**FINANCIAL MANAGEMENT DOCUMENT**

1. ***APPLICATION CONCEPT:***

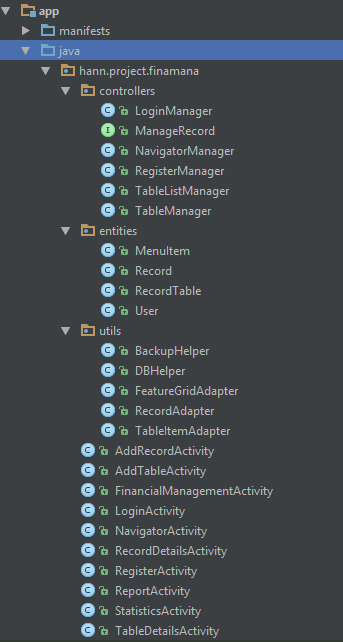
*-* ***Finamana*** *is a* ***mobile*** *application, which help user to manage his Finance.*

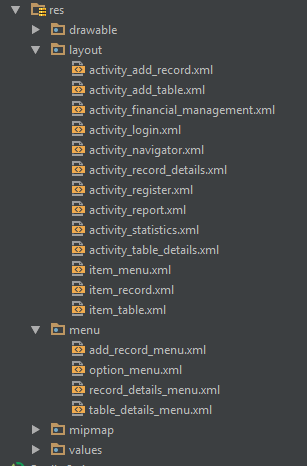
*- Whenever user pay or earn money, a* ***transaction/record*** *is created. The app allows User to record that transaction into its database* ***manually****.*

*- The app manages user’s* ***Records*** *month by month. Every month of a year has only 1* ***Table****, and a Table consists of many* ***Records*** *of that month. This feature of Tables and Records is called* ***Financial Management****.*

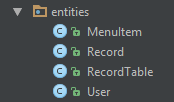
*- The application also provide some feature for user to have an overall look at their transactions, which are* ***Financial Report*** *and* ***Statistics****.*

1. ***APPLICATION STRUCTURE:***

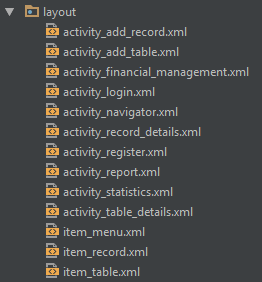
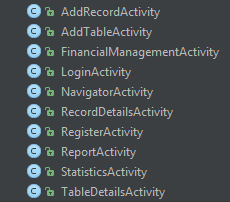
- Finamana is built on the MVC Design Pattern Structure.



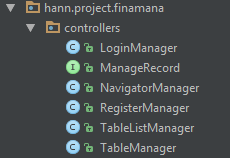
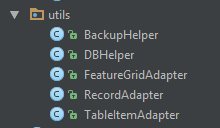
*-* ***Model****: Take responsibility for forming the data model. All the models classes are included in the package* ***hann.project.finamana.entities.***



*-****View****: is responsible for presentation all the needed data. They are xml files, which are included in the* ***res/layout*** *directory. Every activity has its own java class to interact with models and controllers, which are located in package* ***hann.project.finamana****.*



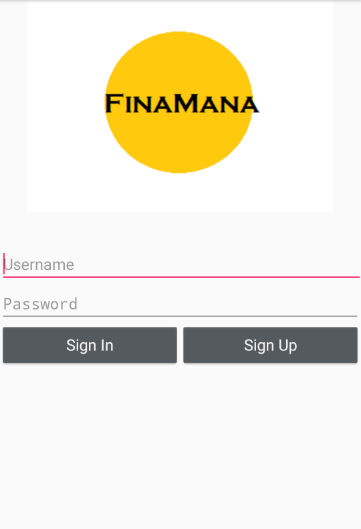
*-****Controller****: Are those classes, which handle the business logic and algorithms. They are located in the package* ***hann.project.finamana.controllers.*** *And there are some class help supporting controllers, they are in* ***hann.project.finamana.utils.***



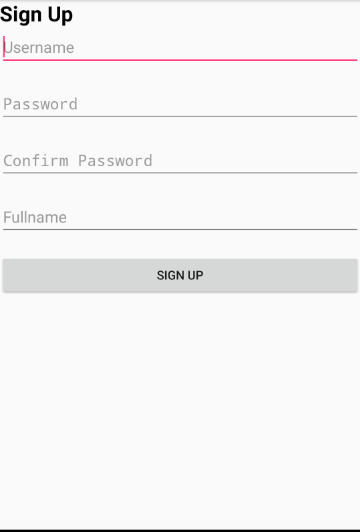
1. ***APPLICATION FEATURES:***

***3.1-Login and Register:***

- The app requires user to **SIGN IN** to use its features. If user does not have account, he can get one by feature **SIGN UP**.



- Click **Sign Up** will get user to the Register page.

Requirement:

- **username** length: 6 ~ 16 characters.

- **password** length: 6 ~ 16 characters.

- **confirm password** must match **password**.

- **Fullname** length under 30 characters, can be null.

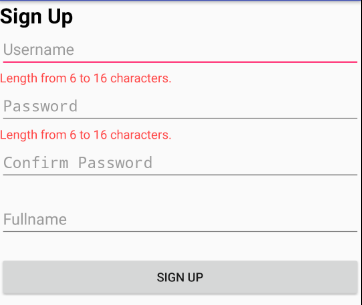
- **username** must not be existed.

*Register Unit test:*

1. **Test Sign up with all blank fields.**

-Parameters: Leave all the fields not inputted.

-Action: Click **Sign Up.**

-Expected:

-Actual: Passed.

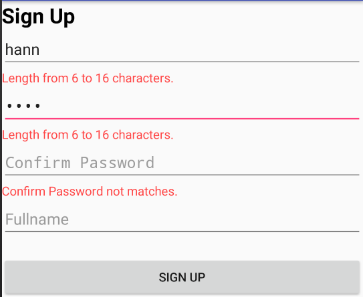
1. **Test Sign up with username and input length < 6.**

-Parameter:

Username: hann

Password: qwer

-Action: Click **Sign Up**.

 -Expected:

-Actual: Passed.

1. **Test Sign up with not match confirm password.**

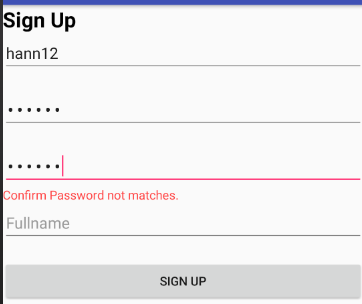
-Parameter:

Username: hann12

Password: qwerty

Confirm password: qwerty

-Action: click **Sign up**

-Expected:

-Actual: Passed.

1. **Test Sign up correctly.**

-Parameter:

Username: hann12

Password: qwerty

Confirm: qwerty

-Action: click **Sign up**

**-**Expected:

Popup: Register Successfully.

Redirect to login page.

-Actual: Passed.

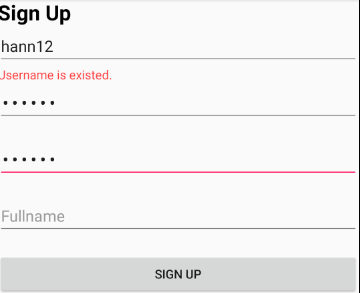
1. **Test Sign up with duplicated user.**

-Parameter:

Username: hann12

Password: qwerty

Confirm: qwerty

-Expected:

-Actual: Passed.

1. **Test Sign up with wrong fullname.**

-Parameter:

Username: hann13

Password: qwerty

Confirm: qwerty

Fullname: London bridge is falling down, falling down, falling down.

-Action: Click **Sign up**

-Expected: 

-Actual: Passed.

*Login Unit test:*

1. **Test Login with wrong username and password.**

Parameters:

Username: (blank)

Password: (blank)

Username: hann12

Password: qqqqqq

-Action: Click **Sign In** button.

-Expected: Pop up “Incorrect username or password”

-Actual: Passed.

1. **Test Login with registered username and password.**

Parameters:

Username: hann12

Password: qwerty

-Action: Click **Sign In** button.

-Expected:

Popup: “LOGIN SUCCESS”

Redirect to **Navigator Screen**.

-Actual: Passed.

***3.2-Navigator Activity:***

- This activity provide user a menu of features that user can interact with.

**+ Financial Management:** Redirect user to Financial Management Activity. Allow user to create Tables and add Records to them.

**+ Financial Report:** Redirect user to Financial Report Activity.Provide user a financial report in current month and all months ever recorded.

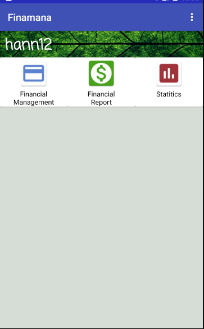
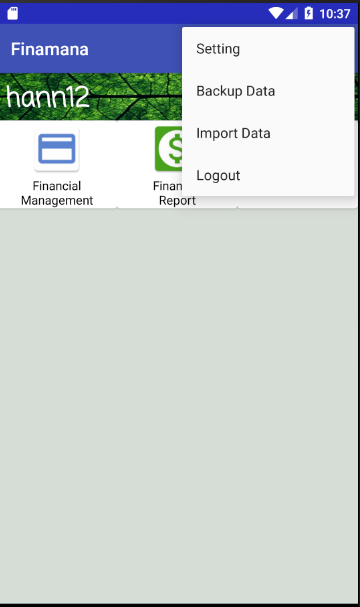
**+ Statistics:** Redirect user to Statistics Activity. Show to user statistical data that can help them have a better financial decision in future.

**+ Setting:** Future feature.

**+ Backup data:** Back up the database file to SD card.

**+ Import data:** Import data from back up file.

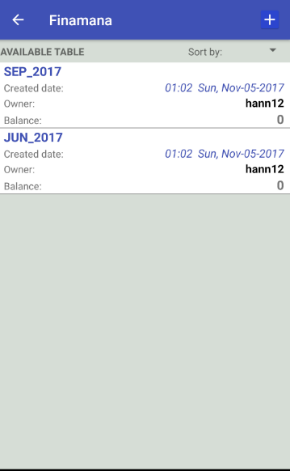
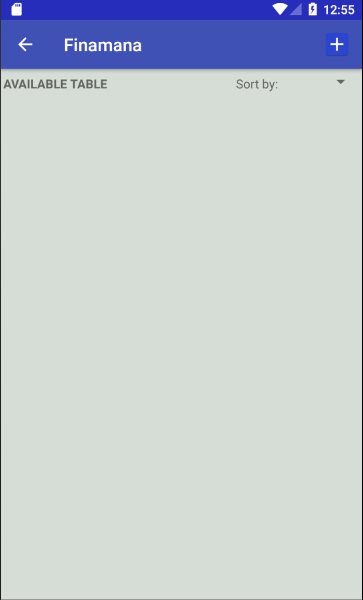
**+ Logout:** Log user out of the app.

- On the left top is user fullname. If fullname is not provided by user, username take that place instead.

***3.3-Financial Management Activity:***

-This activity shows a list of created Table to user. If no table have been created yet, the screen should show empty.

- Newer Table should be placed on top down.

- Features:

+Sort tables: Future feature.

+ **Create a new table**: by clicking on the button  located on the top right of the screen. It allow user to create new table of a specific month.

+ **See Table Details**: by clicking on a Table, User can see all the details of the Table.

+ **Back**: back to the **Navigator Activity**.

- A item in table list:



+Table name: is placed on top left.

***3.3.1-Create a new Table:***

-Create a table of a specific month.

-User only need to choose the year and the month of that year.

-Requirement: Every month of a year has its own Table.



*Create new Table Unit Test:*

1. ***Test Create a new Table****.*

*-Parameters:*

*Year: 2017*

*Month: November*

*-Action: Click button* ***Done***

*-Expected:*

*Popup: “A table has successfully added”.*

*Redirect to Financial Management Activity*



*-Actual: Passed.*

1. ***Test Create existed table.***

*-Requirement: Create a November 2017 Table first.*

*-Parameters:*

*Year: 2017*

*Month: November*

*-Action: Click button* ***Done***

*-Expected:*

*Popup: “Table existed, failed to create.”*

*Table list remains 1 item.*

*-Actual: Passed.*

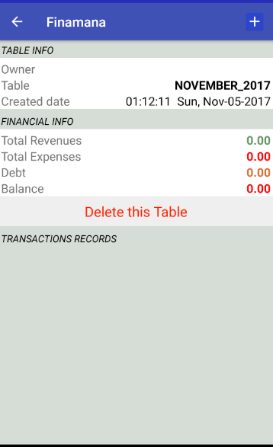
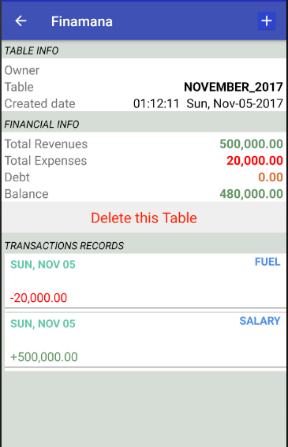
1. ***Test button Back.***

*-Action: Click button* ***Back****.*

*-Expected: Redirect to* ***Navigator Activity****.*

*-Actual: Passed.*

***3.3.2-See Table Details:***

 *-See the details the selected Table.*

*Requirements:*

*+ Fullname is displayed as Owner.*

*+ Table is the combined of Month and Year.*

*+ Created date is recorded when user create the Table.*

*+ Revenues a.k.a Income: The money user earn.*

*+ Expenses a.k.a Outcome: The money user pay.*

*+ Balance = Revenues – Expenses*

*+ If Debt increase, Revenues increase.*

*+ When Debt is paid, Debt decrease and Expenses increase by the same amount.*

*+ If no record is added, Transactions records remain empty.*

*+ Newer record is placed on top-down perspective.*

*-A record:*



*+ Record Date is placed on top left.*

*+ Category of the transaction is placed on top right.*

*+ Below Record Date is Description (can be empty).*

*+ Money Amount:*

*If Revenues: green color with (+)*

*If Expenses: red color with(-)*

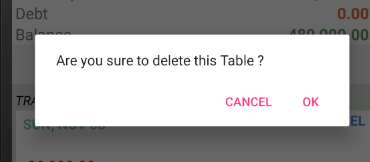
*Because* ***DEBT*** *increase Revenues, so it is considered as an* ***earn category****.*

***DEBT\_PAY*** *increase Expenses, so it is a* ***pay category.***

*-Features:*

*+* ***Add a new Record:*** *by clicking**, Allow user to add a new record to the Table.*

*+ A Dialog appears to make sure that user does not click this button by mistake. By choosing* ***OK****, user allow the app to delete this Table and all the related Records. Choose* ***CANCEL****, to stop the delete activity.*



*+****See the Record Details:*** *by clicking on an record, Allow User to Edit the record.*

*Table Details Unit Test:*

1. *Test button* ***Delete***

*-Actions:*

*Click button* ***Delete*** *on table* ***November\_2017***

*-Expected:*

*If choose* ***Cancel****, dialog disappear, everything remains the same.*

*If choose* ***OK:***

*Redirect user back to Financial Management Activity.*

*The deleted Table disappear on the table list UI.*

*-Actual: Passed.*

*2. Test Delete* ***Table****, also Delete all the related* ***Records****.*

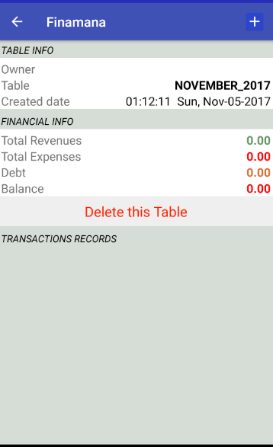
*Because Table and Records data are store in two different table in the database. Records table has a foreign key reference to Table.tableId.*

*-Action:*

***-Delete*** *the table* ***November\_2017*** *with some Records have been added.*

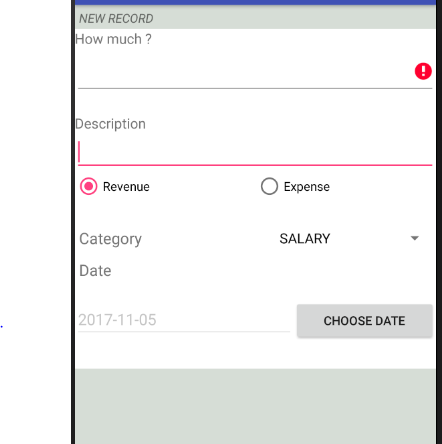
***-****Add a new Table with month of November and Year of 2017.*

***-****Expected:*



-Actual: Passed.

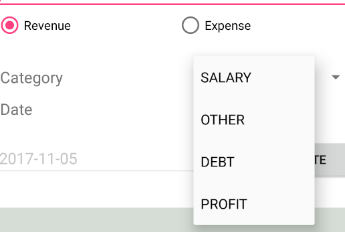
***3.3.2.1-Add a new Record:***

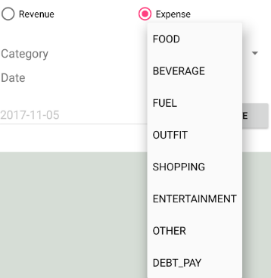
 *-This feature allows user to add a new Record to the Table.*

*Requirement:*

*- Money amount cannot be empty or <= zero.*

*- Description can be empty.*

 *- Category Spinner changes its items depends on Revenue or Expense.*



- Date follow format **yyyy-MM-dd** and must match the Table’s Month and Year.

- Button **Done**  will not redirect to Table Details Activity until those requirements are met. When button **Done** is passed, all the input data will be recorded to the database by a new row.

*Add new Record Unit Test:*

1. *Test Money Amount:*

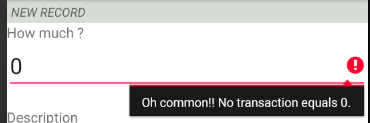
Parameters:

Money amount: “0”

Description: “”

Category: SALARY

Date: 2017-11-05

Expected: 

Actual: Passed.

--------------------------------------------------------

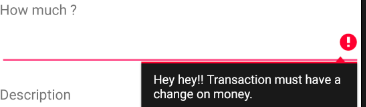
Parameters:

Money amount: “”

Description: “”

Category: SALARY

Date: 2017-11-05

Expected: 

Actual: Passed.

1. Test Date:

Input:

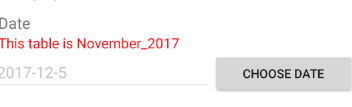
Money amount: **500000**

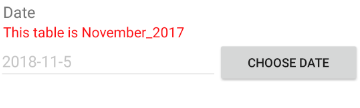
Description: “”

Category: **SALARY**

Date: “**2017-12-5**” and “**2018-11-5**”

Action: Click button **Done**

Expected: 



Actual: Passed.

1. Test Add New record:

Input:

Money amount: **500000**

Description: “**From trading.**”

Category: **PROFIT**

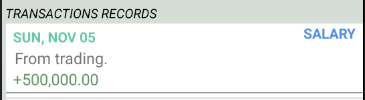
Date: “**2017-11-5”**

Action: Click button **Done.**

Expected:

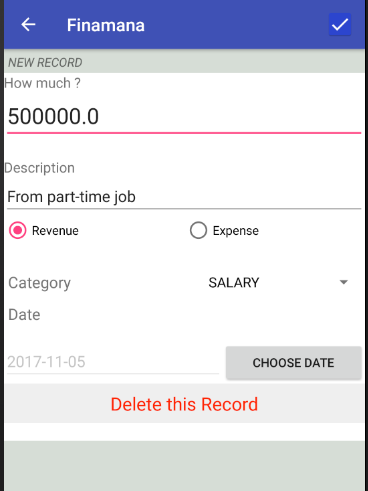
Popup: “Successfully added a Record to the Table”

A new record appears on top of the Transaction list.



Actual: Passed.

***3.3.2.2 See the Record Details:***

- This feature shows a more details version of a Record and allows user to edit it.

Requirement:

- All the data must match the record.

- Everything same to Add new Record (Including Unit Test).

-Feature:

- Button **Delete**: *A Dialog appears to make sure that user does not click this button by mistake. By choosing* ***OK****, user allow the app to delete this Record from the Table. Choose* ***CANCEL****, to stop the delete activity.*

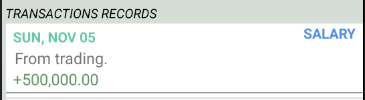
- Button **Save**: *Update the Record with new parameters.*

*See the Record Details Unit Test:*

1. *Test button Delete:*

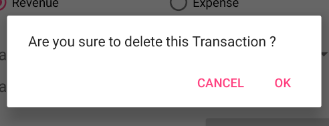
*Actions:*

*- Add a new Record*



*- Click on the record to see details.*

*- Click button Delete.*



*Expected:*

*-If choose Cancel: Back to see details, nothing deleted.*

*-If choose OK:*

*+Redirect to Table Details Activity.*

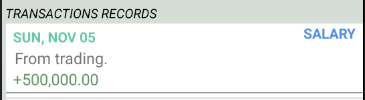
*+The record above disappear from the Transation list.*

*Actual: Passed.*

1. *Test button Save:*

*Actions:*

*-Add a new Record*



*- Click on the record to see the details.*

*- Parameters:*

Money amount: **600000**

Description: “For Traveling**”**

Choose **EXPENSE**

Category: **OUTFIT**

Date: “**2017-11-5”**

**-** click button **Save**.

Expected:

- Transactions count remain the same.

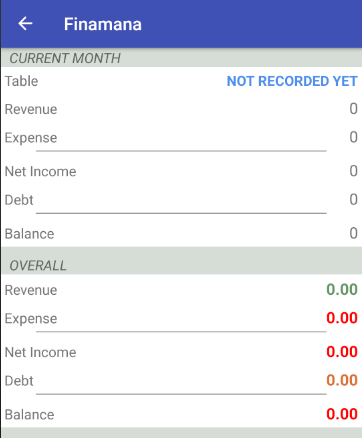
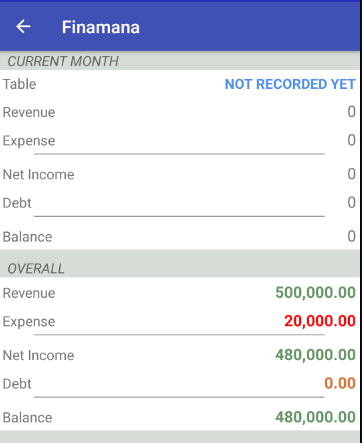
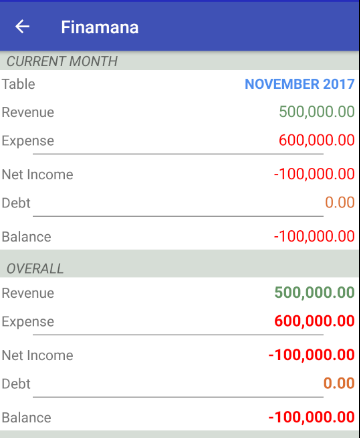
- This appear at the position of the edited record.



Actual: Passed.

***3.2-Financial Report Activity:***

- Show user a summary of their financial activities.

*- Requirement:*

*-* ***Current month*** *shows information of the real time current month Table. If current month Table still not recorded. All information remain zero.*

*-* ***Over all*** *shows the total information of all the tables ever created.*

*- Net Income = Revenue – Expense.*

*- Balance = Net Income – Debt.*

***3.3-Statistics:***

- Show to user statistical data that can help them have a better financial decision in future.

- Categories is sort by how frequent they appeared.

