

SETTLE GROUP EXPENSES

A Project Report

Submitted in partial fulfillment of the
Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Sumit Anant Bane

Seat Number

Under the esteemed guidance of

Ms. Larissa Pegado

Assistant Professor



DEPARTMENT OF INFORMATION TECHNOLOGY

PTVA'SATHAYE COLLEGE

(Affiliated to University of Mumbai)

MUMBAI, 400057

MAHARASHTRA

2019-2020

PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

PNR No.:

Seat no:

1. Name of the Student: Sumit Anant Bane

2. Title of the Project: Settle Group Expenses

3. Name of the Guide: Larissa Pegado

4. Teaching experience of the Guide: _____

5. Is this your first submission? Yes ☐ No ☐

Signature of the Student

Signature of the Guide

Date:

Date:

Signature of the Coordinator

Date:

PTVA's SATHAYE COLLEGE
(Affiliated to University of Mumbai)
MUMBAI-MAHARASHTRA-400057

DEPARTMENT OF INFORMATION TECHNOLOGY



CERTIFICATE

This is to certify that the project entitled, "**Settle Group Expenses**", is bona fide work of **Sumit Anant Bane** bearing Seat no: _____ submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

Internal Guide

Coordinator

External Examiner

Date :

College Seal

ABSTRACT

The purpose of this project is to reduce the burden of remembering the settlements among the group members. This application will not only settle the expenses but also have additional features like maintaining the budgets and reminding the members about their To-Do list.

Because of this project the problem miscalculation is resolved. People don't need to spend extra time after their trip for this work.

This application was developed as I came across the same problem multiple times whenever I went on a trip with my friends. We needed to first remember our expenses and then settle the debts which were quite hefty tasks.

This application is useful for each and every person in the world who plans a trip with his/her friends/colleagues etc. as most of the time people choose to pay their friend's/colleague's expense whenever there is a crisis of having change

This project meets the objectives I specified in this report. It works according to my plan and my design criteria.

ACKNOWLEDGEMENT

I would like to thank all those who made this project possible and a successful one. It is my earnest endeavor to express my sincere thanks to the faculty for their kind co-operation, help and never ending support.

I would take this opportunity and express gratitude to my Professor **Ms. Larissa Pegado** for her support and encouragement, without which the successful completion of this project would have been impossible.

Without her encouragement, constant guidance, interest and her innovative ideas I would not have completed this project. It was because of her knowledge and creative ideas that I could enhance the working efficiency of my project.

I would also give credits to all other professors and staff of I.T. Department who directly or indirectly helped me with this project.

I would like to thank my parents for providing everything to me that was essential for this project and last but not the least, I would appreciate the help which I received from my friends and classmates.

DECLARATION

I hereby declare that the project entitled, “**Settle Group Expenses**” done at PTVA’s **Sathaye College**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to another university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as a final semester project as a part of our curriculum.

Sumit A. Bane

TABLE OF CONTENTS

Chapter 1	Introduction	11
1.	Background:	11
2.	Objectives:	11
3.	Purpose:	12
4.	Stakeholders:	12
5.	Scope:	12
5.1.	In Scope:	12
5.2.	Out of Scope:	13
6.	Deliverables:	13
Chapter 2	Literature Review	14
1.	Technologies used:	14
2.	Techniques implemented:	14
3.	Difficulties faced:	15
4.	Features/Functionalities:	15
5.	Comparative Analysis:	15
6.	Areas where I can improve:	16
Chapter 3	Requirements and Analysis	17
1.	Problem Definition:	17
2.	Requirement Specifications:	17
3.	Planning and Scheduling	18
3.1.	Activity Table	18
3.2.	Gantt Chart	18
3.3.	PERT Chart:	19
3.3.1	Table	19
3.3.2	Chart	19
4.	Hardware & Software requirements:	20
4.1	Hardware:	20
4.2.	Software:	20
5.	Preliminary Product Description:	20
6.	Conceptual Models:	21
6.1.	ER Diagram:	21
6.2.	Data Dictionary	21
6.2.1	Table: Group	21
6.2.2	Table: Members	21
6.2.3	Table: To-Do	22
6.2.4	Table: Advance Payments	22
6.2.5	Table: Expenses	22
6.2.6	Table: Expense Status	22
6.2.7	Table: Multiple Members Expenses	23

6.2.8	Table: Multiple Member Expense Bridge	23
6.3.	Context Diagram	23
6.3.1	Level 0	23
6.3.2	Dataflow Diagram	24
6.4.	Use Case Diagram	25
6.5.	Activity diagram	26
Chapter 4	System Design	27
1.	Create Group:	27
1.1	Algorithm:	27
1.2	Flowchart:	28
1.3	Validations:	28
2.	Add Members:	29
2.1	Algorithm:	29
2.2	Flowchart:	30
2.3	Validations:	30
3.	Add Expense:	31
3.1	Algorithm:	31
3.2	Flowchart:	32
3.3	Validations:	32
4.	Add Advance Payment:	33
4.1	Algorithm:	33
4.2	Flowchart:	34
4.3	Validations:	34
Chapter 5	Implementation and Testing.....	35
1.	Screenshots and their description:	35
1.1	Create Group:	35
1.1.1	Image:	35
1.1.2	Details:	35
1.2	Add Members:	35
1.2.1	Image:	36
1.2.2	Details:	36
1.3	Add To-Do:	36
1.3.1	Image:	37
1.3.2	Details:	37
1.4	Add Expense:	37
1.4.1	Image:	38
1.4.2	Details:	38
1.5	Add Advance Payment:	39
1.5.1	Image:	39
1.5.2	Details:	39
1.6	Transactions:.....	40

1.6.1	Image:	40
1.6.2	Details:	40
1.7	Settle Debts:	41
1.7.1	Image:	41
1.7.2	Details:	41
2.	Code:	42
3.	Test Cases:	44
3.1	Functional:	44
3.1.1	Module - Create Group:	44
3.1.2	Module - Home Activity:	45
3.1.3	Module - Add Member:	45
3.1.4	Module - Member Details:	48
3.1.5	Module - Add Expense:	48
3.1.6	Module - Expense Details:	49
3.1.7	Module - Multiple Members Expense:	51
3.1.8	Module - Advance Payment:	54
3.1.9	Module - Transactions:	56
3.1.10	Module – Settle Debts:	57
3.2	Non-Functional Testing:	58
Chapter 6	Results	59
1.	Test Report:	59
2.	Project Summary Report:	59
2.1	Overview:	59
2.2	The Problem:	59
2.3	The Solution:	59
2.4	Highlights:	60
Chapter 7	Conclusions	61
1.	Significance of the system:	61
2.	Limitations of the system:	61
3.	Future Scope of the project:	61
	Bibliography	62

LIST OF TABLES

<i>Activity Table</i>	18
<i>PERT Chart (Table)</i>	19
<i>Data Dictionary</i>	21

LIST OF FIGURES

<i>Gantt Chart</i>	18
<i>PERT Chart</i>	19
<i>ER Diagram</i>	21
<i>Context Diagram (Level 0)</i>	23
<i>Dataflow Diagram</i>	24
<i>Use Case Diagram</i>	25
<i>Activity Diagram</i>	26

Chapter 1 Introduction

1. Background:

“Travelling – It leaves you speechless, then turns you into a storyteller” - Ibn Battuta
(an explorer who widely travelled the medieval world)

Everyone loves to travel and when it comes to vacations, people travel across the globe with their friends and family.

Whenever I went on a trip with my friends we needed to spend money on various occasions. Whether it was buying tickets or booking hotel rooms, it was difficult for us to contribute equal money every time. So one or few members used to pay for a particular thing at a time which led to another task not complex yet confusing of settling the expenses at the end of the trip.

These settlements left me with the idea to create any software or application for the same. I am referring an application named “Settle Up” which is available on Google Play Store.

My project is meant to reduce the confusions which arise at the end of the settlement process. It will generate a detailed report on all the expenditures and their settlement.

2. Objectives:

The main objectives of my project are as follows

- To split all the expenses among the members and settle them at the end of the trip.
- To provide a detailed report at the end, about which member should pay how much to which member.
- To maintain the date and time along with the expenditures throughout the trip.

- To remind the members about their budgets if they are spending more than their target amount which was set by them earlier at the beginning of the trip.

3. Purpose:

- The main purpose behind this project was to diminish the chances of arguments which could arise at the end while settling with debts.
- Group members can add their contributions during expenditures and no one has to remember the amount and the time.
- As people trust technology more than humans there will be no need to cross-check or memorize the expenses during the trip.

4. Stakeholders:

- As this project is not for a particular group or an organization people from any part of the world can use it.

5. Scope:

5.1. In Scope:

- Users can completely rely on it as the logic behind the calculations is going to remain the same.
- At the end a summary of settlements will be provided to the users which is what exactly the need.
- For a particular user this product will come in play whenever he/she sets out for a journey which is not quite often.

5.2. Out of Scope:

- If a user forgets to enter an expense of any member at a certain point and he/she realizes it later then he/she may not get the exact time in the report when the expense was done.

6. Deliverables:

- At the end, a report will be generated which will give a brief summary about the expenditures during the trip and their settlements.

Chapter 2 Literature Review

Whenever we are trying something new, we try to get help from people who are experts or have an experience in that field. From 'A kid learning to ride a bicycle from his father' to 'A student gathering information about some course from an alumni' each of us needs to follow in someone's footsteps. Same thing applies when we are working on a project. We need to study about similar projects in past which is nothing but a Literature Review.

A Literature Review is a comprehensive summary of previous research on a topic. It surveys articles, books and other sources relevant to a particular area of research.

I am referring 'Settle Up', 'Group Expense' and 'Tricount' for my project. Among these applications I am studying the 'Settle Up' app deeply.

It seems like these three were also inspired by each other and tried to excel in the areas where others were lacking. And my job will also be the same.

1. Technologies used:

- The software /technologies used by 'Settle Up' are 'Firebase Realtime Database' on the backend, 'Kotlin', 'MVP architecture' and 'Rx Java' in the application.

2. Techniques implemented:

- 'Settle Up' is managed by a team of 5 people. They have a meeting once in a week where they discuss what to do next. They put tasks into Trello boards.

3. Difficulties faced:

- Initially 'Settle Up' made use of 'Google App Engine' for data storage. They had issues while migrating from App Engine to Firebase Runtime Database.

4. Features/Functionalities:

- The 'Group Expense' app has an 'Add Advance Payment' option. Members can settle or pay a part of the debt to other members throughout the trip.
- The 'Settle Up' app has a better UI than other two apps. It displays circles with names of members and their debts within them. The larger the circle, the bigger the debt amount is.
- The 'Tricount' app has online payment facility with 'Paypal', but currencies from many countries are not compatible with the Paypal payment currency.

5. Comparative Analysis:

- The 'Settle Up' and 'Tricount' apps have a first-time tutorial but 'Group Expense' doesn't.
- Online payment is available in 'Tricount' but it is not in the basic versions of 'Settle Up' and 'Group Expense'.
- 'Tricount' and 'Settle Up' have both IOS and AOS versions. They both have an option to add an image of receipt during an expense.
- 'Tricount' was the first to have synchronization in the app on different devices, followed by 'Settle Up'. 'Group Expense' still doesn't have it.

6. Areas where I can improve:

- Many times people spend a lot, even on unnecessary things. A notification can be sent to their device whenever they are spending more than they should be. [Their expense limit can be set earlier by the user].
- Similarly, a notification can be sent at a particular time which will remind them to buy stuff they had already decided to.

Chapter 3 Requirements and Analysis

1. Problem Definition:

The 'Group Expense', 'Settle Up' and 'Tricount' applications provide a lot of features for settling the debts already. My application will be providing some additional features like 'Reminding the users about their budgets', 'providing a basic calculator' and a 'To Do list'.

2. Requirement Specifications:

Users will be requiring the following features:

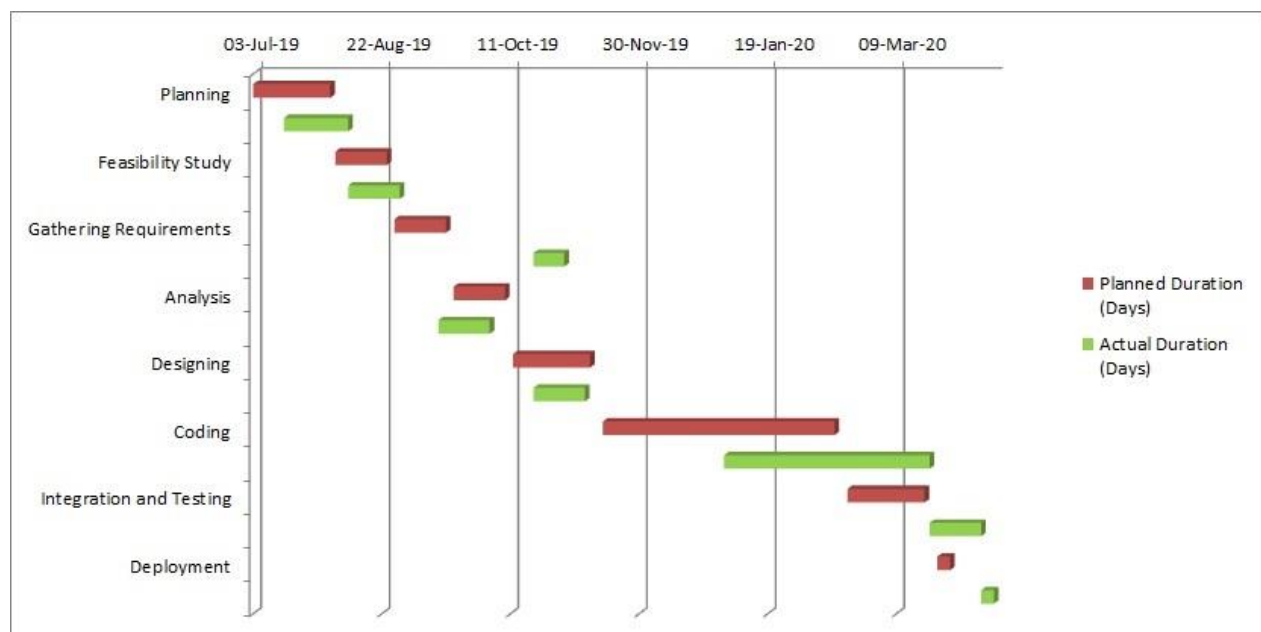
- Adding in all the expenses and then splitting it among each member.
- 'Who will pay to whom' and how much?
- Making payments (settling the debts) during the trip.
- First time tutorial/ walkthrough.

3. Planning and Scheduling

3.1. Activity Table

Activity no.	Activities	Planned Start Date	Planned Duration (Days)	Actual Start Date	Actual Duration (Days)
1	Planning	03-Jul-19	30	15-Jul-19	25
2	Feasibility Study	04-Aug-19	20	09-Aug-19	20
3	Gathering Requirements	27-Aug-19	20	20-Oct-19	12
4	Analysis	19-Sep-19	20	13-Sep-19	20
5	Designing	12-Oct-19	30	20-Oct-19	20
6	Coding	16-Nov-19	90	02-Jan-20	80
7	Integration and Testing	19-Feb-20	30	22-Mar-20	20
8	Deployment	25-Mar-20	5	11-Apr-20	5

3.2. Gantt Chart

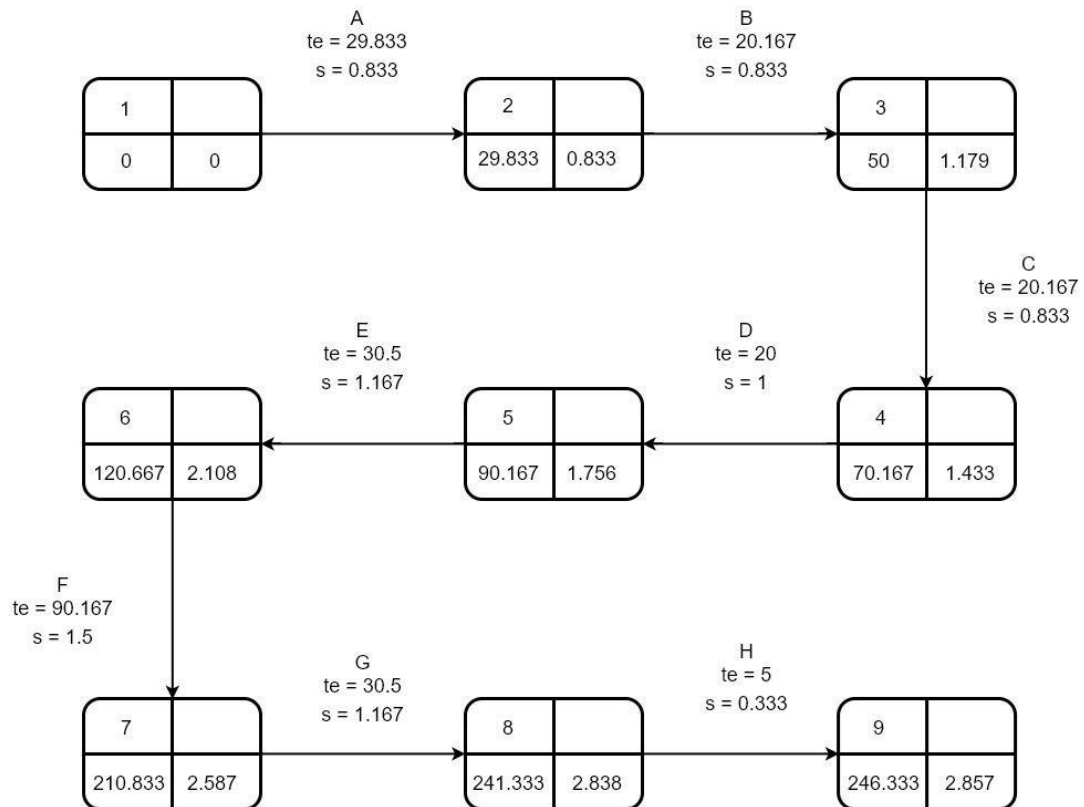


3.3. PERT Chart:

3.3.1 Table

Activity no.	Activities	Optimistic time (a)	Most Likely time (m)	Pessimistic time (b)	Expected time (te)	Standard Deviation (s)
A	Planning	27	30	32	29.833333	0.83333333
B	Feasibility Study	18	20	23	20.166667	0.83333333
C	Gathering Requirements	18	20	23	20.166667	0.83333333
D	Analysis	17	20	23	20	1
E	Designing	28	30	35	30.5	1.16666667
F	Coding	86	90	95	90.166667	1.5
G	Testing & Debugging	28	30	35	30.5	1.16666667
H	Deployment	4	5	6	5	0.33333333

3.3.2 Chart



4. Hardware& Software requirements:

4.1 Hardware:

My system's specifications are as follows:

- Intel i5 9th generation 9300H.
- 8GB memory.
- Storage - 500 GB SSD
- Graphics - 4GB Nvidia GeForce GTX 1650

4.2. Software:

Android Studio 3.2.1 (IDE), Android sdk, Emulator.

- **Frontend** – xml
- **Backend** – java, sqlite database

5. Preliminary Product Description:

The main job of my application will be to add all the expenses of the members during the trip and then settling the debts at the end. It will tell the user, how to settle the debts among the members. It will display who should pay to whom.

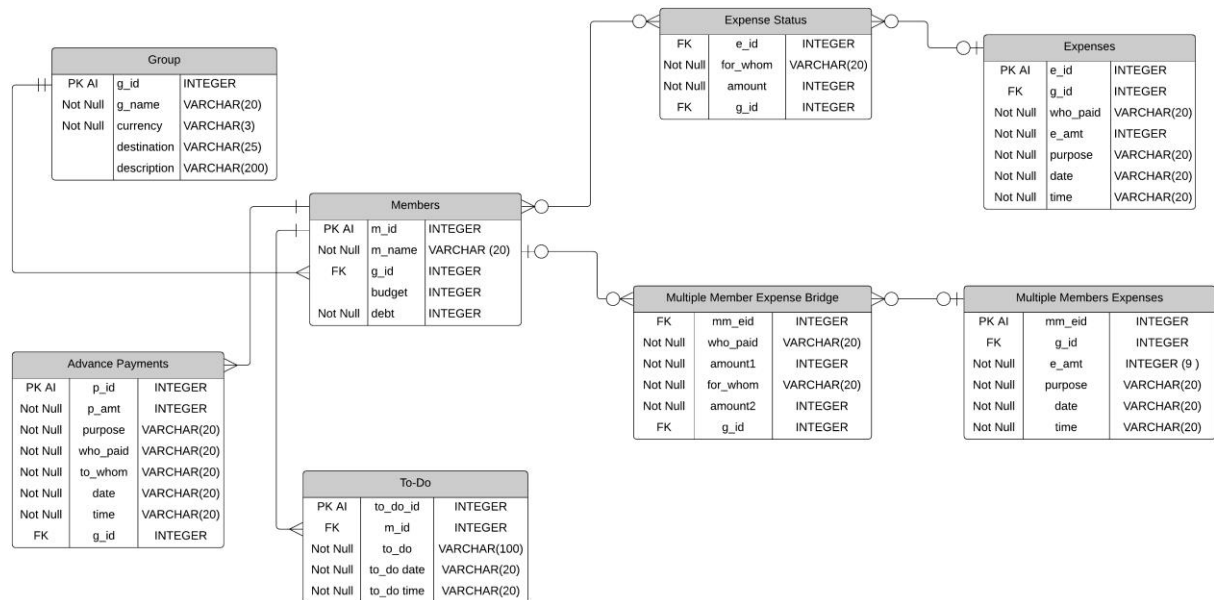
Members can also settle the debts during the trip by selecting the 'Add Payments' option. They will get a reminder about the budget if they are spending more.

Users can quickly add stuff they wish to buy during the trip. This can be quite helpful as people won't need to open another application for doing the same.

The 'first time tutorial' will help new users for accessing it and its various features.

6. Conceptual Models:

6.1. ER Diagram:



6.2. Data Dictionary

6.2.1 Table: Group

Attribute	Data Type	Constraints
g_id	integer	PK Auto Increment
g_name	varchar(20)	Not Null
currency	varchar(3)	Not Null
destination	varchar(25)	
description	varchar(200)	

6.2.2 Table: Members

Attribute	Data Type	Constraints
m_id	integer	PK Auto Increment
m_name	varchar(20)	Not Null
g_id	integer	FK
budget	integer	
Debt	integer	Not Null

6.2.3 Table: To-Do

Attribute	Data Type	Constraints
to_do_id	integer	PK Auto Increment
m_id	integer	FK
to_do	varchar(100)	Not Null
to_do_date	varchar(20)	Not Null
to_do_time	varchar(20)	Not Null

6.2.4 Table: Advance Payments

Attribute	Data Type	Constraints
p_id	integer	PK Auto Increment
p_amt	integer	Not Null
purpose	varchar(20)	Not Null
who_paid	varchar(20)	Not Null
to_whom	varchar(20)	Not Null
date	varchar(20)	Not Null
time	varchar(20)	Not Null
g_id	integer	FK

6.2.5 Table: Expenses

Attribute	Data Type	Constraints
e_id	integer	PK Auto Increment
g_id	integer	FK
who_paid	varchar(20)	Not Null
e_amt	integer	Not Null
purpose	varchar(20)	Not Null
date	varchar(20)	Not Null
time	varchar(20)	Not Null

6.2.6 Table: Expense Status

Attribute	Data Type	Constraints
e_id	integer	FK
for_whom	varchar(20)	Not Null
amount	integer	Not Null
g_id	Integer	FK

6.2.7 Table: Multiple Members Expenses

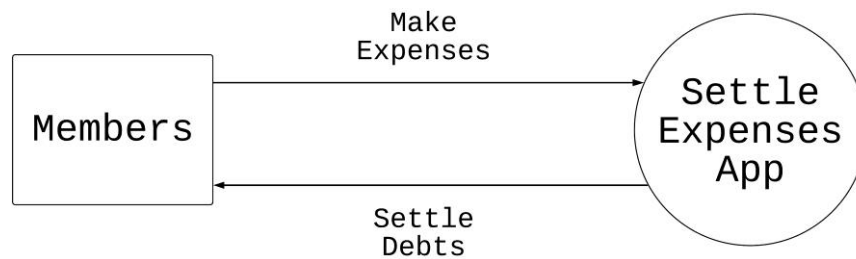
Attribute	Data Type	Constraints
mm_eid	integer	PK Auto Increment
g_id	integer	FK
e_amt	integer	Not Null
purpose	varchar(20)	Not Null
date	varchar(20)	Not Null
time	varchar(20)	Not Null

6.2.8 Table: Multiple Member Expense Bridge

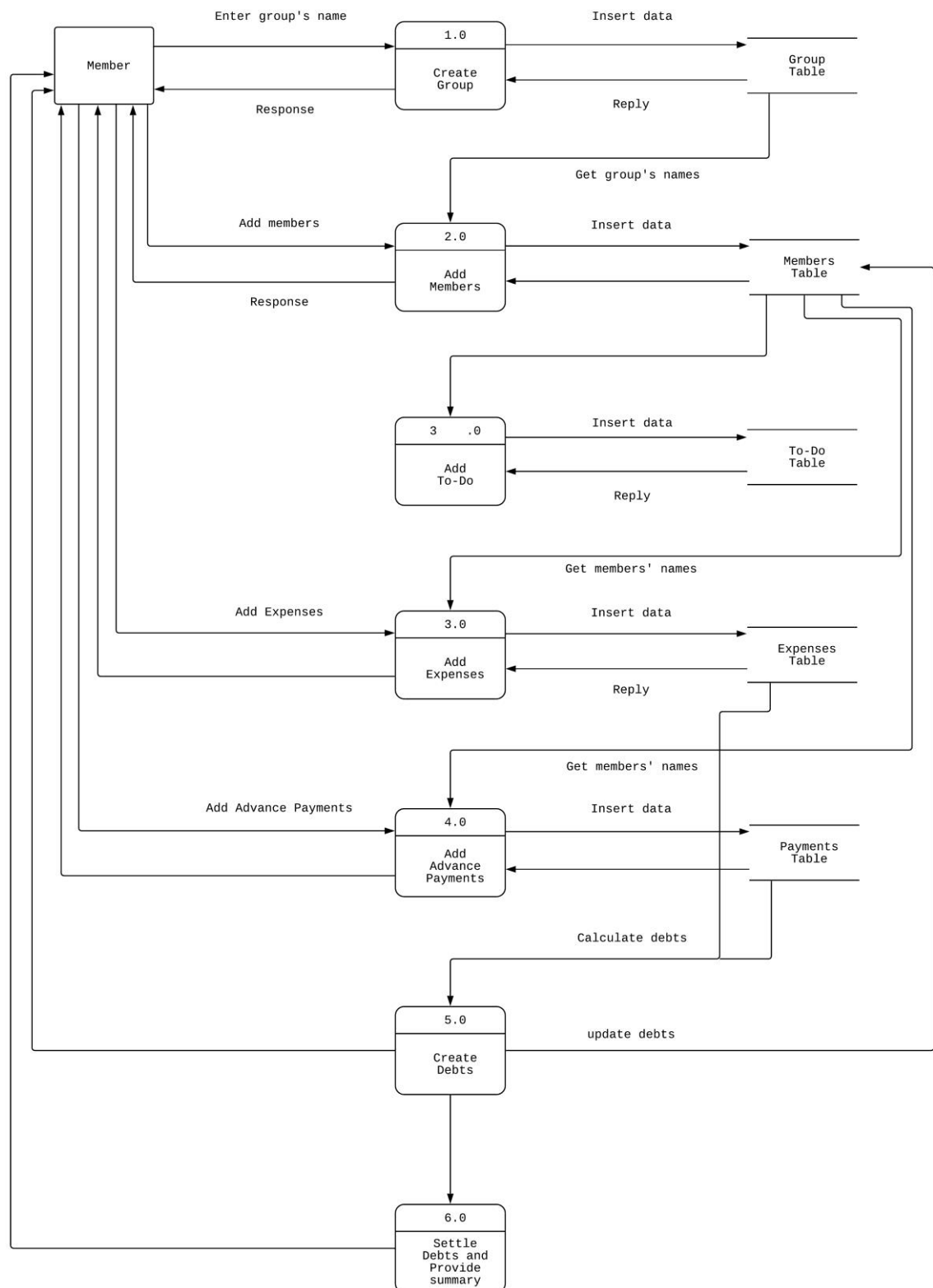
Attribute	Data Type	Constraints
mm_eid	integer	FK
who_paid	varchar(20)	Not Null
amount1	integer	Not Null
for_whom	varchar(20)	Not Null
amount2	integer	Not Null
g_id	integer	FK

6.3. Context Diagram

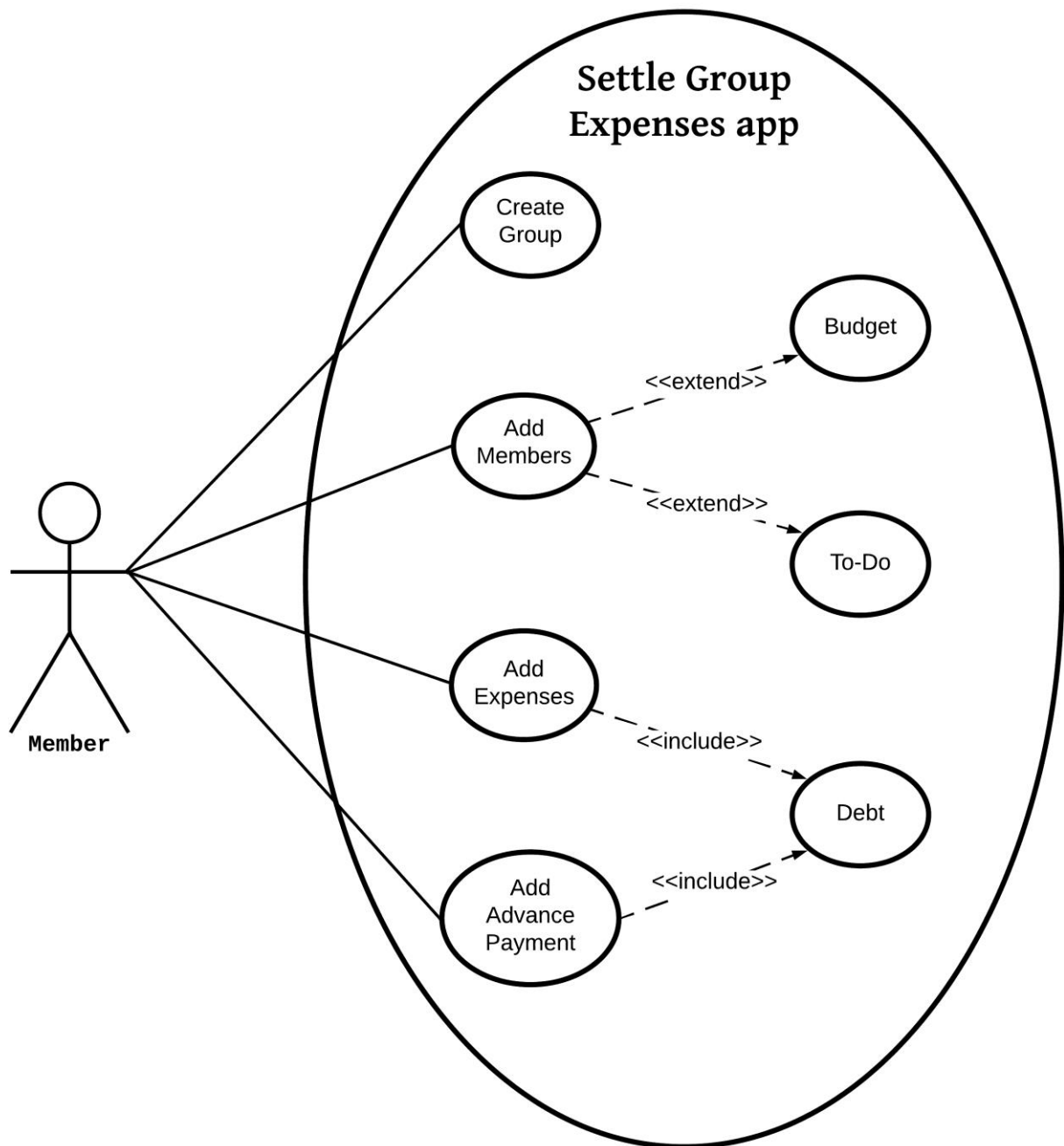
6.3.1 Level 0



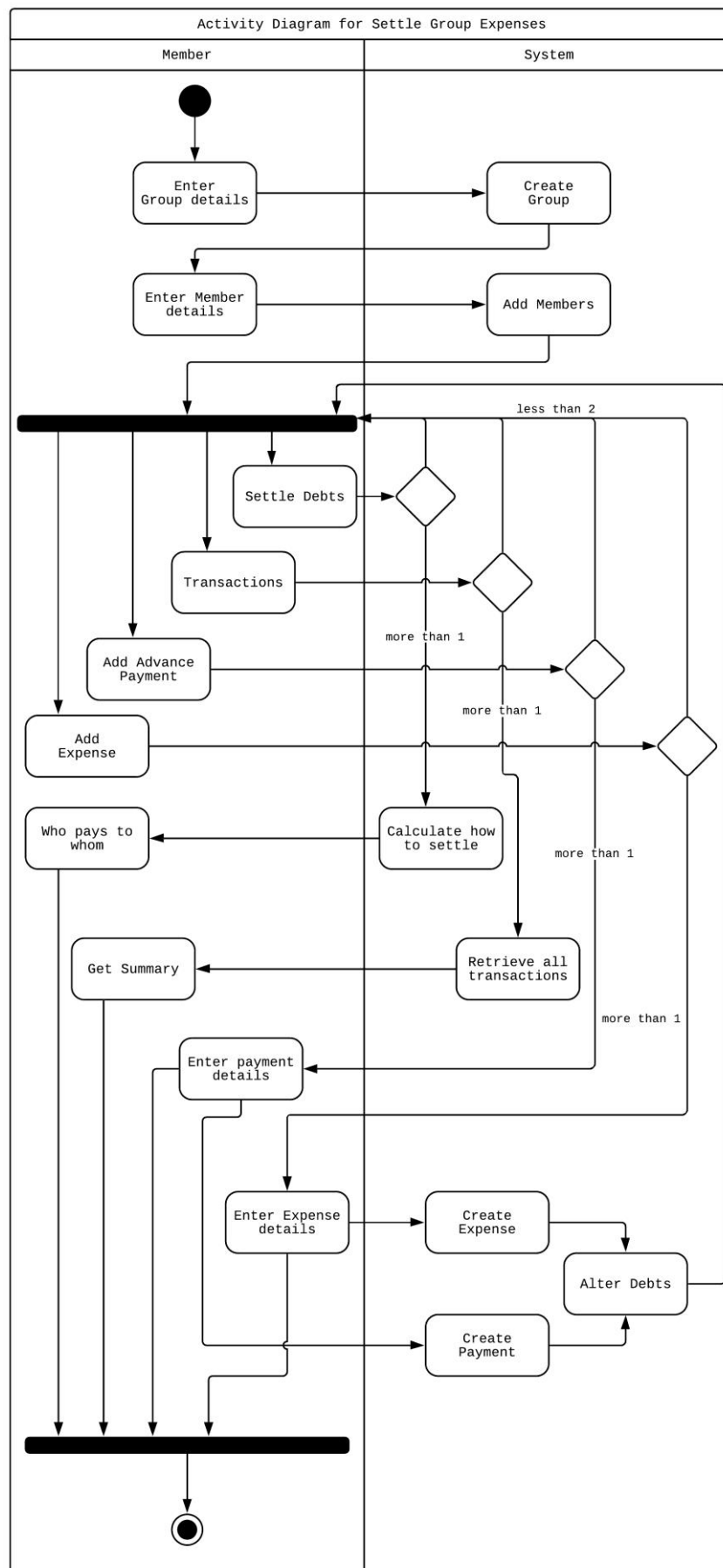
6.3.2 Dataflow Diagram



6.4. Use Case Diagram



6.5. Activity diagram



Chapter 4 System Design

This chapter shows the UI of the application, algorithms behind the working of the application and their related flowcharts.

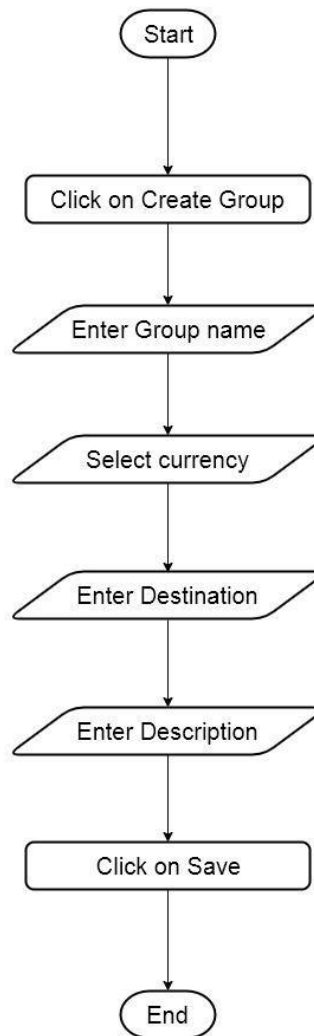
1. Create Group:

Settle Group Expenses	← Settle Group Expenses Save
Karnala (demo)	Group name <input type="text"/>
Create Group	Currency <input type="text" value="INR"/>
	Destination <input type="text"/>
	Description <input type="text"/>

1.1 Algorithm:

- Step 1: Click on Create group.
- Step 2: Enter group's name.
- Step 3: Select currency.
- Step 4: Enter destination.
- Step 5: Enter description.
- Step 6: Click on save.

1.2 Flowchart:



1.3 Validations:

- User must Enter Group name.
- User must select the currency.

2. Add Members:

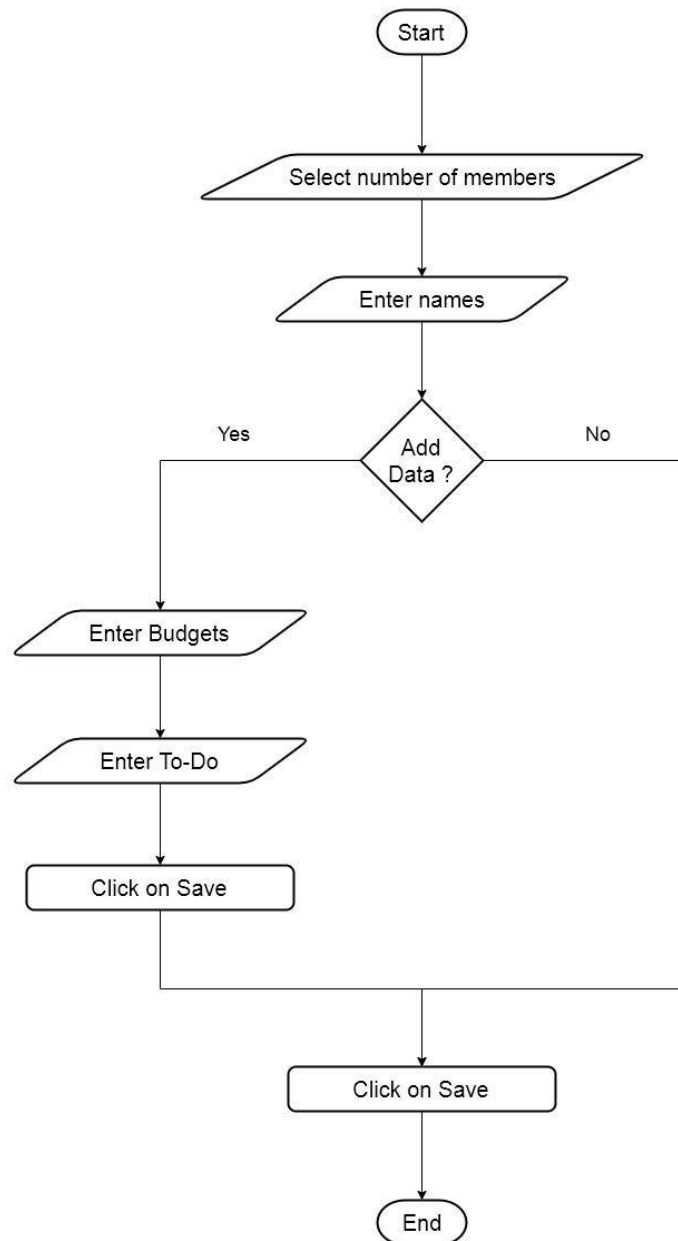
← Group Name	Save
Number of members <input type="text" value="3"/>	
Add members	
<input type="text"/>	⊕
<input type="text"/>	⊕
<input type="text"/>	⊕

← Member_name	Save
Add Budget <input type="text"/>	
Add To-Do <input type="text"/>	

2.1 Algorithm:

- Select number of members.
- Step 2: Enter names.
- Step 3: If you want to add more info
 - (a) Click on (+) button.
 - (b) Add Budget.
 - (c) Add To-Do.
 - (d) Click on save.
- Else
 - (a) Click on Save.

2.2 Flowchart:



2.3 Validations:

- Number of members should not be less than 2.
- Member names should not be empty.

3. Add Expense:

The image displays two screenshots of a mobile application interface for adding an expense.

Left Screenshot: The screen is titled "New Expense" with a "Next" button. A dropdown menu is open for "Member_1 paid", showing options: "Member_2", "Member_3", "Member_4", and "Multiple Members". The amount "70 \$" is entered. A numeric keypad is visible at the bottom.

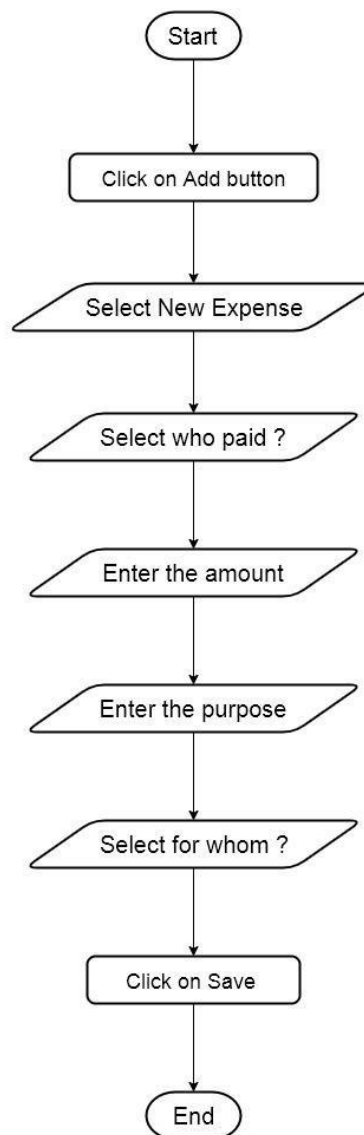
Right Screenshot: The screen is titled "New Expense" with a "Save" button. It shows the following fields:

- Purpose:** A text input field.
- Who paid ?** A list showing "Member_1" with "30 \$" and "Member_2" with "25 \$".
- For whom ?** A list showing "Member_1" with a checked checkbox, "Member_2" with an unchecked checkbox, and "Member_3" with a checked checkbox.

3.1 Algorithm:

- Step 1: Click on Add button.
- Step 2: Select new expense.
- Step 3: Select who paid?
- Step 4: Enter amount.
- Step 5: Enter purpose.
- Step 6: Select for whom?
- Step 7: Click on save.

3.2 Flowchart:



3.3 Validations:

- All fields are required to be filled.

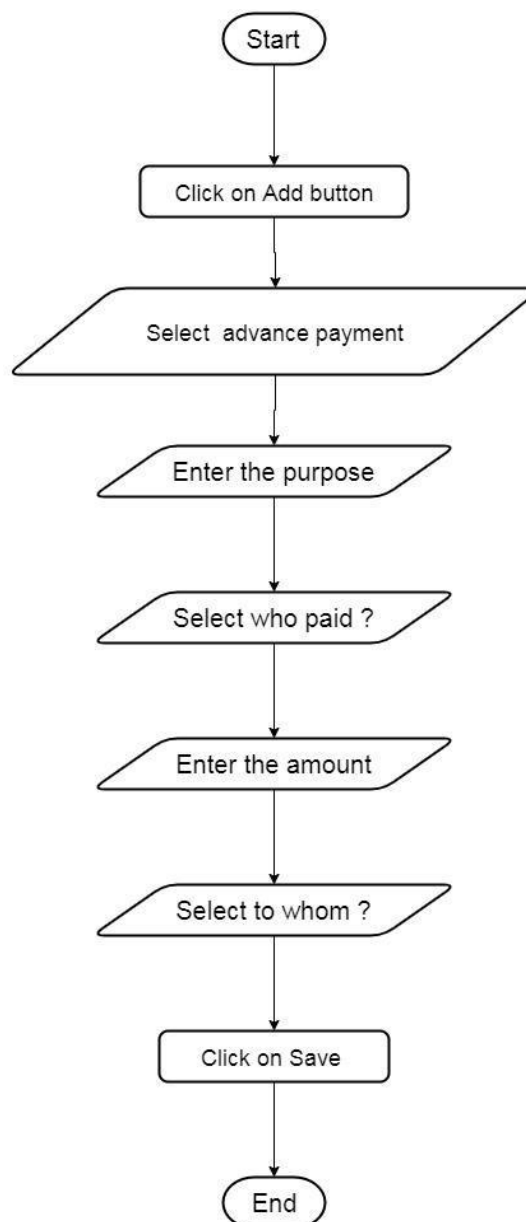
4. Add Advance Payment:

← Advance Payment		Save
Purpose	<input type="text"/>	
Who paid ?		
Member_1 ▼	10 \$	
To whom ?		
Member_2	<input checked="" type="checkbox"/>	
Member_3	<input type="checkbox"/>	

4.1 Algorithm:

- Step 1: Click on Add button.
- Step 2: Select Add Advance Payment.
- Step 3: Enter purpose.
- Step 4: Select who paid?
- Step 5: Enter the amount.
- Step 6: Select to whom?
- Step 7: Click on save.

4.2 Flowchart:



4.3 Validations:

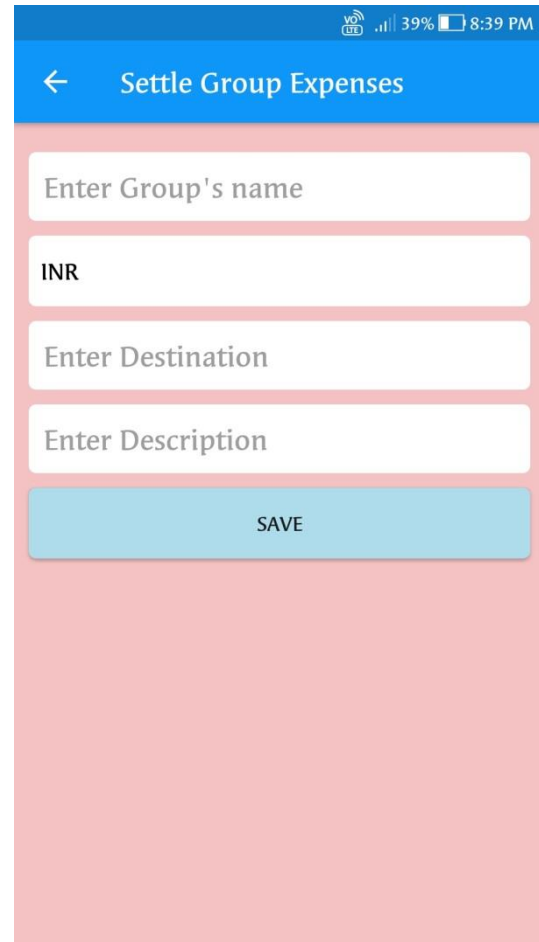
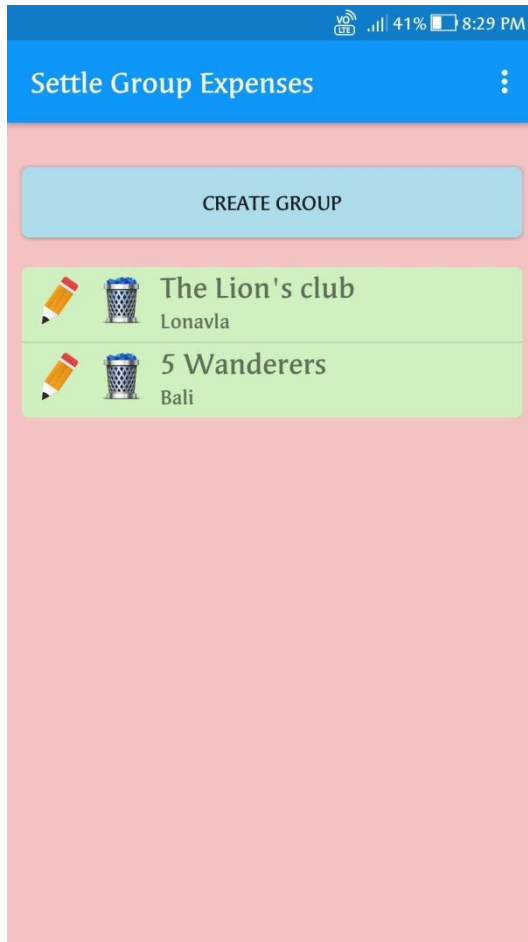
- All fields are required to be filled.

Chapter 5 Implementation and Testing

1. Screenshots and their description:

1.1 Create Group:

1.1.1 Image:



1.1.2 Details:

- Enter the details of the group like Name, currency, destination and description.
- You can edit the group's name and destination.

1.2 Add Members:

1.2.1 Image:

The screenshot shows a mobile application interface for 'The Lion's club'. At the top, there is a blue header bar with the title 'The Lion's club'. Below the header, there are two input fields: 'Enter members's name' and 'Enter budget (optional)'. A light blue button labeled 'ADD MEMBER' is positioned below the input fields. Underneath the button is a list of five members, each with a pencil icon, a trash can icon, and a name: Raj, Vivek, Smith, Robin, and Sandy. The list is set against a light green background. At the bottom right of the screen, there is a green circular button with a white plus sign.

1.2.2 Details:

- Enter the names of the members in the group and their budgets.
- You can edit the member's name and delete a member if he is not involved with any transaction.
- If a member is spending more as per his budget then he will be reminded about it during adding an expense.

1.3 Add To-Do:

1.3.1 Image:



1.3.2 Details:

- Members can see their balance budget and 'to-do' which they added earlier.
- Also they can add new to-do anytime they want to.

1.4 Add Expense:

1.4.1 Image:

The screenshot shows a mobile application interface. At the top, there is a status bar with icons for signal, 39% battery, and the time 8:36 PM. Below this is a blue header bar with the text "The Lion's club". The main content area has a light pink background. It starts with the text "Who Paid" in a dark font. Below this is a white text input field containing the name "Vivek". Further down is a white currency input field showing "0" followed by a vertical red line and the text "INR". At the bottom of the screen is a light blue bar with the word "NEXT" in white capital letters.

1.4.2 Details:

- To add expense select who paid for it. Whether it was paid by a single member or multiple members contributed in it
- Enter all details for the expense, Select the split type among members.

1.5 Add Advance Payment:

1.5.1 Image:

The screenshot shows a mobile application interface for 'The Lion's club'. At the top, there is a blue header bar with the text 'The Lion's club'. Below the header, there is a form with a pink background. The form contains the following elements:

- A text input field labeled 'Enter Purpose' with a calendar icon on the right.
- A text input field showing '0' followed by 'INR'.
- A table with two columns: 'Who paid' and 'To whom'. Both columns have a dropdown menu with the text 'Select Member'.
- A blue button labeled 'ADD'.

1.5.2 Details:

- If someone pays back the debts then it should be added here.
- Enter all the details like who paid to whom and for what reason etc.

1.6 Transactions:

1.6.1 Image:

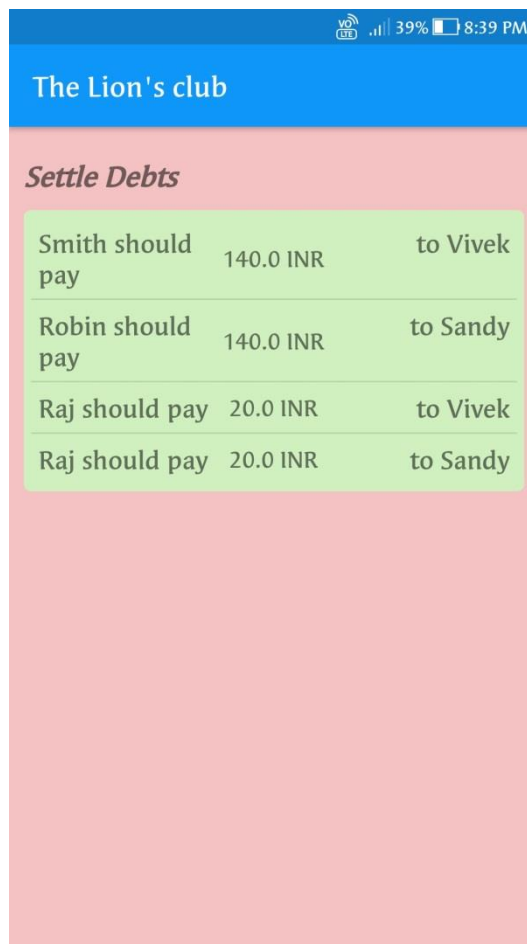


1.6.2 Details:

- All the transactions that were made will be shown here.
- By clicking on any of the transactions you can get entire summary of it.

1.7 Settle Debts:

1.7.1 Image:



The image is a screenshot of a mobile application interface. At the top, there is a status bar with icons for signal, battery (39%), and time (8:39 PM). Below this is a blue header bar with the text 'The Lion's club'. The main content area has a light pink background. A section titled 'Settle Debts' in italics contains a table with a light green background. The table lists four debt entries.

<i>Settle Debts</i>		
Smith should pay	140.0 INR	to Vivek
Robin should pay	140.0 INR	to Sandy
Raj should pay	20.0 INR	to Vivek
Raj should pay	20.0 INR	to Sandy

1.7.2 Details:

- It will tell us how to settle the debts among the members.

2. Code:

The following code is the one which I have used for settling the debts.

```
// ----- Directly settling among those members who have to pay or receive the same amount.
for (int i=0; i< UpdatedDebtList.size(); i++){
    for (int j=0; j< UpdatedDebtList.size(); j++){
        if (UpdatedDebtList.get(i)!=0 && UpdatedDebtList.get(j)!=0 && UpdatedDebtList.get(i) +
UpdatedDebtList.get(j) == 0){
            if (UpdatedDebtList.get(i)>0){
                WhoPaysList.add(NameList.get(i));
                ToWhomList.add(NameList.get(j));
                HowMuchList.add(UpdatedDebtList.get(i));
                UpdatedDebtList.set(i,0d);
                UpdatedDebtList.set(j,0d);
            }
            else if (UpdatedDebtList.get(i)<0){
                WhoPaysList.add(NameList.get(j));
                ToWhomList.add(NameList.get(i));
                HowMuchList.add(UpdatedDebtList.get(j));
                UpdatedDebtList.set(i,0d);
                UpdatedDebtList.set(j,0d);
            }
        }
    }
}

// ----- Removing those members whose debts have been settled
int counter1=NameList.size();
int counter2=UpdatedDebtList.size();
for (int i=0; i< counter2; i++){
    if (UpdatedDebtList.get(i) == 0){
        NameList.set(i, ""+0);
    }
}
for (int i=0; i< counter1; i++){
    if (NameList.get(i) == ""+0){
        NameList.remove(i);
        counter1--;
        i--;
    }
}
for (int i=0; i< counter2; i++){
    if (UpdatedDebtList.get(i) == 0){
        UpdatedDebtList.remove(i);
        counter2--;
        i--;
    }
}
```

```

}
// Settling the debts of the remaining members.
for (int i=0; i< UpdatedDebtList.size(); i++){
    if (UpdatedDebtList.get(i) > 0){
        PayersNameList.add(NameList.get(i));
        PayersAmountList.add(UpdatedDebtList.get(i));
    }
    else{
        ReceiversNameList.add(NameList.get(i));
        ReceiversAmountList.add(UpdatedDebtList.get(i));
    }
}
if (! NameList.isEmpty() && ! DebtList.isEmpty()){
    int counter;
    if (PayersAmountList.size() > ReceiversAmountList.size()){
        counter = PayersAmountList.size();
    }
    else if ((PayersAmountList.size() < ReceiversAmountList.size())){
        counter = ReceiversAmountList.size();
    }
    else {
        counter=PayersAmountList.size();
    }
    try{
        for (int i=0; i<= counter; i++){
            if (PayersAmountList.get(i) + ReceiversAmountList.get(i) > 0){
                WhoPaysList.add(PayersNameList.get(i));
                ToWhomList.add(ReceiversNameList.get(i));
                HowMuchList.add(- ReceiversAmountList.get(i));
                PayersNameList.add(PayersNameList.get(i));
                PayersAmountList.add(PayersAmountList.get(i) + ReceiversAmountList.get(i));
                ReceiversNameList.set(i, ""+0);
                ReceiversAmountList.set(i, 0d);
                PayersNameList.set(i, ""+0);
                PayersAmountList.set(i, 0d);
            }
            else {
                WhoPaysList.add(PayersNameList.get(i));
                ToWhomList.add(ReceiversNameList.get(i));
                HowMuchList.add(PayersAmountList.get(i));
                ReceiversNameList.add(ReceiversNameList.get(i));
                ReceiversAmountList.add(PayersAmountList.get(i) + ReceiversAmountList.get(i));\
                PayersNameList.set(i, ""+0);
                PayersAmountList.set(i, 0d);
                ReceiversNameList.set(i, ""+0);
                ReceiversAmountList.set(i, 0d);
            }
        }
        counter++;
    }
}

```

```

    }
}
catch (Exception e){
//Toast.makeText(this, ""+e, Toast.LENGTH_SHORT).show();
}
}

for (int i=0; i<HowMuchList.size(); i++){
    HowMuchList.set(i,(Double.parseDouble(df.format(HowMuchList.get(i)))));
}

```

3. Test Cases:

Project Title	Settle Group Expenses
Created By	Sumit A. Bane
Creation Date	13-04-2020

3.1 Functional:

3.1.1 Module - Create Group:

Module Name	Create Group
Test Scenario ID	TS_CG
Test Scenario Description	Verify the Create Group Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_CG_01	Enter Group's name, Select Currency, Enter Destination and Description	1. Enter Group's name. 2. Select Currency. 3. Enter Destination. 4. Enter Description. 5. Click on 'Save'.	Test Data	1. The Lion's Club. 2. INR. 3. Lonavla. 4. A trip in winters.	User should be redirected to home activity	Successful creation of group	Successful creation of group	Pass
TC_CG_02	Select Currency, Enter Destination and Description	1. Select Currency. 2. Enter Destination. 3. Enter Description. 4. Click on 'Save'.	Test Data	1. INR. 2. Lonavla. 3. A trip in winters.	User should stay on the same page and Toast prompting to enter the data	Unsuccessful creation of group	Unsuccessful creation of group	Pass
TC_CG_03	Enter Group's name, Destination and Description	1. Enter Group's name. 2. Enter Destination. 3. Enter Description. 4. Click on 'Save'.	Test Data	1. The Lion's Club. 2. Lonavla. 3. A trip in winters.	User should be redirected to home activity	Successful creation of group with currency as INR	Successful creation of group with currency as INR	Pass
TC_CG_04	Enter Group's name, Select Currency, Enter Description	1. Enter Group's name. 2. Select Currency. 3. Enter Description. 4. Click on 'Save'.	Test Data	1. The Lion's Club. 2. INR. 4. A trip in winters.	User should be redirected to home activity	Successful creation of group with Destination as 'Destination not added'	Successful creation of group with Destination as 'Destination not added'	Pass
TC_CG_05	Enter Group's name, Select Currency, Enter Destination	1. Enter Group's name. 2. Select Currency. 3. Enter Destination. 4. Click on 'Save'.	Test Data	1. The Lion's Club. 2. INR. 3. Lonavla.	User should be redirected to home activity	Successful creation of group	Successful creation of group	Pass

3.1.2 Module - Home Activity:

Module Name	Home Activity
Test Scenario ID	TS_EG
Test Scenario Description	Verify the Edit Group Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_EG_01	Enter Group's Name and Destination	1. Enter Group's name. 2. Enter Destination. 3. Click on 'Save'.	Test Data	1. The Lions. 2. Manali.	Alert Dialog box should disappear	Successful edit of group	Successful edit of group	Pass
TC_EG_02	Enter only Destination	1. Enter Destination. 2. Click on 'Save'.	Test Data	1. Manali.	Toast message - 'Enter Group's name'. Alert Dialog box should not disappear	Unsuccessful edit of group	Unsuccessful edit of group	Pass
TC_EG_03	Enter only Group's Name	1. Enter Group's name. 2. Click on 'Save'.	Test Data	1. The Lions.	Alert Dialog box should disappear. Destination set to 'Destination not added'	Successful edit of group	Successful edit of group	Pass
TC_EG_04	Don't enter any data	1. Click on 'Save'.			Toast message - 'Enter Group's name'. Alert Dialog box should not disappear	Unsuccessful edit of group	Unsuccessful edit of group	Pass

3.1.3 Module - Add Member:

Module Name	Add Members
Test Scenario ID	TS_AM
Test Scenario Description	Verify the Add Member Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_AM_01	Enter Member's Name and Budget	1. Enter Member's name. 2. Enter Budget. 3. Click on 'Add Member'.	Test Data	1. Raj. 2. 2500.	The Member should be added in the list.	Member added	Member added	Pass
TC_AM_02	Enter only Member's Name	1. Enter Member's name. 2. Click on 'Add Member'.	Test Data	1. Manish.	The Member should be added in the list.	Member added	Member added	Pass
TC_AM_03	Enter only Budget	1. Enter Budget. 2. Click on 'Save'.	Test Data	1. 2000.	Toast message - 'Enter Member's name'.	Member not added	Member not added	Pass
TC_AM_04	Enter Member's name with same name as entered already	1. Enter Member's name. 2. Click on 'Save'.	Member name added already and Test Data	1. Manish.	Toast message - 'Member with same name already exist'.	Member not added	Member not added	Pass
TC_AM_05	Enter Member's name after 15 members are added	1. Enter Member's name. 2. Click on 'Save'.	Test Data	1. Dinesh.	Toast message - 'Maximum 15 members are allowed'.	Member not added	Member not added	Pass

Module Name	Add Members
Test Scenario ID	TS_EM
Test Scenario Description	Verify the Edit Member Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_EM_01	Enter a different name	1. Enter Member's name. 2. Click on 'Save'.	Test Data	1. Dinesh.	Dialog Box should disappear without any message	Member's name edited.	Member's name edited.	Pass
TC_EM_02	Enter the same name	1. Enter Member's name. 2. Click on 'Save'.	Test Data	1. Dinesh.	Toast message - 'Same name as previous'. Dialog Box should not disappear.	Member's name not edited.	Member's name not edited.	Pass
TC_EM_03	Enter a different name which is present in the Group	1. Enter Member's name. 2. Click on 'Save'.	Test Data ie a name which is already added	1. Manish.	Toast message - 'Member with same name already exist'. Dialog Box should not disappear.	Member's name not edited.	Member's name not edited.	Pass
TC_EM_04	Don't enter any name	1. Click on 'Save'.			Toast message - 'Enter a name'. Dialog Box should not disappear.	Member's name not edited.	Member's name not edited.	Pass

Module Name	Add Members
Test Scenario ID	TS_DM
Test Scenario Description	Verify the Delete Member Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Post Conditions	Expected Result	Actual Result	Status
TC_DM_01	Deleting a member who has debt	1. Click on 'Delete' button.	Member must have some debt	Snackbar message - 'Member cannot be deleted as he has debts to settle'.	Member not deleted.	Member not deleted.	Pass
TC_DM_02	Deleting a member who doesn't have any debt	1. Click on 'Delete' button.	Member must not have any debt	Alert Dialog asking - 'Do you want to delete this member ?'	Member deleted.	Member deleted.	Pass

Module Name	Add Members
Test Scenario ID	TS_FB
Test Scenario Description	Verify the Floating Button Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Post Conditions	Expected Result	Actual Result	Status
TC_FB_01	Move to 'Add Expense' activity	1. Click on Floating Action Button. 2. Click on 'Add Expense' menu	More than 1 member added in the group	Add Expense' activity must be opened	User redirected to 'Add Expense' activity	User redirected to 'Add Expense' activity	Pass
TC_FB_02	Move to 'Add Expense' activity	1. Click on Floating Action Button. 2. Click on 'Add Expense' menu	Less than 2 members added in the group	Toast message - 'At least 2 members needed'.	User not redirected to 'Add Expense' activity	User not redirected to 'Add Expense' activity	Pass
TC_FB_03	Move to 'Add Advance Payment' activity	1. Click on Floating Action Button. 2. Click on 'Add Advance Payment' menu	More than 1 member added in the group	'Advance Payment' activity must be opened	User redirected to 'Advance Payment' activity	User redirected to 'Advance Payment' activity	Pass
TC_FB_04	Move to 'Add Advance Payment' activity	1. Click on Floating Action Button. 2. Click on 'Add Advance Payment' menu	Less than 2 members added in the group	Toast message - 'At least 2 members needed'.	User not redirected to 'Advance Payment' activity	User not redirected to 'Advance Payment' activity	Pass
TC_FB_05	Move to 'Transactions' activity	1. Click on Floating Action Button. 2. Click on 'Transactions' menu	More than 1 member added in the group	'Add Payment' activity must be opened	User redirected to 'Transactions' activity	User redirected to 'Transactions' activity	Pass
TC_FB_06	Move to 'Transactions' activity	1. Click on Floating Action Button. 2. Click on 'Transactions' menu	Less than 2 members added in the group	Toast message - 'At least 2 members needed'.	User not redirected to 'Transactions' activity	User not redirected to 'Transactions' activity	Pass
TC_FB_07	Move to 'Settle Debts' activity	1. Click on Floating Action Button. 2. Click on 'Settle Debts' menu	More than 1 member added in the group	'Transactions' activity must be opened	User redirected to 'Settle Debts' activity	User redirected to 'Settle Debts' activity	Pass
TC_FB_08	Move to 'Settle Debts' activity	1. Click on Floating Action Button. 2. Click on 'Settle Debts' menu	Less than 2 members added in the group	Toast message - 'At least 2 members needed'.	User not redirected to 'Settle Debts' activity	User not redirected to 'Settle Debts' activity	Pass

3.1.4 Module - Member Details:

Module Name	Members Details
Test Scenario ID	TS_AT
Test Scenario Description	Verify the Add To-Do Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_AT_01	Enter a To-Do and select date and time	1. Enter To-Do. 2. Select date and time. 3. Click on 'Add'.	Test Data	1. Buy Strawberries. 2. 16-4-2020. 3. 5:00 pm.	To-Do should be added in the list.	To-Do added successfully.	To-Do added successfully.	Pass
TC_AT_02	Enter only To-Do	1. Enter To-Do. 2. Click on 'Add'.	Test Data	1. Buy Strawberries.	Toast message - 'Select date'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_03	Select only date and time	1. Select date and time. 2. Click on 'Add'.	Test Data	1. 16-4-2020. 2. 5:00 pm.	Toast message - 'Enter To-Do'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_04	Select only date.	1. Select date. 2. Click on 'Add'.	Test Data	1. 16-4-2020.	Toast message - 'Enter To-Do'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_05	Select only time.	1. Select time. 2. Click on 'Add'.	Test Data	1. 5:00 pm.	Toast message - 'Enter To-Do'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_06	Enter a To-Do and select date	1. Select date. 2. Click on 'Add'.	Test Data	1. Buy Strawberries. 2. 16-4-2020.	Toast message - 'Select time'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_07	Enter a To-Do and select time	1. Select time. 2. Click on 'Add'.	Test Data	1. Buy Strawberries. 2. 5:00 pm.	Toast message - 'Select date'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass
TC_AT_08	Don't enter To-Do and don't select date and time	1. Click on 'Add'.			Toast message - 'Enter To-Do'. To-Do should not be added in the list.	To-Do not added.	To-Do not added.	Pass

3.1.5 Module - Add Expense:

Module Name	Add Expense
Test Scenario ID	TS_SET
Test Scenario Description	Verify the Select Expense Type Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_SET_01	Select a member and enter an amount greater than '0'	1. Select member. 2. Enter amount. 3. Click on 'Next'.	Test Data	1. Manish. 2. 500.	User should be redirected to 'Expense Details' activity	'Who paid' and 'amount' added	'Who paid' and 'amount' added	Pass
TC_SET_02	Select a member and enter an amount equal to '0'	1. Select member. 2. Enter amount. 3. Click on 'Next'.	Test Data	1. Manish. 2. 0.	Toast message - 'Amount should be greater than 0'. User should stay on same activity.	'Who paid' and 'amount' not added	'Who paid' and 'amount' not added	Pass
TC_SET_03	Select a member and don't enter amount.	1. Select member. 2. Click on 'Next'.	Test Data	1. Manish.	Toast message - 'Enter a valid amount'. User should stay on same activity.	'Who paid' and 'amount' not added	'Who paid' and 'amount' not added	Pass
TC_SET_04	Select 'MULTIPLE MEMBERS'.	1. Select 'MULTIPLE MEMBERS'.			User should be redirected to 'Multiple Members Expense' activity	'Multiple Members Expense' activity open	'Multiple Members Expense' activity open	Pass

3.1.6 Module - Expense Details:

Module Name	Expense Details
Test Scenario ID	TS_AE
Test Scenario Description	Verify the Add Expense Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_AE_01	Enter purpose and select date and time	1. Enter purpose. 2. Select date and time 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 14-4-2020. 3. 9:00 am.	All the fields should disappear and 'For whom' layout must be visible	Purpose, date and time of the expense added	Purpose, date and time of the expense added	Pass
TC_AE_02	Enter only purpose.	1. Enter purpose. 2. Click on 'Add'.	Test Data	1. Bus Tickets.	All the fields should disappear and 'For whom' layout must be visible	Purpose, date and time of the expense added with current date and time	Purpose, date and time of the expense added with current date and time	Pass
TC_AE_03	Select only date	1. Select date. 2. Click on 'Add'.	Test Data	1. 14-4-2020.	Toast message - 'Enter purpose'.	Purpose not added	Purpose not added	Pass
TC_AE_04	Select only time	1. Select time. 2. Click on 'Add'.	Test Data	1. 9:00 am.	Toast message - 'Enter purpose'.	Purpose not added	Purpose not added	Pass
TC_AE_05	Enter purpose and select date	1. Enter purpose. 2. Select date. 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 14-4-2020.	All the fields should disappear and 'For whom' layout must be visible	Purpose and date added and system's current time added	Purpose and date added and system's current time added	Pass
TC_AE_06	Enter purpose and select time	1. Enter purpose. 2. Select time. 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 9:00 am.	All the fields should disappear and 'For whom' layout must be visible	Purpose and time added and system's current date added	Purpose and time added and system's current date added	Pass
TC_AE_07	Don't enter purpose	1. Click on 'Add'.			Toast message - 'Enter purpose'.	Purpose not added	Purpose not added	Pass

Module Name	Expense Details
Test Scenario ID	TS_FW
Test Scenario Description	Verify the For Whom Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_FW_01	Select those members for whom someone pays. The amount is same for all.	1. Select Members 2. Click on 'Save'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added successfully.	Expense added successfully.	Pass
TC_FW_02	Not selecting any member.	1. Click on 'Save'.	1. 'Who paid' and 'purpose' added already.		Toast message - 'Select at least one member'	Expense not added.	Expense not added.	Pass
TC_FW_03	Select those members for whom someone pays. The amount is not same for all.	1. Click on 'Split Amount'. 2. Click on 'Edit Amount' button. 3. Enter the amount. Click on 'Save'.	1. 'Who paid' and 'expense' added already. 2. Total amount must be equal to expense amount. 3. Test data.	1. Member names and respective amounts.	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added successfully.	Expense added successfully.	Pass
TC_FW_04	Select those members for whom someone pays. The amount is not same for all.	1. Click on 'Split Amount'. 2. Click on 'Edit Amount' button. 3. Enter the amount. Click on 'Save'.	1. 'Who paid' and 'expense' added already. 2. Total amount must not be equal to expense amount. 3. Test data.	1. Member names and respective amounts.	Toast message - 'Total amount differs with expense amount'.	Expense not added.	Expense not added.	Pass
TC_FW_05	Checking some checkboxes, then clicking on 'Split Amount' and then again clicking on 'Equal Amount'.	1. Check some checkboxes. 2. Click on 'Split Amount'. 3. Click on 'Equal Amount'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names	The selections must be set to default ie. 'Unchecked'.	No checkbox selected.	No checkbox selected.	Pass
TC_FW_06	Entering amounts for members, then clicking on 'Equal Amount' and then again clicking on 'Split Amount'.	1. Enter amounts for members. 2. Click on 'Equal Amount'. 3. Click on 'Split Amount'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	The amounts must be set to default ie. '0.0'.	Amounts set to '0.0'.	Amounts set to '0.0'.	Pass
TC_FW_07	Checking some checkboxes and scrolling the list several times.	1. Check some checkboxes. 2. Scroll the list up and down multiple times. 3. Click on 'Save'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added with only selected members and the amount split equally.	Expense added with only selected members and the amount split equally.	Pass
TC_FW_08	Entering amounts for members and scrolling the list several times.	1. Enter amounts for members. 2. Scroll the list up and down multiple times. 3. Click on 'Save'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added with only selected members and their respective amounts.	Expense added with only selected members and their respective amounts.	Pass
TC_FW_09	Checking some checkboxes and pressing 'Back button'.	1. Check some checkboxes. 2. Press the 'Back button'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names	Who paid' and 'purpose' deleted.	'For Whom' layout disappears and 'Add Expense' layout gets visible.	'For Whom' layout disappears and 'Add Expense' layout gets visible.	Pass
TC_FW_10	Entering amounts for members and pressing 'Back button'.	1. Enter amounts for members. 2. Press the 'Back button'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	Who paid' and 'purpose' deleted.	'For Whom' layout disappears and 'Add Expense' layout gets visible.	'For Whom' layout disappears and 'Add Expense' layout gets visible.	Pass

3.1.7 Module - Multiple Members Expense:

Module Name	Multiple Members Expense
Test Scenario ID	TS_MMAE
Test Scenario Description	Verify the Multiple Members Add Expense Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_MMAE_01	Enter purpose and amount, select date and time	1. Enter purpose. 2. Enter amount. 3. Select date and time 4. Click on 'Add'	Test Data	1. Bus Tickets. 2. 200. 3. 14-4-2020. 4. 9:00 am.	All the fields should disappear and 'Who Paid' layout must be visible	Purpose, amount, date and time of the expense added.	Purpose, amount, date and time of the expense added.	Pass
TC_MMAE_02	Enter only purpose.	1. Enter purpose. 2. Click on 'Add'.	Test Data	1. Bus Tickets.	Toast message - 'Enter a valid amount'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_03	Select only date.	1. Select date. 2. Click on 'Add'.	Test Data	1. 14-4-2020.	Toast message - 'Enter purpose'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_04	Select only time.	1. Select time. 2. Click on 'Add'.	Test Data	1. 9:00 am.	Toast message - 'Enter purpose'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_05	Enter only amount.	1. Enter amount. 2. Click on 'Add'.	Test Data	1. 200.	Toast message - 'Enter purpose'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_06	Enter purpose and select date.	1. Enter purpose. 2. Select date. 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 14-4-2020.	Toast message - 'Enter a valid amount'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_07	Enter purpose and select time.	1. Enter purpose. 2. Select time. 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 9:00 am.	Toast message - 'Enter a valid amount'.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_08	Enter purpose and amount.	1. Enter purpose. 2. Enter amount. 3. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 200.	All the fields should disappear and 'Who Paid' layout must be visible	Purpose and amount of the expense added with current date and time.	Purpose and amount of the expense added with current date and time.	Pass
TC_MMAE_09	Enter purpose and amount and select date.	1. Enter purpose. 2. Enter amount. 3. Select date. 4. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 200. 3. 14-4-2020.	All the fields should disappear and 'Who Paid' layout must be visible	Purpose, amount and date added and system's current time added.	Purpose, amount and date added and system's current time added.	Pass
TC_MMAE_10	Enter purpose and amount and select time.	1. Enter purpose. 2. Enter amount. 3. Select time. 4. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 200. 3. 9:00 am.	All the fields should disappear and 'Who Paid' layout must be visible	Purpose, amount and time added and system's current date added.	Purpose, amount and time added and system's current date added.	Pass
TC_MMAE_11	Enter purpose and amount equal to '0'.	1. Enter purpose. 2. Click on 'Add'.	Test Data	1. Bus Tickets. 2. 0.	Toast message - 'Amount should be greater than '0''.	Purpose and amount not added.	Purpose and amount not added.	Pass
TC_MMAE_12	Don't enter anything.	1. Click on 'Add'.			Toast message - 'Enter purpose'.	Purpose and amount not added.	Purpose and amount not added.	Pass

Module Name	Multiple Members Expense
Test Scenario ID	TS_MMWP
Test Scenario Description	Verify the Multiple Members Who Paid Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_MMWP_01	Entering amounts for members and clicking on 'Add'.	1. Enter amounts for members. 2. Click on 'Add'.	1. 'Purpose' and 'amount' added already. 2. Test Data.	1. Member names and respective amounts.	'Who Paid' layout disappears and 'For Whom' layout is set visible.	'Who Paid' and their respective amounts added.	'Who Paid' and their respective amounts added.	Pass
TC_MMAE_02	Not entering amount in Alert dialog and clicking on 'Save'.	1. Click on 'Save'.	1. 'Purpose' and 'amount' added already.		Toast message - 'Enter a valid amount '.	Alert Dialog box does not disappears.	Alert Dialog box does not disappears.	Pass
TC_MMAE_03	Not entering amounts and clicking on 'Add'.	1. Click on 'Add'.	1. 'Purpose' and 'amount' added already.		Toast message - 'Toatl amount differs with expense amount'.	'Who Paid' not added.	'Who Paid' not added.	Pass
TC_MMAE_04	Entering amounts but their sum is different' for expense amount.	1. Enter amounts for members. 2. Click on 'Add'.	1. 'Purpose' and 'amount' added already. 2. Test Data.	1. Member names and respective amounts.	Toast message - 'Toatl amount differs with expense amount'.	'Who Paid' not added.	'Who Paid' not added.	Pass
TC_MMAE_05	Entering amounts and pressing 'Back Button'.	1. Enter amounts for members. 2. Click on Back Button'.	1. 'Who paid' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	'Who Paid' layout disappears and 'Multiplt Members Expense' layout is set visible.	'Who Paid' and their respective amounts not added. Expense details deleted.	'Who Paid' and their respective amounts not added. Expense details deleted.	Pass
TC_MMAE_06	Pressing 'Back Button'.	1. Click on Back Button'.	1. 'Who paid' and 'purpose' added already.		'Who Paid' layout disappears and 'Multiplt Members Expense' layout is set visible.	'Who Paid' and their respective amounts not added. Expense details deleted.	'Who Paid' and their respective amounts not added. Expense details deleted.	Pass

Module Name	Multiple Members Expense
Test Scenario ID	TS_MMFW
Test Scenario Description	Verify the Multiple Members For Whom Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_MMFW_01	Select those members for whom someone pays. The amount is same for all.	1. Select Members 2. Click on 'Save'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added successfully.	Expense added successfully.	Pass
TC_MMFW_02	Not selecting any member.	1. Click on 'Save'.	1. 'Who paid' and 'purpose' added already.		Toast message - 'Select at least one member'	Expense not added.	Expense not added.	Pass
TC_MMFW_03	Select those members for whom someone pays. The amount is not same for all.	1. Click on 'Split Amount'. 2. Click on 'Edit Amount' button. 3. Enter the amount. Click on 'Save'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Total amount must be equal to expense amount. 3. Test data.	1. Member names and respective amounts.	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added successfully.	Expense added successfully.	Pass
TC_MMFW_04	Select those members for whom someone pays. The amount is not same for all.	1. Click on 'Split Amount'. 2. Click on 'Edit Amount' button. 3. Enter the amount. Click on 'Save'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Total amount must not be equal to expense amount. 3. Test data.	1. Member names and respective amounts.	Toast message - 'Total amount differs with expense amount'.	Expense not added.	Expense not added.	Pass
TC_MMFW_05	Checking some checkboxes, then clicking on 'Split Amount' and then again clicking on 'Equal Amount'.	1. Check some checkboxes. 2. Click on 'Split Amount'. 3. Click on 'Equal Amount'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names	The selections must be set to default ie. 'Unchecked'.	No checkbox selected.	No checkbox selected.	Pass
TC_MMFW_06	Entering amounts for members, then clicking on 'Equal Amount' and then again clicking on 'Split Amount'.	1. Enter amounts for members. 2. Click on 'Equal Amount'. 3. Click on 'Split Amount'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	The amounts must be set to default ie. '0.0'.	Amounts set to '0.0'.	Amounts set to '0.0'.	Pass
TC_MMFW_07	Checking some checkboxes and scrolling the list several times.	1. Check some checkboxes. 2. Scroll the list up and down multiple times. 3. Click on 'Save'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added with only selected members and the amount split equally.	Expense added with only selected members and the amount split equally.	Pass
TC_MMFW_08	Entering amounts for members and scrolling the list several times.	1. Enter amounts for members. 2. Scroll the list up and down multiple times. 3. Click on 'Save'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	Toast message - 'Expense added'. User redirected to 'Add Expense' activity.	Expense added with only selected members and their respective amounts.	Expense added with only selected members and their respective amounts.	Pass
TC_MMFW_09	Checking some checkboxes and pressing 'Back button'.	1. Check some checkboxes. 2. Press the 'Back button'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names	'For Whom' layout disappears and 'Who Paid' layout is set visible.	'For Whom' and their respective amounts not added. 'Who Paid' details deleted.	'For Whom' and their respective amounts not added. 'Who Paid' details deleted.	Pass
TC_MMFW_10	Entering amounts for members and pressing 'Back button'.	1. Enter amounts for members. 2. Press the 'Back button'.	1. 'Who paid', 'amount' and 'purpose' added already. 2. Test Data.	1. Member names and respective amounts.	'For Whom' layout disappears and 'Who Paid' layout is set visible.	'For Whom' and their respective amounts not added. 'Who Paid' details deleted.	'For Whom' and their respective amounts not added. 'Who Paid' details deleted.	Pass

3.1.8 Module - Advance Payment:

Module Name	Advance Payment
Test Scenario ID	TS_AP
Test Scenario Description	Verify the Advance Payment Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_AP_01	Enter purpose and amount, select 'who paid', 'to whom', date and time	1. Enter purpose. 2. Select date and time. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 8:45 pm. 4. 50. 5. Manish. 6. Raj.	Toast message - 'Payment added'.	Payment added successfully.	Payment added successfully.	Pass
TC_AP_02	Enter only purpose.	1. Enter purpose. 2. Click on 'Add'.	Test Data	1. Return bus fare.	Toast message - 'Enter a valid amount'.	Payment not added.	Payment not added.	Pass
TC_AP_03	Select only date.	1. Select date. 2. Click on 'Add'.	Test Data	1. 18-4-2020.	Toast message - 'Enter purpose'.	Payment not added.	Payment not added.	Pass
TC_AP_04	Select only time.	1. Select time. 2. Click on 'Add'.	Test Data	1. 8:45 pm.	Toast message - 'Enter purpose'.	Payment not added.	Payment not added.	Pass
TC_AP_05	Enter only amount.	1. Enter amount. 2. Click on 'Add'.	Test Data	1. 50.	Toast message - 'Enter purpose'.	Payment not added.	Payment not added.	Pass
TC_AP_06	Enter purpose and select date.	1. Enter purpose. 2. Select date. 3. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020.	Toast message - 'Enter a valid amount'.	Payment not added.	Payment not added.	Pass
TC_AP_07	Enter purpose and select time.	1. Enter purpose. 2. Select time. 3. Click on 'Add'.	Test Data	1. Return bus fare. 2. 8:45 pm.	Toast message - 'Enter a valid amount'.	Payment not added.	Payment not added.	Pass
TC_AP_08	Enter purpose and amount.	1. Enter purpose. 2. Enter amount. 3. Click on 'Add'.	Test Data	1. Return bus fare. 2. 50.	Toast message - 'Select who paid'.	Payment not added.	Payment not added.	Pass
TC_AP_09	Enter purpose and amount and select 'who paid'.	1. Enter purpose. 2. Enter amount. 3. Select 'who paid'. 4. Click on 'Add'.	Test Data	1. Return bus fare. 2. 50. 3. Manish.	Toast message - 'Select to whom'.	Payment not added.	Payment not added.	Pass
TC_AP_10	Enter purpose and amount and select 'to Whom'.	1. Enter purpose. 2. Enter amount. 3. Select 'to whom'. 4. Click on 'Add'.	Test Data	1. Return bus fare. 2. 50. 3. Raj.	Toast message - 'Select who paid'.	Payment not added.	Payment not added.	Pass
TC_AP_11	Enter purpose and amount and select 'who paid', 'to whom'.	1. Enter purpose. 2. Select date. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 50. 4. Manish. 5. Raj.	Toast message - 'Payment added'.	Payment added with system's current date and time.	Payment added with system's current date and time.	Pass
TC_AP_12	Enter purpose and amount and select 'who paid', 'to whom' and date.	1. Enter purpose. 2. Select date. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 50. 4. Manish. 5. Raj.	Toast message - 'Payment added'.	Payment added with system's current time.	Payment added with system's current time.	Pass

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_AP_13	Enter purpose and amount and select 'who paid', 'to whom' and time.	1. Enter purpose. 2. Select time. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 8:45 pm. 4. 50. 5. Manish. 6. Raj.	All the fields should disappear and 'Who Paid' layout must be visible	Payment added with system's current date.	Payment added with system's current date.	Pass
TC_AP_14	Enter all fields and select 'to whom' equal to 'who paid'.	1. Enter purpose. 2. Select date and time. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 8:45 pm. 4. 50. 5. Manish. 6. Manish.	Toast message - 'Selected for Who Paid'.	'To Whom' set to 'Select Member'.	'To Whom' set to 'Select Member'.	Pass
TC_AP_15	Enter all fields and select 'who paid' equal to 'to whom'.	1. Enter purpose. 2. Select date and time. 3. Enter amount. 4. Select 'to whom'. 5. Select 'who paid'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 8:45 pm. 4. 50. 5. Raj. 6. Raj.	Toast message - 'Selected for To Whom'.	'Who Paid' set to 'Select Member'.	'Who Paid' set to 'Select Member'.	Pass
TC_AP_16	Enter all fields except amount.	1. Enter purpose. 2. Select date and time. 3. Enter amount. 4. Select 'who paid'. 5. Select 'to whom'. 6. Click on 'Add'.	Test Data	1. Return bus fare. 2. 18-4-2020. 3. 8:45 pm. 4. Manish. 5. Raj.	Toast message - 'Enter a valid amount'.	Payment not added.	Payment not added.	Pass
TC_AP_17	Don't enter anything.	1. Click on 'Add'.			Toast message - 'Enter purpose'.	Payment not added.	Payment not added.	Pass

3.1.9 Module - Transactions:

Module Name	Transactions
Test Scenario ID	TS_TR
Test Scenario Description	Verify the Transactions Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_TR_01	Check expense list when no expense was added.		No expense was added in the group.	1. Expense table.	Snackbar message - 'No expense added yet'.	ListView not visible.	ListView not visible.	Pass
TC_TR_02	Check expense list when one or more expenses were added.		One or more expenses were added in the group	1. Expense table.	No snackbar message.	ListView set visible.	ListView set visible.	Pass
TC_TR_03	Check multiple members expense list when no expense was added.	1. Click on 'Multiple Contributions'.	No expense was added in the group.	1. Multiple Members Expense table.	Snackbar message - 'No expense added yet'.	ListView not visible.	ListView not visible.	Pass
TC_TR_04	Check multiple members expense list when one or more expenses were added.	1. Click on 'Multiple Contributions'.	One or more expenses were added in the group	1. Multiple Members Expense table.	No snackbar message.	ListView set visible.	ListView set visible.	Pass
TC_TR_05	Check payments list when no payment was added.	1. Click on 'Payments'.	No payment was added in the group.	1. Advance Payments table.	Snackbar message - 'No payment added yet'.	ListView not visible.	ListView not visible.	Pass
TC_TR_06	Check payments list when one or more payments were added.	1. Click on 'Payments'.	One or more payments were added in the group	1. Advance Payments table.	No snackbar message.	ListView set visible.	ListView set visible.	Pass
TC_TR_07	Check the 'Gone' and 'Visible' functioning of the ListViews.	1. Random clicks on 'Multiple Contributions', 'Payments', and 'Expenses' several times.	Expenses and Payments may or may not be added in the group.	1. Expense table. 2. Multiple Members Expense table. 3. Advance Payments table.	Snackbar message if expenses or payments were added otherwise no message.	One ListView visible at a time provided it contains data.	One ListView visible at a time provided it contains data.	Pass
TC_TR_08	Check expense list OnClick Listener.	1. Click on 'Expense' list row.	One or more expenses were added in the group	1. Expense table.	User redirected to 'Expense Summary' activity.	All details of the expense displayed correctly.	All details of the expense displayed correctly.	Pass
TC_TR_09	Check multiple members expense list OnClick Listener.	1. Click on 'Multiple Members Expense' list row.	One or more expenses were added in the group	1. Multiple Members Expense table.	User redirected to 'Multiple Members Expense Summary' activity.	All details of the expense displayed correctly.	All details of the expense displayed correctly.	Pass
TC_TR_10	Check payment list OnClick Listener.	1. Click on 'Payments' list row.	One or more payments were added in the group	1. Advance Payments table.	User redirected to 'Payment Summary' activity.	All details of the payment displayed correctly.	All details of the payment displayed correctly.	Pass

3.1.10 Module – Settle Debts:

Module Name	Settle Debts
Test Scenario ID	TS_SD
Test Scenario Description	Verify the Settle Debts Functionality

Test Case ID	Test Case Description	Test Steps	Pre Conditions	Test Data	Post Conditions	Expected Result	Actual Result	Status
TC_SD_01	Check Debt list when no expense or payment was added.	1. Check the list. 2. Check the debt column in members table.	No expense or payment was added in the group.	1. Members table.	Snackbar message - 'No expense added yet'.	ListView not visible.	ListView not visible.	Pass
TC_SD_02	Check Debt list when one or more expenses or payments were added.	1. Check the list. 2. Check the debt column in members table.	One or more expenses or payments were added in the group	1. Members table.	No snackbar message.	ListView set visible.	ListView set visible.	Pass
TC_SD_03	Check if no one is paying extra amount.	1. Check the list. 2. Check the debt column in members table. 3. Check expense, multiple members expense and payments table.	One or more expenses or payments were added in the group	1. Members table. 2. Expense table. 3. Multiple Members Expense table. 4. Advance Payments table.		Members paying only that much amount, they owe to someone.	Members paying only that much amount, they owe to someone.	Pass
TC_SD_04	Check if no one is receiving extra amount.	1. Check the list. 2. Check the debt column in members table. 3. Check expense, multiple members expense and payments table.	One or more expenses or payments were added in the group	1. Members table. 2. Expense table. 3. Multiple Members Expense table. 4. Advance Payments table.		Members receiving only that much amount they paid for someone.	Members receiving only that much amount they paid for someone.	Pass

3.2 Non-Functional Testing:

The devices that I used for non-functional tests were Asus Zenfone 3, Redmi 7A and Vivo Y83Pro. Their features are as follows:

- Asus Zenfone 3 - Qualcomm Snapdragon 625, 3GB Ram, Android Oreo.
- Redmi 7A - Qualcomm Snapdragon 439, 2GB Ram, Android Pie.
- Vivo Y83Pro - MediaTek Helio P22, 4GB Ram, Android Pie.

Test	Asus Zenfone 3	Redmi 7A	Vivo Y83 Pro
Performance Test for Alogrithm for first instance.	(1) 5 members - 17 ms. (2) 10 members - 42 ms. (3) 15 members - 73 ms.	(1) 5 members - 20 ms. (2) 10 members - 40 ms. (3) 15 members - 74 ms.	(1) 5 members - 29 ms. (2) 10 members - 52 ms. (3) 15 members - 82 ms.
Performance Test for Alogrithm when ran more than once.	(1) 5 members - 11 ms. (2) 10 members - 15 ms. (3) 15 members - 18 ms.	(1) 5 members - 15 ms. (2) 10 members - 17 ms. (3) 15 members - 26 ms.	(1) 5 members - 11 ms. (2) 10 members - 25 ms. (3) 15 members - 30 ms.
Stress Test.	Worked smoothly.	Not executed - Incomplete.	Not executed - Incomplete.
UI Test.	Layouts, Data movement and pop-ups working fine.	The 'Payments' text in activity Transactions did not fit correctly because the screen was less wide. Other things worked fine.	Layouts, Data movement and pop-ups working fine.
Compatibility Test.	Compatible.	Compatible.	Compatible.
Instability Test.	All components installed correctly.	All components installed correctly except the edges of the logo did not coincide correctly leaving some white space.	All components installed correctly.

Chapter 6 Results

1. Test Report:

Project Title	Tested By	Module Count	Total Test Cases	Total Passed	Total Failed	Total Incomplete
Settle Group Expenses	Sumit A. Bane	10	122	119	2	1

2. Project Summary Report:

2.1 Overview:

An android application to settle the expenses among the members of a group during a trip. People can enter details of all the transactions and get rid of the tedious calculations at the end of the trip.

2.2 The Problem:

- It is really a difficult task for a person to calculate the pending amounts of each and every person in the group even with the help of a calculator. Even if he tries to settle by paying back amounts by recalling every transaction, it will consume a lot of time. Also they have to note down all the expenses for the same.
- This problem is faced by every group in the world unless it happens to be a family and/or the members don't mind the money that other member(s) owe them.
- Currently there are lot of applications like Settle Up, Tricount and Splitwise. They all have some special features of their own.

2.3 The Solution:

- The app that I have developed will solve this problem of settling the debts of the members without putting them into any kind of calculations.
- The simple UI of the application will make it easy for the people to add

transactions and also understand how to settle among them.

- People can read the instructions which they can find on the home page, this will help them understand the app better.

2.4 Highlights:

- Members can add their budget which will remind them at times when they are spending more than they should.
- Each member can add To-Do for themselves in the app itself.
- Once the transaction is added nobody can edit or delete. There can be possibilities of members accusing the member (who is handling the app) for manipulating the amounts.

Chapter 7 Conclusions

It has been an amazing year while developing this application at the same time making the project report, giving exams, submitting assignments, giving aptitudes, concerning faculties about future, being a part of the fest and what not. I have learned many things about Android now and still wish to learn more in the future. Many times I asked my friends for help and sometimes I helped them. I noticed that everyone in the class was helping each other. I tried to cover everything about my project in this report and reached to these conclusions.

1. Significance of the system:

This application is useful for anyone in the society when they travel with their friends, families or colleagues. I have kept the UI simple and easy and also I have added some instruction for users to understand the app better. Besides adding expenses and payments users can also add their budgets and to-do during their trip. They will be reminded of their balance amount when they are about to spend 90% of their total budget and when they have spent the entire amount.

2. Limitations of the system:

- I have kept the maximum no of members that can be added in a group to 15 to make sure that there is no heavy load onto the database.
- Once the expense or payment is added, it cannot be edited or deleted.

3. Future Scope of the project:

- The database can be switched from sqlite to Firebase Realtime Database for more quick and smooth functioning.
- The app can be synchronized with cloud server so that every member can view or add transactions.
- I am planning to review the app from my friends and get their suggestions to improvise wherever possible.

Bibliography

- Android Studio - <https://developer.android.com/docs>,
<https://stackoverflow.com/>, <https://github.com/>, <https://codinginflow.com/>.
- SQLite - <https://www.tutorialspoint.com/sqlite/index.htm>.
- Diagrams - <https://www.lucidchart.com/pages/>, <https://drawio-app.com/>.
- Test Cases - <https://www.office.com/launch/excel?auth=1>.
- Non-Functional Testing - <https://www.cigniti.com/blog/non-functional-testing-aspects-of-mobile-apps-3/>.
- Settle Up -
[https://play.google.com/store/apps/details?id=cz.destil.settleup&utm_source=global_c
&utm_medium=prtnr&utm_content=Mar2515&utm_campaign=PartBadge&pcampaign
id=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1](https://play.google.com/store/apps/details?id=cz.destil.settleup&utm_source=global_c&utm_medium=prtnr&utm_content=Mar2515&utm_campaign=PartBadge&pcampaignid=MKT-Other-global-all-co-prtnr-py-PartBadge-Mar2515-1).
- Kulkarni, S. (August 2018). Software Project Management. *An Overview of Project Planning*, pages 3.1 – 3.39. Tech-Max Publications, 1st edition.
- Shetty, D.B. (2019). Software Quality Assurance. *Fundamentals of Testing*, pages 37 – 99. Sheth Publishers, 2nd edition.
- Vávra, D. (May 2012). Mobile Application for Group Expenses and Its Deployment.
- Logo - <https://www.freelogodesign.org/>.
- Project Idea - <https://nevonprojects.com/year-projects-for-computer-engineering/>.