**BUDGY**

By

&roid (Group 2)

Alisher Matianiu – 300319513

Jenukshan Colombas – 300323000

Kosha Raval – 300324116

Rebecca Moses D’mello – 300322984

Supervised by

Ivan Wong

**Abstract**

We know that the basic needs of humans are food, clothing, and shelter. In order to achieve those needs and also have a lavish life an individual usually spends lots of money. Tracking those expenses sometimes becomes challenging. There are many applications available in market which help in some or the other way like “Mint” which links the app with the bank account and alerts the user when the balance is running low, “YNAB” which helps user to plan on how they can utilize money, Good budget app which allows many people to share the money.

We aim to make our system less human dependent. Our system will make it easier to analyse the expenses and manage money accordingly. The Main Screen will also display how much money he has saved from the past and what are his current expenses. It is similar to the Scotia Bank Mobile application where the user can see how much credits are left and how much credits have been utilized. Apart from managing budget there is an additional feature where the user can keep track of how much tax has been applied on his expenses and generate a report month wise which can be shared.

**Index**

|  |  |  |
| --- | --- | --- |
| Sr. No. | Particulars | Page No. |
| 1 | Overview | 4 |
| 2 | Features and technical challenges | 5 |
| 3 | Installation guide | 6 |
| 4 | Functions and user manual | 7 |
| 5 | Incomplete functions | 16 |
| 6 | Disclaimer | 19 |
| 7 | Reference | 20 |
| 8 | Individual Report: Alisher Matianiu – 300319513 | 21 |
| 9 | Individual Report: Jenukshan Colombas – 300323000 | 24 |
| 10 | Individual Report: Kosha Raval – 300324116 | 26 |
| 11 | Individual Report: Rebecca Moses D’mello – 300322984 | 28 |

**Overview**

**Video Link:**

**https://www.youtube.com/playlist?list=PLLU8dJrXuHnXwY8o1cw6ggQkOAmfeOn2\_**

Budgy is a budget management application which helps users in keeping track of their financial activities like the money earned and also the amount spent on various things like grocery, rent etc. There are different activities which are involved in the application like an attractive splash screen which displays a customized icon on the screen for a few milliseconds. After the Splash Screen the control goes to the Login Screen and Registration Activity which allows the user to provide some personal information like Name, Email Address, Password and Confirm Password. All the data provided are saved using SQLite Storage and can be used by the user to access the application again.  After the user successfully registers, he is directed to the login activity which validates the data i.e., the username and also the password.

After the user has logged in successfully, he has access to the main screen which is the dashboard. The Dashboard provides users with the functionality to add the category of expenses and the amount spent on it. Categories once added are displayed on the screen along the amount. The dashboard also displays the categories performed in the last 7 days which helps the user to get an insight of what all activities were performed and to track his expenses. Basically, the dashboard displays budget, expense and savings to get an idea about how much money he has, the expense he has performed or how much he has saved. It has a model which allows the user to add category and the amount. The dashboard also allows the user to upload the receipt of the bill or invoice which he has so that he can verify it in later stages for verification purposes. There is a navigation bar which makes the flow of application quite easy. The options provided in the navigation bar are the Dashboard, Fixed Income and Expense, Reports, Settings and Invite.  It provides flexibility to the user to navigate through the application and select services of their choice.

Setting is the activity which allows the user to view the credentials like Name, Username and Password. The Expense Category has a button which allows the user to add the type of categories which are reflected on the dashboard along with the tax. Manage Income and Expense activity has two functionalities one is the Income and the other is expenses which are associated with a button and allows the user to add Income and expenses which gets reflected into the dashboard. The Report activity displays the report of all the activities. There is an invite activity which allows users to share the application with other people via email or by message so that other people can download the application and access its services.

**Features and technical challenges**

There are various features involved in "Budgy" one of the main is keeping track of income and expenses easily and efficiently. It allows the user to upload the invoice of the purchase so that he is able to easily figure out what the expense was related to. In various other applications like "Mint" and "YNAB" maintaining invoice features is not provided. The main goal behind developing these applications was providing features which people would use in their day-to-day activities and the most important thing was managing finance.

During the implementation of the application, we came across few technical challenges where few were complicated to troubleshoot while others were easy to troubleshoot. Minor challenges involved having a menu bar only in the splash screen which was fixed by setting the theme, combining activities and making it run accurately was kind of time consuming as one minor change undone was creating a huge problem in the application, the navigation bar wasn’t able to display the list of all the other activities which was fixed by making changes into the manifest file, unable to include image using preferences.

Major challenges faced during the development process involved designing and developing the main screen which was the dashboard. Initially, we had used the hash map for displaying the categories and amount but later after learning about recycler view, we were able to make it more efficient.  Some tasks which involved the database handler were also creating major issues and the operations performed on database were difficult to troubleshoot.

In general, debugging Android applications is challenging, since there are lots of interconnected files and code pieces. Most of the errors come up in run-time, that makes it difficult to troubleshoot. Database related debugging lacks the option to directly view the database on the emulator. Instead, wherever a developer wants to check the database, the underlying file in the system has to be copied and then opened in the SQLite browser. Having to go through these steps during each database update or fix, creates an overhead.

Other challenges took place when an in-built Android Studio emulator was hanging, unable to run the application properly. At the beginning we were trying to find problems in the code, but the solution was to restart the emulator and reinstall the app from scratch. Sometimes our computers would also cause problems while running an application, and that might be due to an intensive load, that the system is experiencing when building Android applications.

**Installation Guidelines**

Minimum Android Version to run the app:

Please follow these steps to run the application:

1. Unzip the archive to a desired location on your computer.
2. Open Android Studio >> Open Existing Projects
3. Navigate to the project and click “Open”
4. If an SDK error pops up, click “Ok”
5. After the project is opened and gradle build finished executing, select Nexus 5X emulator and click on the green play button to install the application.



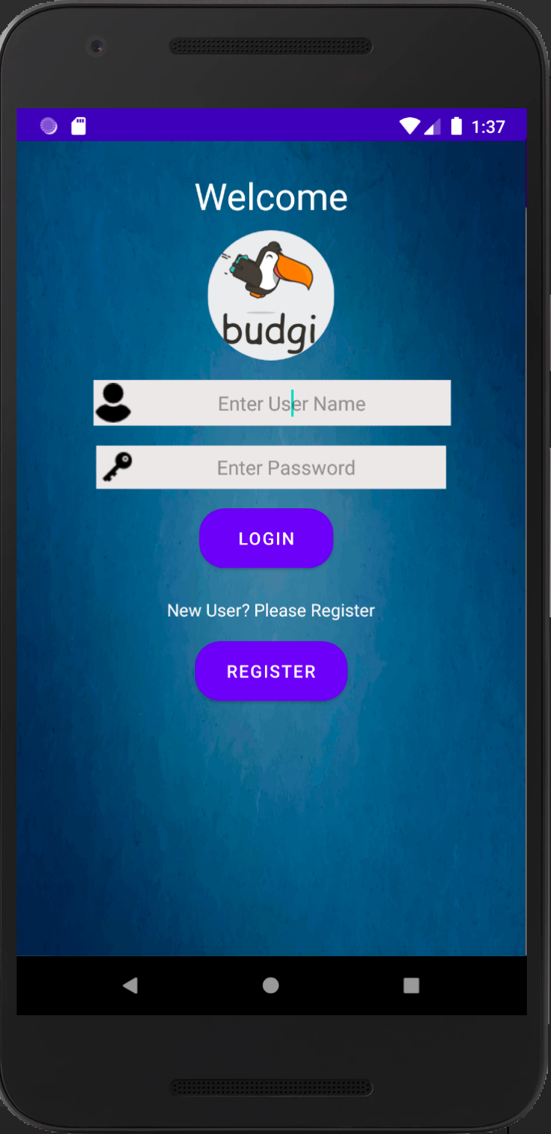
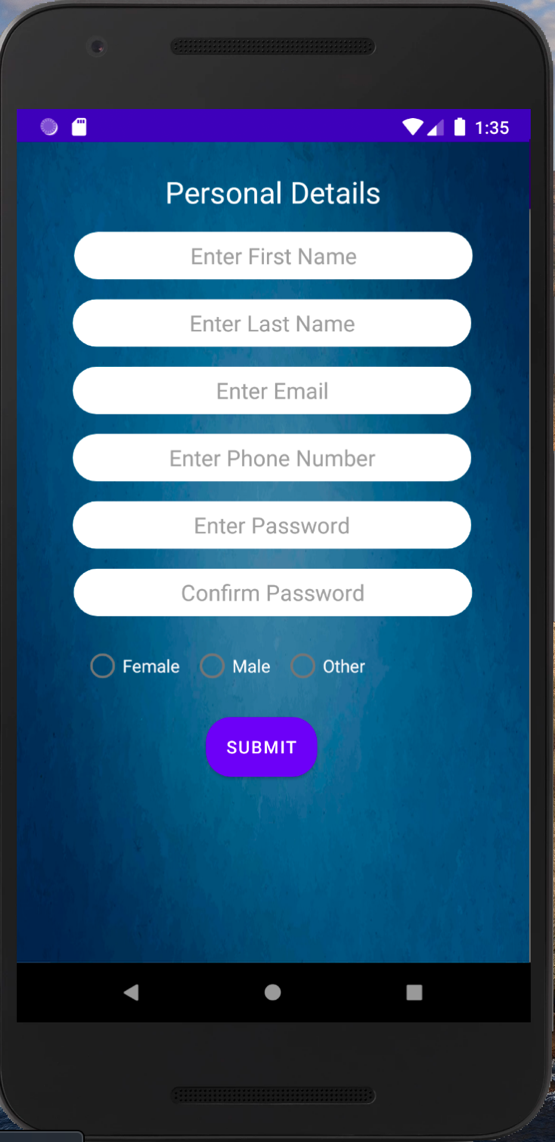
1. If you are unable to run the application, try to copy the local.properties file from your other applications created on your system and replace the local.properties file in the current folder.
2. When the app is installed, it opens automatically.
3. As the app is installed, it can be opened just by clicking its icon.



**Functions and user manual**

**Registration:**

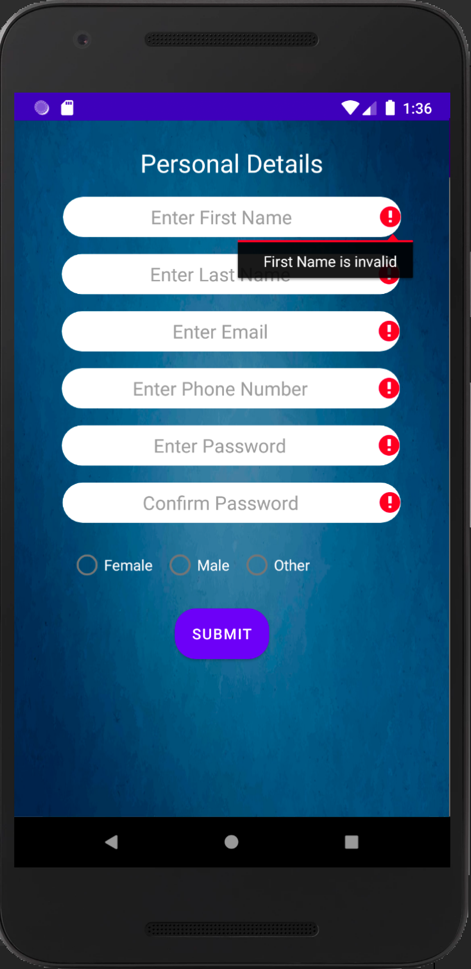
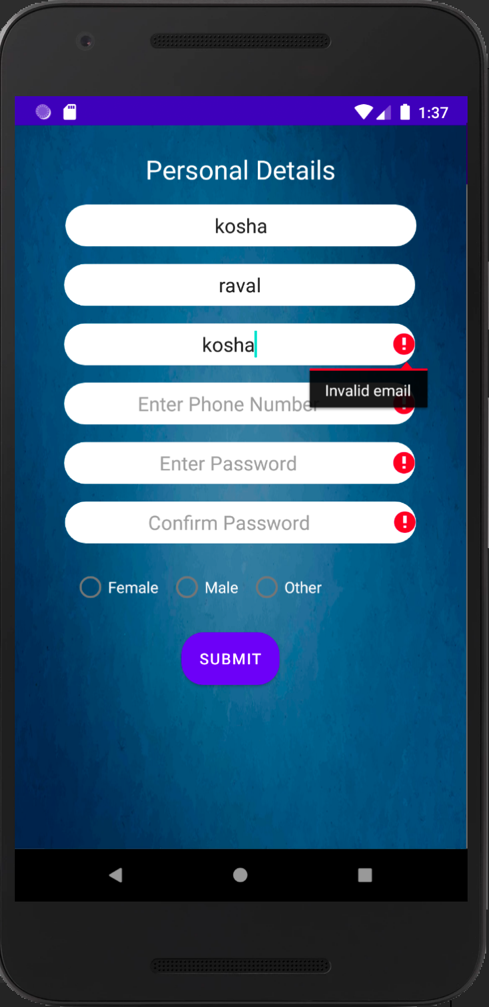
New user will click on the REGISTER button and will be directed to the registrations form. All fields on the form are mandatory. On submit the user is registered and is taken to LOGIN page.

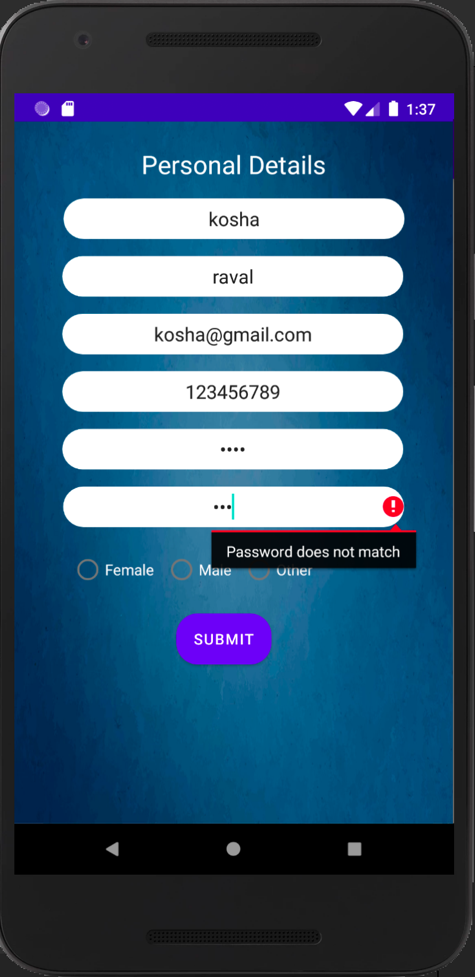
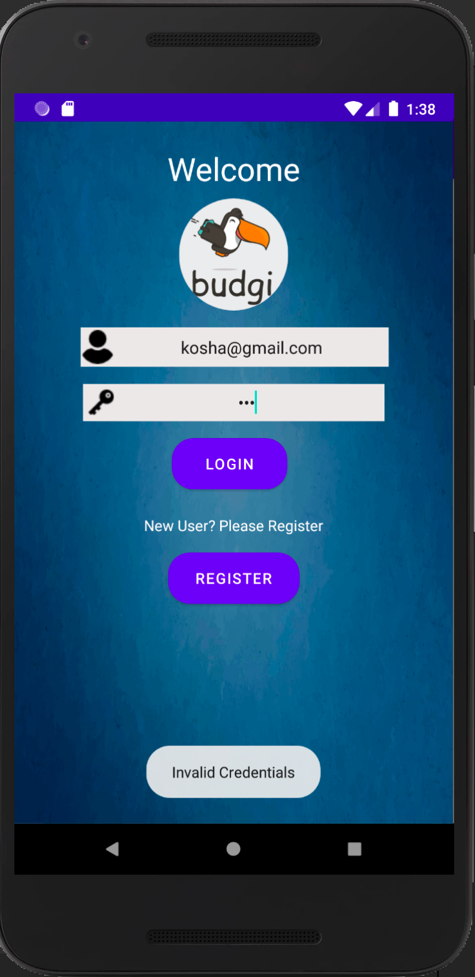
 

When the user clicks on the submit button without entering any of the values in the field the page will give errors as shown in below images. There is email address for format validation for the enter email field and which will be validated on submit button.

**Login**:

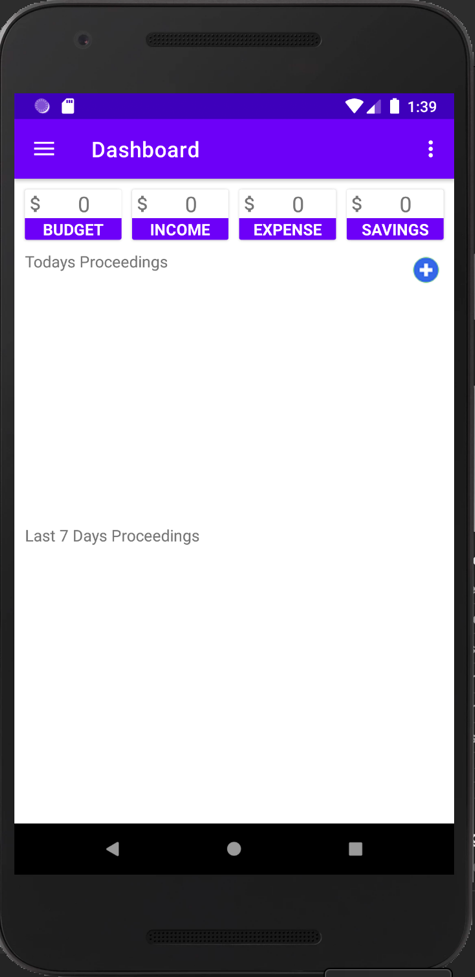
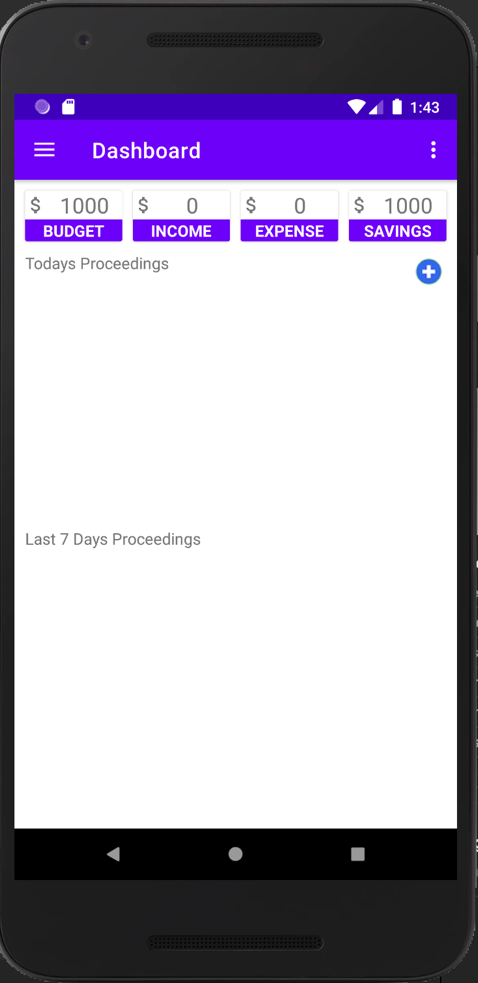
The user will login into the application using the email address and password. The user is validated with the database record. If the user credentials are invalid an error message is displayed as a toast message. If the user credentials are valid then the user is taken to the DASHBOARD.

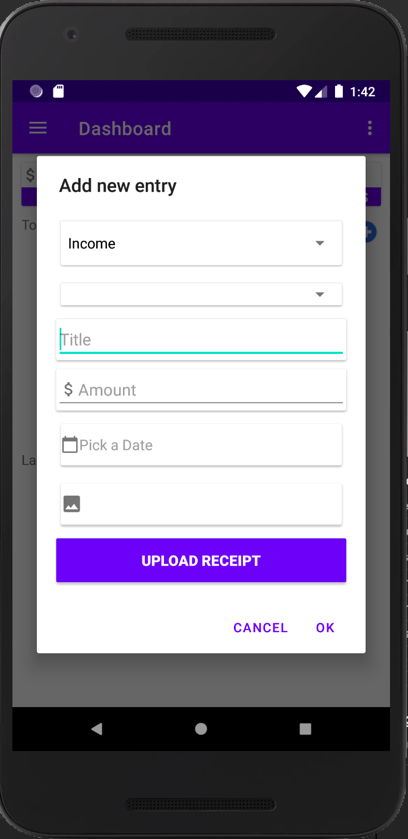
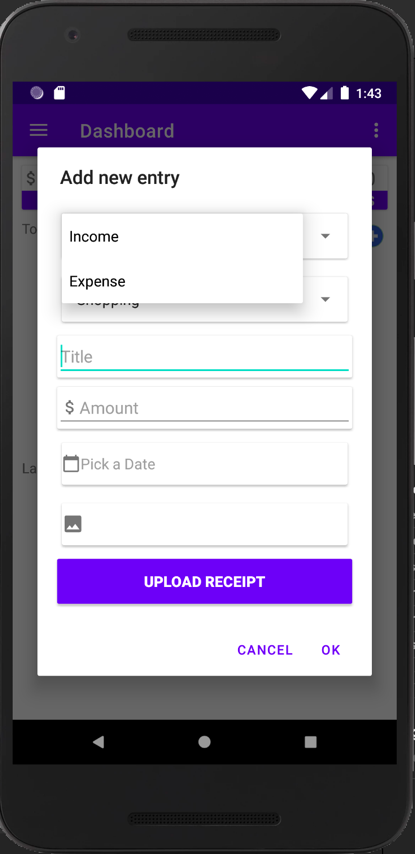
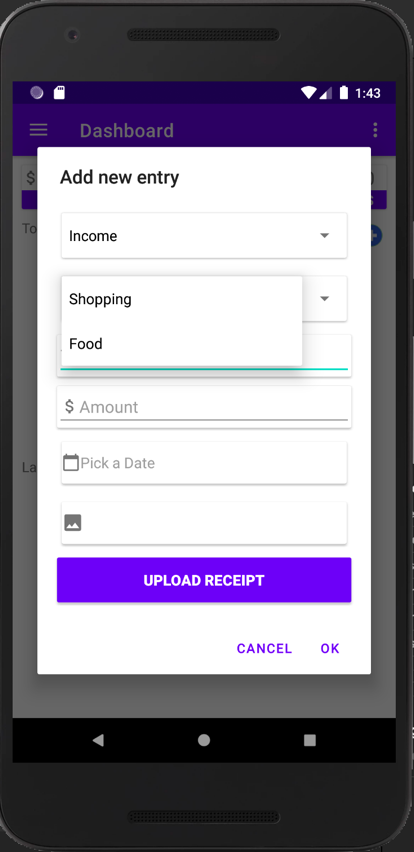
**Dashboard:**

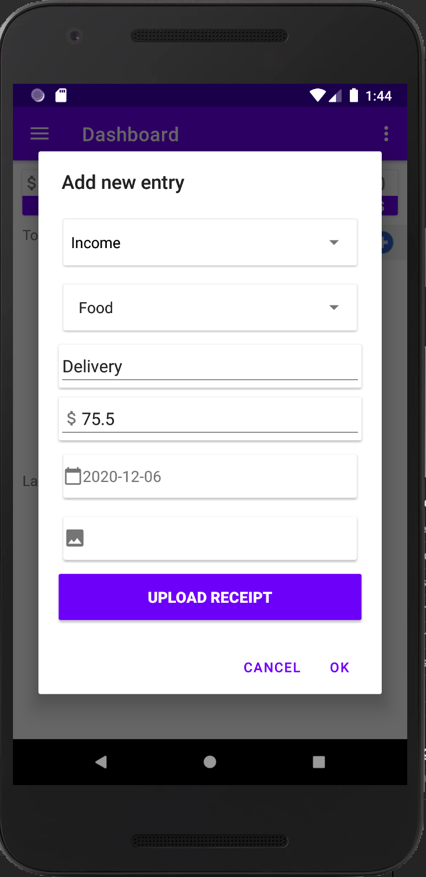
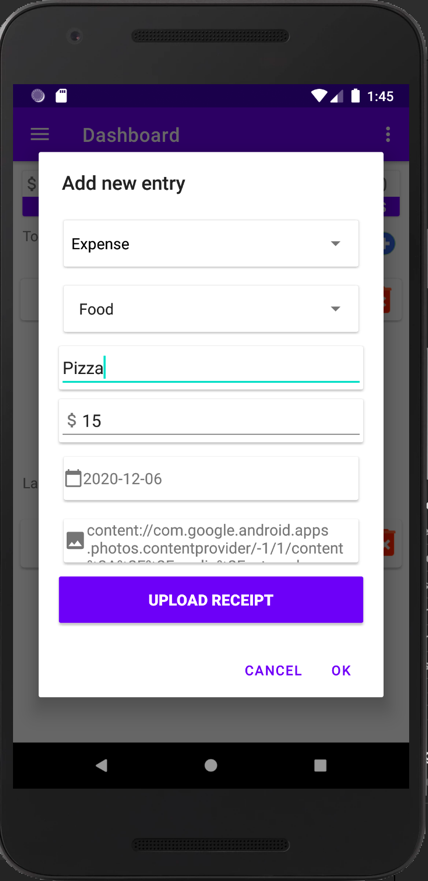
For 1st user this fragment will be like a blank canvas. Once the user adds budget and categories on the settings page, they will be reflected on the dashboard. The cards in top most row show monthly amounts: Budget, Income, Expenses, Savings. Income and Expenses cards are addition of fixed income and variable income and fixed expenses and variable expenses respectively. Savings card is calculated by subtracting the expenses from the budget. These cards are refreshed whenever there is a new transaction added or a transaction is removed.

**Add Transaction:**

For adding a new income or expense entry the user needs to click on the blue + sign. A customized dialog will open. All fields on this dialog are mandatory. The 1st spinner is for selecting the type of the transaction: Income or Expense. The 2nd spinner is populated with the categories user entered on the settings page. The title is a more specific value of the transaction. For example: if the category is Food then in title it could be a restaurant name. The user then enters the amount of the transaction. Picks a date for transaction. Last field is where the user adds a receipt image. On clicking ok the new transaction is added to the dashboard in today’s proceedings list and in the last 7days proceedings.

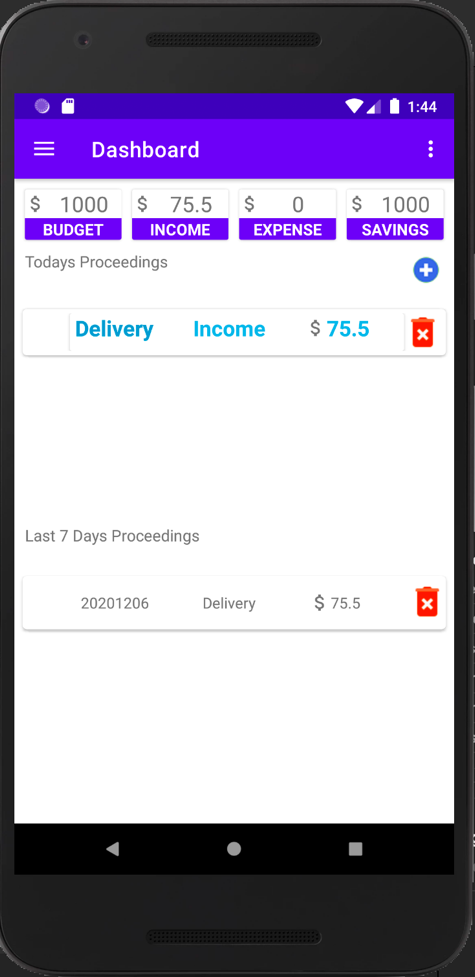
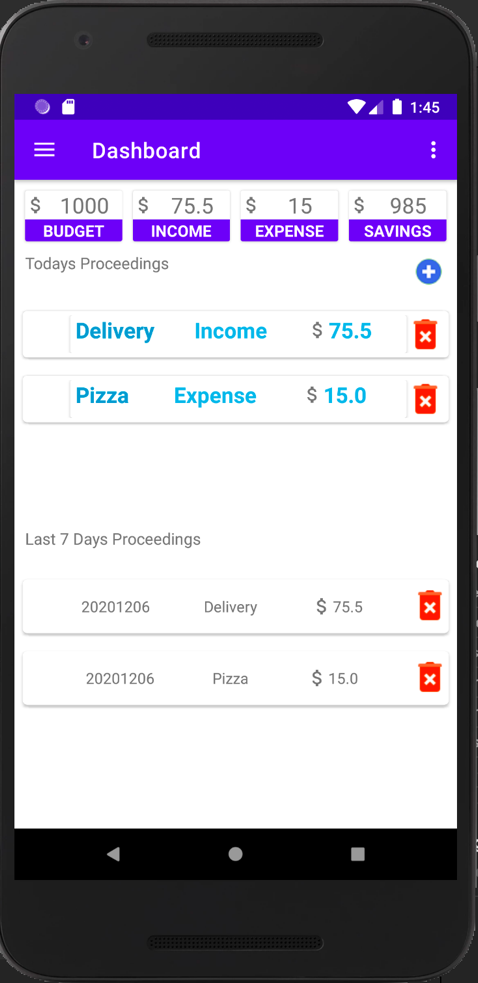
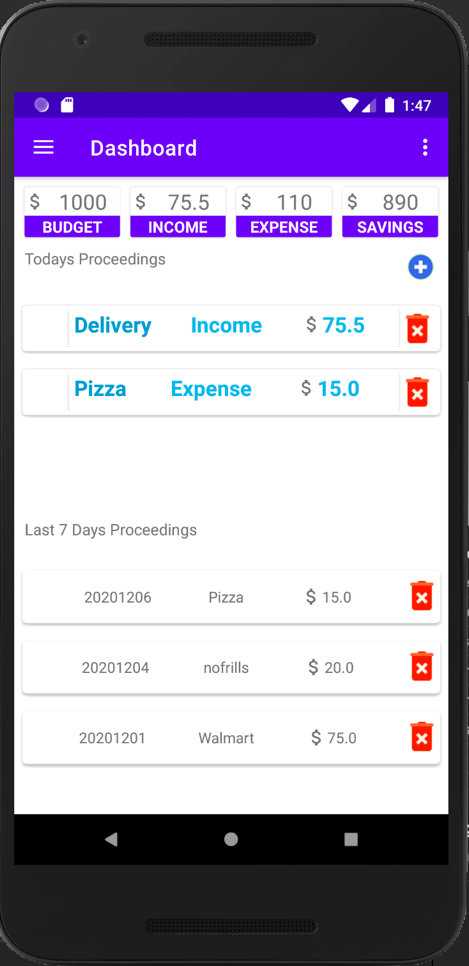
 

**Today’s Proceedings View:**

This recycler view displays current days transaction that are added from Add Transaction Dialog. They could be both income and expense. The transactions can be deleted.

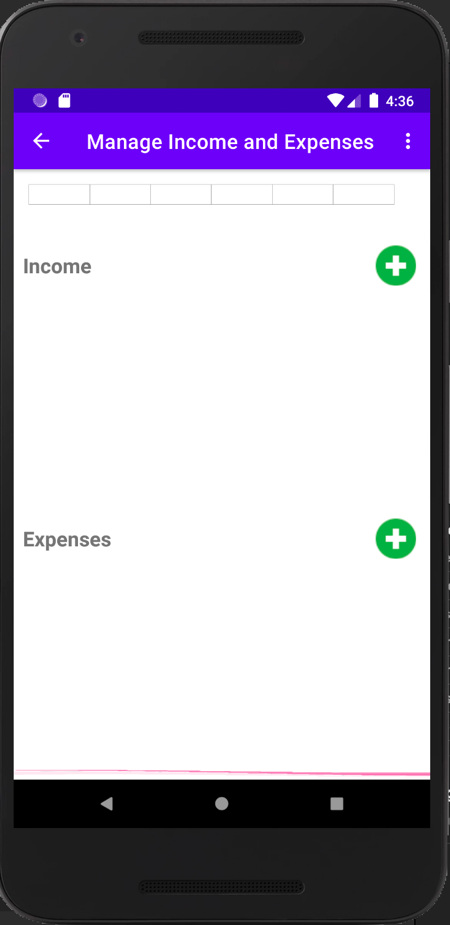
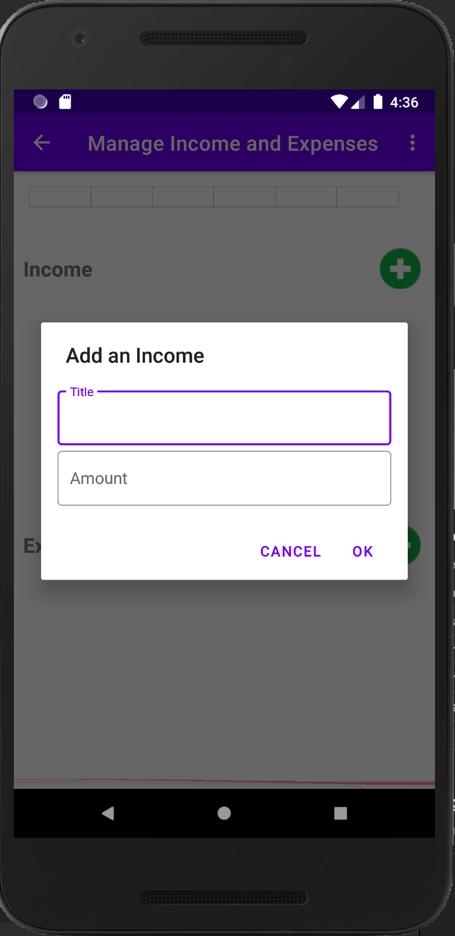
**Last 7 days Proceedings View:**

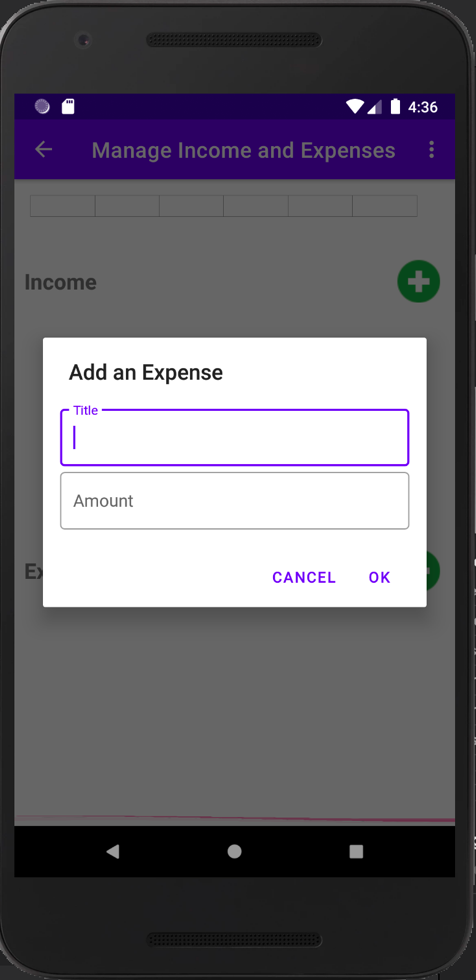
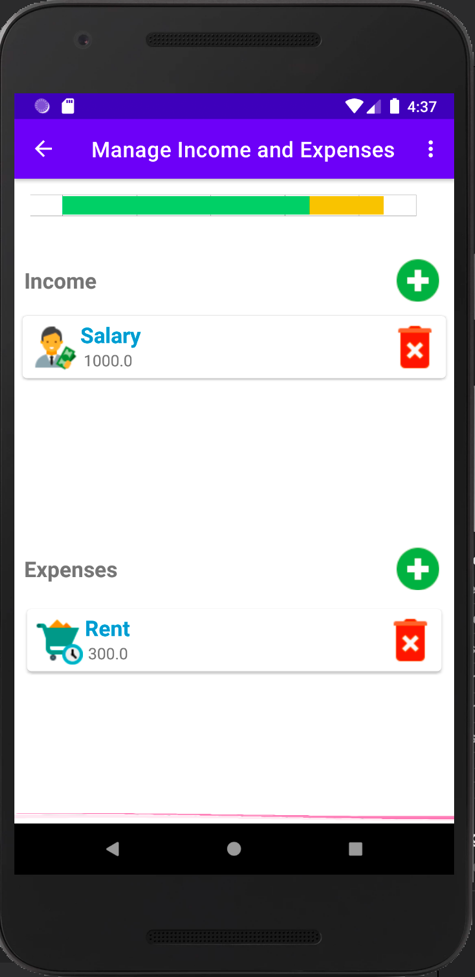
This recycler view displays past 7days transaction that are added from Add Transaction Dialog. They could be both income and expense. The transactions can be deleted. The data is displayed in descending order of dates. The user can add a transaction for a day that is past then 7days but cannot delete it.

**Fixed Income and Expense:**

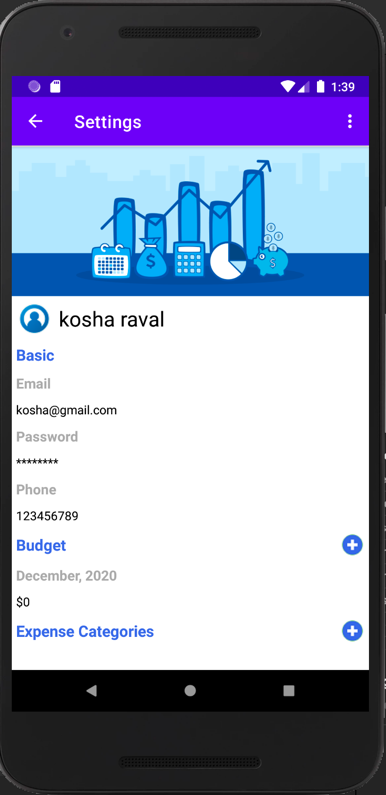
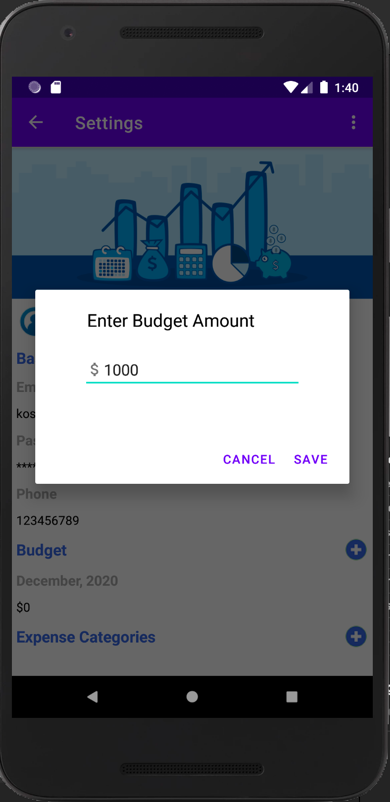
On this page the user can add fixed income and expenses that are constant for month. New fixed income and expense can be added by clicking on W plus button which is present in both the income and expense list respectively. Progress bar on the top shows fixed income and expense progress for that month. The fixed income and expenses can be removed after clicking on the button that is visible beside every entry.

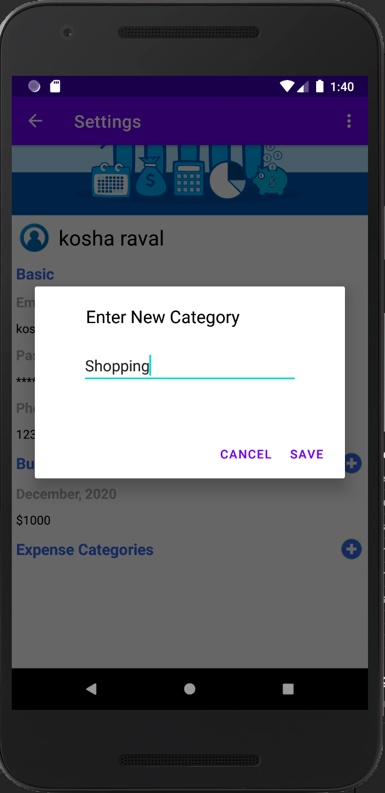
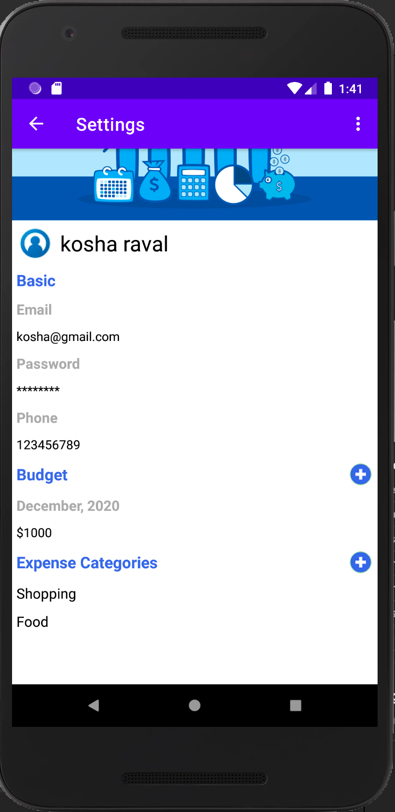
 

**Settings:**

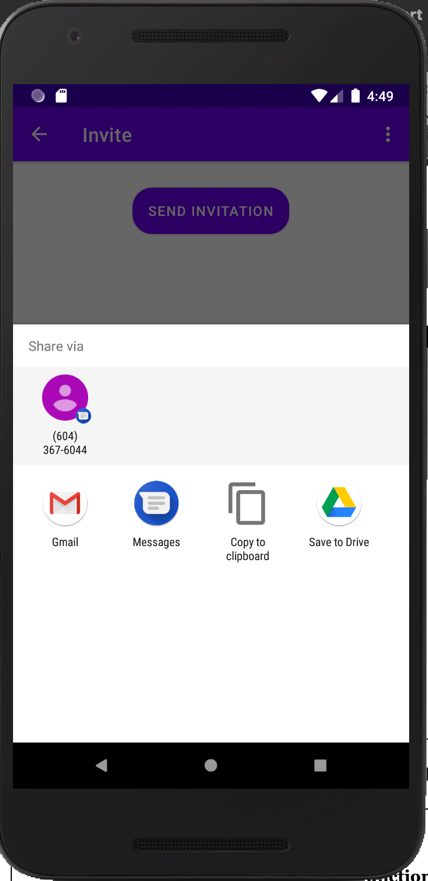
For 1st time user the Dashboard will be blank. The user needs go to the SETTINGS page from the navigation. On settings page user information is visible. The user has to enter monthly BUDGET and should add CATEGORIES of income or expenses that he/she will be using. For example: shopping, food, bills. The Budget is added by clicking on the blue+ button beside BUDGET. Multiple categories are added by clicking the blue plus button next to CATEGORIES. Below screenshot the steps.

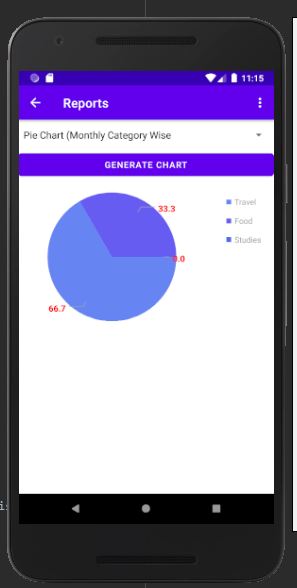
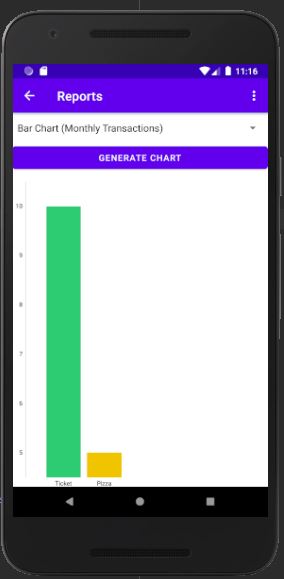
**Invitation:**

On this page there is a dummy link provided to the put extra method which allows people to download the application and access the services provided by the application. The link can be shared via Gmail, SMS.



**Reports:**

On this page the user can select either bar chart or pie chart and click on Generate Chart button to view the graphical representation of all records that they have inserted throughout the month.

**Incomplete functions**

Our project log drafted for the previous Interim report is as follows and have achieved all the tasks as per the due date mentioned.

The future scope functions would be storing the image receipt uploaded by the user on add transaction into database and displaying that image as image icon on the recycler view. The image would be expanded into a new fragment when that particular transaction item is clicked.

Editable user data in the Settings page. When the user clicks on email / name / phone / password custom dialog windows are meant to pop up to allow editing the user data and then updating records in the database for the corresponding entry in the user table.

Category status: Active / Inactive. This status was meant to exclude some categories and their underlying income / expense data, while displaying the information on a dashboard page. The status could be set on the Settings page for each category item in the recycler view. The default status is Active.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Date | Description | Assigned to: (Member name: hours spent) | Status | Date Due |
| Week 1 | Sep 18, 2020 | Discussion of Project Ideas | All Members: 3 | Completed | Sep 24, 2020 |
| Week 2 | Sep 25, 2020 | Project Proposal and User Interface | Kosha: 2  Jenukshan:2  Alisher:2  Rebecca:2 | Completed | Oct 1, 2020 |
| Week 3 | Oct 2, 2020 | UI designing | Kosha:2  Jenukshan:2  Alisher:2  Rebecca:2 | Completed | Oct 8, 2020 |
| Week 4 | Oct 9, 2020 | UI Integration of all activities | Kosha:3  Jenukshan:3  Alisher:3  Rebecca:3 | Completed | Oct 15, 2020 |
| Week 5 | Oct 16, 2020 | Detailed discussion of individual functionality | Kosha:4.5  Jenukshan:4.5  Alisher:4.5  Rebecca:4.5 | Completed | Oct 22, 2020 |
| Week 6 | Oct 23, 2020 | Implementation of Individual Functionality | Kosha: Dashboard (3)  Jenukshan: Fixed Income (5)  Alisher: Settings (3)  Rebecca: Dashboard (3) | Completed | Oct 29, 2020 |
| Week 7 | Oct 30, 2020 | Interim Report, Integration & Testing of all Functionalities and Resolving issues. | Kosha: 5  Jenukshan:5  Alisher:5  Rebecca:5 | Completed | Nov 5, 2020 |
| Week 8 | Nov 6, 2020 | Analyzing if more features can be added to make the app more efficient | Kosha:2  Jenukshan:2  Alisher:2  Rebecca:2 | Completed | Nov 12, 2020 |
| Week 9 | Nov 13, 2020 | Working on Database | Kosha:2  Jenukshan:2  Alisher:2  Rebecca:2 | Completed | Nov 19, 2020 |
| Week 10 | Nov 20, 2020 | Finalizing and testing Database connectivity | Kosha:2  Jenukshan:2  Alisher:2  Rebecca:2 | Completed | Nov 26, 2020 |
| Week 12 | Dec 06, 2020 | Final Presentation | All | Completed | Dec 4, 2020 |

**Disclaimer**

The application was developed by referring to various applications like "Mint" and few others. But the features designed and implemented in Budgy are completely unique and doesn't not come from any other application directly. They were developed with the flow of the application and the ideas and content covered in the course.

**References**

The Image used for the application i.e., for the Splash Screen, Login Page was taken from the link:  <https://99designs.ca/inspiration/logos/budgeting>

Book - Corinne Hoisington. Android Boot Camp for Developers Using Java, A Guide for Creating your first android app, Third Edition.

Trouble Shooting and finding solutions to a few issues we have referred to Stack Overflow.

**Individual Contribution**

**INDIVIDUAL REPORT Alisher Matianiu 300319513**

**Task description**

My task was to create a settings page that allows a user to view their details, create categories that will be used while adding income and expense data in the dashboard page. In addition, a custom budget amount for the current month can be added in the settings page, that is also used in the dashboard page to calculate savings for the current month.

Report: wrote Installation guide, contributed to Features and technical challenges and Incomplete functions.

Files created:

1. Models folder >> Budget, LoggedInUser
2. Settings folder >> CategoryAdapter, Settings, SettingsViewModel
3. layout folder >> fragment\_settings, recyclerview\_categoryitem, settings\_add\_budget, settings\_add\_category
4. strings\_settings and edited images used in layout files.

Files contributed to:

1. DatabaseHelper.java >> created a table to store budget data, and methods to

create/add/update the data.

1. LoginActivity >> included the code to save logged in user data and budget amount.

**Task 1: Displaying User Data**

The first objective was to fetch all logged in user data in order to display it on my page. Since the settings page is not the first page that opens up after a successful login, there was a need to save the user data upon logging in and making this data available throughout the application. So, my solution was to create a LoggedInUser class that contains a static User object that keeps all details of a logged in user, once it is retrieved from the database. This approach has a benefit of accessing a database only once to have all data in place.

Having all the data in a static class, I was able to access it on the settings page just by calling a class name “LoggedInUser” and the underlying methods in order to get data and assign it to the respective text views for displaying full name, email and phone.

In the proposal we mentioned that the user data fields will be editable and the changes will be saved in the database by updating the underlying user data fields. Unfortunately, I was unable to complete this functionality.

**Task 2: Adding Categories**

The second objective was to create a category adding function and being able to display all categories associated with the user in the categories section. For this purpose, when the plus icon next to the category section is clicked, the dialog view pops up and asks a user to enter a category name. Once the category name field is entered a user can click on the “save” button to save a category. If a field is empty, it will not be saved in the database. Otherwise the information such as user id, category name and status is saved in the database under category table.

The purpose of the status field in a category table was to distinguish between Active and Inactive categories. The status was meant to be editable in the category recycler view items, so that the user could exclude some categories with underlying income/expense records. Following that, on the dashboard page it could be possible to list only Active categories and income/expense records associated with them. This was an additional feature we have thought of, but unfortunately it was not implemented.

In order to display a list of categories associated with the logged in user, a recycler view was used. Whenever a settings page is opened or a new category is added, the list that keeps categories is updated by fetching all category names from the database. Consequently, the recycler view is updated based on this list.

**Task 3: Adding and Updating Budget Amount**

The third objective was to allow a user to edit the budget amount for the current month. This section was initially planned to display tax related information, but later we have decided to replace it with a customizable budget amount for the current month. This budget data is used in the dashboard page to calculate savings. Each user has a custom budget amount during every month. Upon logging in the budget data for the current month is retrieved from the database. If the data is not in the database, the default record with a budget amount of $0 is created and stored in the database. To make the budget amount accessible in both dashboard and settings pages, I have added it as a static member of a LoggedInUser class. As the budget data is retrieved or edited, the budget amount in LoggedInUser class is updated.

In the settings page, when the plus icon next to the budget section is clicked, the dialog view pops up and asks a user to enter a budget amount. Once the budget amount is added a user can click on the “save” button to update the budget amount. If a field is empty, budget data will not be updated. Otherwise, the budget information associated with the user in the current month and year is updated. LoggedInUser class’ budget amount is updated as well.

**In addition**

Modified application flow by terminating activities when starting other activities, since the activities got stacked up, which led to inconsistent application flow. (For example, when the user has successfully registered and moved on to a login page, on the back button click it was possible to come back to the registration page.)

**INDIVIDUAL REPORT Jenukshan Colombas 300323000**

**Task description**

In this project, my task was creating the fixed income and expense page. On this page the user can add his/her monthly fixed income and expenses. The fixed income and expenses are respectively shown in the recycler views. There is a progress bar that shows the amount of income and expense for that month.

Report: Contributed to Features and technical challenges and Incomplete functions, editing and combining of the video presentation. Designed the base of the project.

Files created:

1. Models folder >> FixedAmount
2. Fixed folder >> FixedAmountItem, FixedDailog, FixedIncomeExpense, FixedIncomeExpenseAdapter, FixedIncomeExpenseViewModel
3. Reports folder >>
4. layout folder >> fragment\_fixed, example\_fixed\_amount, dialog\_fixed

Files contributed to:

1. DatabaseHelper.java >> created a Base database file and created FixedAmount table and methods to create/add/update the data.
2. Helped in Solving all the technical issues on every page.

**Task 1: Creating base of project.**

In this task I created the base design and layout of the project which include the Main activity, the navigation bar and mapping of the navigation bar, menu bar and navigation header bar. This helped in creating and maintaining the application layout throughout the development. Created DatabaseHelper file with base functions for the group and created FixedAmount table and methods inserting, updating and deleting data.

**Task 2: Manage Fixed Income Expense:**

The requirement of this task was that the user should be able add fixed income and expense throughout the month into the system. Created a customized alter dialog box which is used for inserting both fixed income and fixed expense. The alert box has two mandatory field to be entered. The title of the fixed income or expense and the amount. On clicking on ok button, the recorded would be added to the respective recycler view of income and expense and it is also inserted into the FixedAmount Table. The recycler view created for displaying the fixed income and expense is also a REUSABLE component and it allows the user delete a record by clicking delete button on the right. A recycler view is used to display the title of the record, amount, a small icon differentiating between income and expense and a delete button. This is done by using fragment\_fixed, example\_fixed\_amount, dialog\_fixed layout and FixedAmountItem, FixedDailog. On top of the fragment used a ratio bar to show the ratio between the fixed income and expense. The green color is for the income and the yellow part is for the expenses. This ration changes as an when there is update in the fixed income and expense recycles view i.e. when a user adds a new record or deletes an existing record.

**Task 3: Reports**

The objective of this task was to give user a graphical representation of the all transactions they have entered throughout the month. I created a pie chart using information stored for fixed and variable income and expense. The pie chart is generated on the basis of all the categories present in the category table which are inserted by the user on the settings page. This giving user information about how many percentages of income or expense does each of the category holds.

**In Addition:**

I have helped in solving all the technical challenges and issues and bugs fixes on every page. Helped in creating a better designed GUI of the application. Consolidated the group video recording.

**INDIVIDUAL REPORT Kosha Raval 300324116**

**Task description**

My task in the project was creating the DASHBOARD of the application. The dashboard page shows all the information in one place. The 4cards on the top most row give information about the Budget, Summation of Fixed Income and Variable Income, Summation of Fixed and variable expense, savings for a particular month. This is followed by two recycler view: 1st shows all the transaction for today and 2nd shows all the transactions in past 7 days. New transactions are added through a customized alert dialogue.

Report: Functions and user manual, formatting of the final report, consolidating the final report and individual reports

Files created:

1. Models folder >> CategoryItem, DailyTransaction.
2. Dashboard folder >> Dashboard.java, AddTransactionDialog, DailyTransactionAdapter, DailyTransactionItem, DashboardViewModel, WeeklyTransactionAdapter
3. layout folder >> fragment\_dashboard, add\_transaction, recyclerview\_categorylist, recyclerview\_weeklyreport
4. values >> strings\_dashboard.

Files contributed to:

1. DatabaseHelper.java >> created a table Daily Transaction and Category, and methods to create/add/update the data.
2. Consolidated all the code from all the group members.

**Task 1: Dashboard and Adding new transaction:**

Dashboard is the first page that opens after successfully login. For a first-time user the dashboard page is like a blank canvas there are no values filled in the top row and there is no data to be displayed in the recycler views. Once the user enters the budget and adds he/her customized categories, the dashboard is update with this information from the database. The budget is shown by querying to the Budget table. The Income card is addition of the fixed incomes quired from FixedAmount table and variable income quired from the DailyTransaction Table. Similarly, the Expense card is also calculated. The Saving card is calculated by subtracting the value from the Budget card and the Expense card.

New transactions are added by the opening a customized alert dialog box that is opened on click of a button. In this alert box the user has to select from two spinners which are Category Type i.e., Income or Expense and Category List which the user has entered on the Settings page. The user must enter a title, amount and pick date for the transaction. On clicking ok button, the transaction is added to the DailyTransaction Table. The customized alert box is implemented in using add\_transaction layout and AddTransactionDailog java class.

**Task 2: Populating both the recycler views:**

Second task was to display all the transaction inserted by the user. Transaction information needed a consistent format for displaying repetitive information. Hence the requirement to design two recycler views. The 1st recycler views show all the daily transaction that are added by the user using the add\_transaction layout. The daily transaction recycler views fetch the information by querying the table for transaction that are added today. The transaction type, category title ana amount are display using a card view. This recycler view is implemented using recyclerview\_categorylist layout, DailyTransactionAdapter for appending the values retired from the database, DailyTransactionItem and DailyTransaction Model for connecting the database data and the displayed data and Dashboard.java class.

The second recycler view shows information about transaction that have been done by the user in past seven days. This recycler view has a similar implementation as the daily transaction recycler view using following components recyclerview\_weeklyreport layout, WeeklyTransactionAdapter, DailyTransactionItem and Dashboard.java class and DailyTransaction Model.

**Task 3: Updating the top cards**

The user needs to view a consolidated information and which is easy access and understanding, which is why I designed the four cards that give this access. These cards calculated using all the tables in the project. The budget is populated from the settings page. The income and expense card are summation of fixed amount table and daily transaction table. The Savings cards is calculated based on the other cards. The calculation for this card is done on the Dashboad.java in a method, so that it can be refreshed every time there is a change done on any of the recycler views or a new transaction is added or the budget is updated.

**In addition:**

In addition to this I’m the group leader. I took the responsibility of arranging the group meetings, consolidation of the project code and reports. I wrote the functions and user manual for the application in the final report.

**INDIVIDUAL REPORT Rebecca Moses D’mello 300322984**

**Task description**

My task was to create a splash screen, Login and Registration Activity, Invitation Page, Signout Activity and Report Generated with bar chart.

Report: Abstract, overview, features and technical challenges (first3 paragraph), incomplete functions, disclaimer, references. The last 2 paragraphs of Technical Challenges were drafted by Alisher Matianiu.

Files created:

1. Splash Screen Activity >> Login Activity >> Registration Activity
2. Database Handler Class
3. Report class inside ui package
4. strings\_settings and edit\_round.xml and button\_rounded.xml files in drawables

Files contributed to:

1. DatabaseHelper.java >> created a table to store budget data, and methods to create/add/update the data.
2. LoginActivity >> included the code to save logged in user data and budget amount.

**Task 1: Splash Screen**

Splash Screen is the first screen which is displayed when the user opens the application. The Screen has an image which is added in drawable folder and there are various attributes set for it like layout\_width and layout\_height is set to match\_parent, contentDescriptions for the imageView is set and retrieved from strings.xml and the most important attribute is srcCompat which helps user access image from the drawables. The SplashScreen.java class contains an onCreate method which allows the screen to be displayed for few milliseconds.

**Task 2: Login/ Registration**

Registration Page involves various components like the editable textbox which has attributes set like hint which is specified from strings.xml, textAlignment to center and id to uniquely identify each component, radio group to cluster radio buttons and has set of attributes used like id and settings the layout\_width and layout\_height to wrap\_content in order to get the radio buttons next to each other. A submit button which adds all the data into the SQLite database and redirects the control to login screen so that user is validated and has access to all other services provided by the application. The Edit Textbox components are customized by creating a file named edit\_round.xml in drawable folder. In the edit\_round.xml file we have specified the color and corner radius of all the edit boxes. This was an efficient way opted in order to specify the attributes once and use them multiple times on different activities. For styling the button same approach was used by creating a button\_rounded.xml file in drawables and specifying attributes for it. In Registration.java class we are using the DatabaseHelper db class which extends the SQLiteOpenHelper. In the DatabaseHelper class we have created table named “CREATE\_USER\_TBL” which has colums to store first name, last name, email, phone, password and gender. All the components are binded and validation have been done inside the onclick method inside java class which displays a customized error displayed on every component. The Activity uses layout constraint which is used to display the components in their proper order when the application runs.

Login Activity allows only registered user to access all the services provided by “Budgy” for their budget management. There are various components displayed like the TextView which retrieves “Welcome” value from strings.xml file in values folder with layout\_width and layout\_height set to wrap\_content, setting textColor and textSize for appropriate display of text. Two Edit TextBoxes are used to take input from user and attributes used for them are background, hint value is retrieved from strings.xml, textAlignment, textSize, drawableLeft to get the customized icons placed on the left corner of edit textbox, id to uniquely identify the component. Login button allows access for the valid user. Text View is displayed on the bottom of the login button which guides the new user to first register into the application. Registration button asks user to fill all the fields. In Login Java class, data has been accessed from the database and validated to check if user is authorized to access the system or not using the getUserByEmailAndPassword method written in DatabaseHelper.java class.

**Task 3: Invite Activity and Signout**

In Invite Activity, there is a button named “Send Invitation” which allows user with the functionality to send the link to people so that they can download budgy and access its features. In Invite class, a static link has been added. Signout function is displayed in the menu bar of invitation activity and the back button is disabled which asks the user to signout from the system and prevent form going backward.

**Task 4: Report Generation of Bar Chart with guidance taken from Jenukshan.**

In the Report generation, we are first adding the categories from the settings and the dashboard. When we add data, the report class displays all the categories done in the past 30 days.