20)
	location-address	varchar (50)
	L_status	TEXT

E-R Diagram



Data schema : User{userid,phone_no.,name,account_no,driving license,emergency contact,allergy}

Event{event_id,eventname,date_time,location1,location2,location3,location4,Groupno, Event_description}

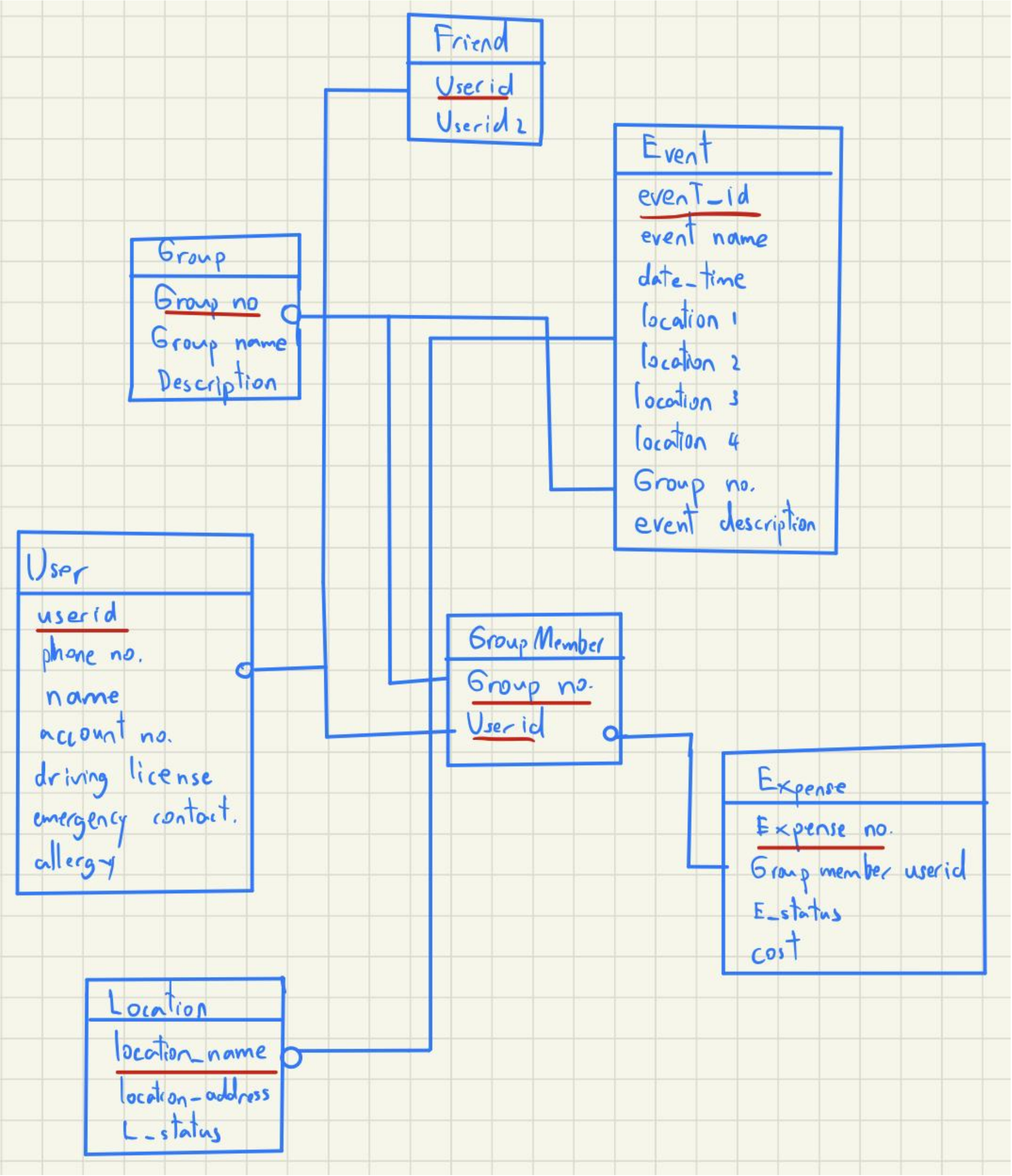
Friend{Userid,Userid2}

Group{Group no.,Groupname,Description}

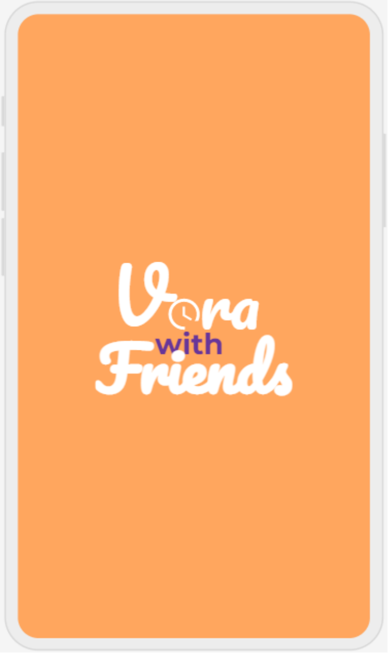
GroupMember{Groupno.,Userid}

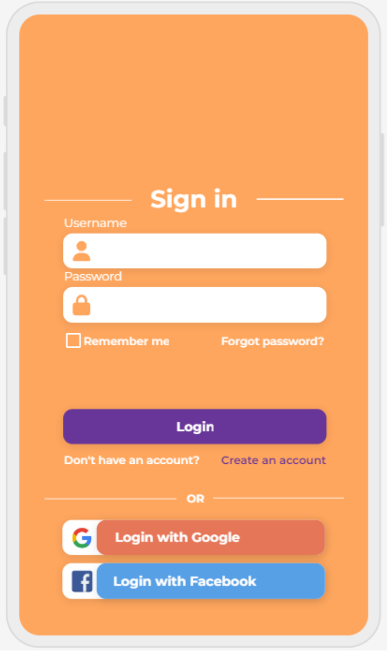
Expense{Expenseno,Groupmember_userid,status,cost}

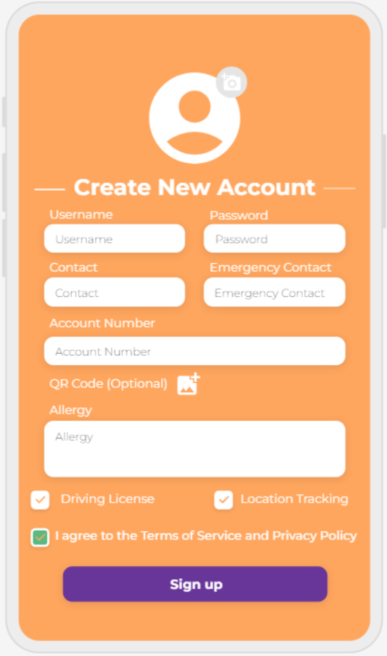
Location{location_name,location_address,status}


Friend
Userid
Userid2
Event
evenT_id
event name
date_time
location 1
location 2
location 3
location 4
Group no.
event description
Group
Group no
Group name
Description
User
userid
phone no.
name
account no.
driving license
emergency contact.
allergy
Group Member
Group no.
User id
Expense
Expense no.
Group member userid
E_status
cost
Location
location_name
location_address
L_status

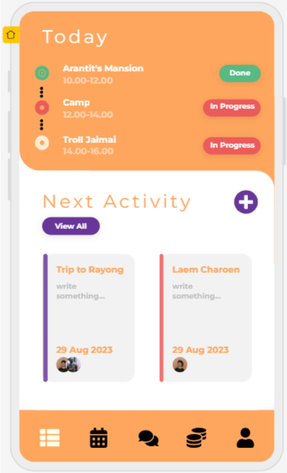
User interface document

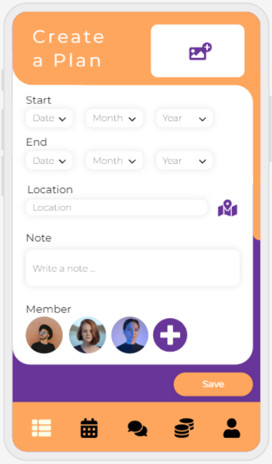


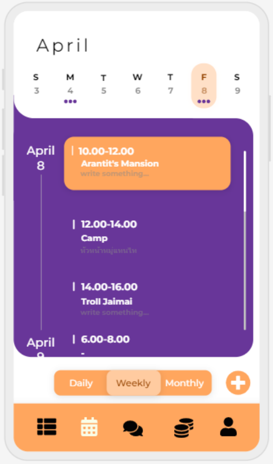




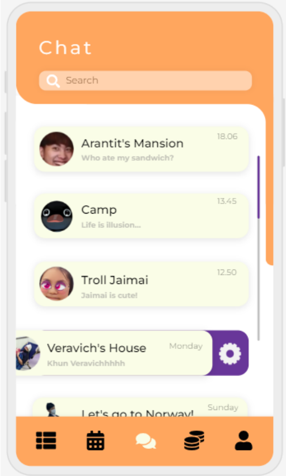
The screen on the left is the screen that will show up when the user first opens the application. The middle screen is the login screen. If a user already has an account then they can just input their username and password to login. They have a choice to login using google account and facebook as well. If the user doesn't have an account, they can create a new account which will take the user to the screen on the right. The screen on the right is the screen for new users. They will be asked to input the personal information that is necessary for creating a user profile. Users will be asked to agree with our term of service and privacy policy.

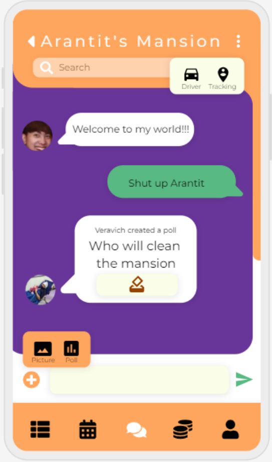






When the user already logins, the application will take you to the screen on the left which is the screen that shows the upcoming event and the user can also view all the events. When the user wants to create a new event, they can press the add button which will take them to the next screen in the middle. The screen in the middle is for creating a new plan. The user can input the details of the plan and add their friends to join their plan. The screen on the right is the calendar which will contain the daily, weekly and monthly plans.

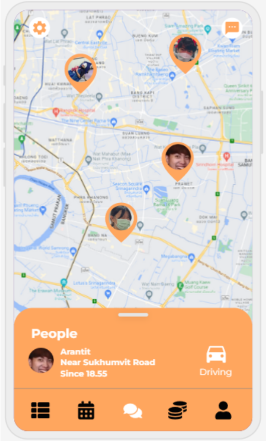




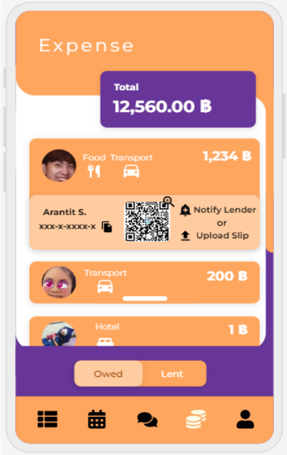
Welcome to my world!!!

Shut up Arantit

Veravich created a poll
Who will clean the mansion



The screen on the left is the overall group conversation page. By sliding the chat left, you can go to the setting of that chat. When pressed on the chat, it will take the user to the middle screen which is the specific group chat. The user can chat with their friends who are involved in the plan. There are also functions to create a poll, select the driver, and track the location of their friends. When pressing the location tracking icon, the application will take the user to the screen on the right. There the user can see the location of their friends real-time. If there is a person who moves at a speed of more than 40 km then it will show the icon of a car on the right of the name..



Total
12,560.00 ฿

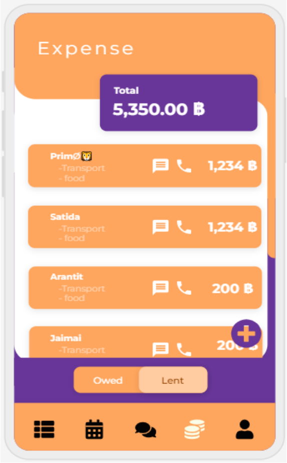
Food Transport 1,234 ฿

Arantit S.
xxx-x-xxxx-x

Notify Lender or Upload Slip

Transport 200 ฿

Hotel 1 ฿



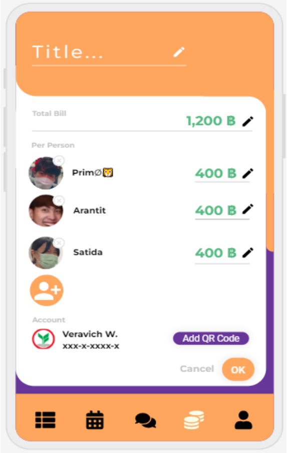
Total
5,350.00 ฿

PrimØ🦁 -Transport - food 1,234 ฿

Satida -Transport - food 1,234 ฿

Arantit -Transport - food 200 ฿

Jaimai -Transport 20



Total Bill 1,200 ฿

Per Person

PrimØ🦁 400 ฿

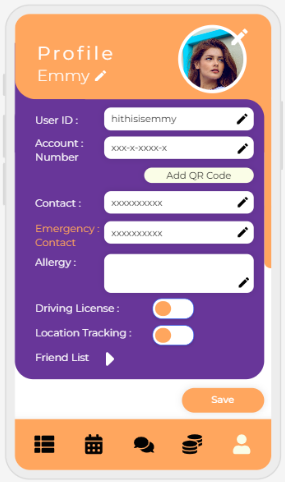
Arantit 400 ฿

Satida 400 ฿

Account

Veravich W.
xxx-x-xxxx-x

These screens involve expense management. The screen on the left shows the information of who the user owed. It will provide the user with the amount and the account number and/or QR code of the person the user has to pay back to. The middle screen shows the list of friends who the user lent the money to and by how much. The right screen will be the screen where users can create new expenses. They will have to input the title and total bill. By default, the application will divide the bill among everyone in the group and the account will be the one that the user put in their user profile. Users can edit if they want to only divide the expenses among selected members or each person only pays a certain amount. They can choose to change the account number for this specific expanse collection.



Emmy

User ID : hithisisemmy

Account : Number xxx-x-xxxx-x

Contact : xxxxxxxxxx

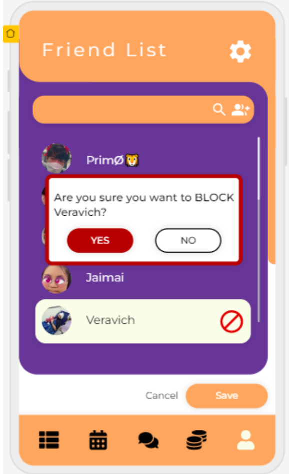
Emergency : Contact xxxxxxxxxx

Allergy :

Driving License :

Location Tracking :

Friend List



PrimØ🦁

Are you sure you want to BLOCK Veravich?

Jaimai

Veravich

The profile pages allow you to see and edit the user information such as User ID, account number, contact,etc. Moreover, in the last line of the profile page you can click the arrow next to the friend list then it will bring you to the friend list page which you can view the list of your friends and manage the friend status as BLOCK or keep as friend then you can save the edited information.