

# **Requirements and Analysis Document for La-Luna**

Ali Malla, Deaa Khankan, Ali Alkhaled, Bilal Al Malek

date 2021-09-27

version

1

# Innehållsförteckning

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Definitions, acronyms, and abbreviations . . . . .	3
<b>2</b>	<b>Requirements</b>	<b>4</b>
2.1	User Stories . . . . .	4
2.1.1	Smidig navigation på applikationsnivå . . . . .	4
2.1.2	Tilläggnig av nya utgifter . . . . .	4
2.1.3	Lista över tidigare utgifter . . . . .	4
2.1.4	Kvarstående pengar . . . . .	5
2.1.5	Tilläggnig av nya kategorier . . . . .	5
2.1.6	Spendering för aktuell månad . . . . .	5
2.1.7	Statistisk månadspendering . . . . .	5
2.1.8	Begränsning av kategoribudgeter . . . . .	6
2.1.9	Spendering på kategorinivå . . . . .	6
2.1.10	Kategorisida . . . . .	6
2.1.11	Redigering av registrerade utgifter . . . . .	7
2.1.12	Borttagning av registrerade utgifter . . . . .	7
2.2	Definition of Done . . . . .	7
2.3	User interface . . . . .	8
2.3.1	Färg . . . . .	8
2.3.2	Övergripande . . . . .	8
2.3.3	Hemsidan (Home page) . . . . .	9
2.3.4	Analyssidan (Analysis Page) . . . . .	11
2.3.5	Kategorisidan (Categories page) . . . . .	13
2.3.6	Rekommendationer/Hjälp-sidan (QA Page) . . . . .	16
<b>3</b>	<b>Domain model</b>	<b>17</b>
3.1	Class responsibilities . . . . .	17
3.1.1	Category . . . . .	17
3.1.2	Expense . . . . .	17
3.1.3	IDatabaseHandler . . . . .	18
3.1.4	SqliteHandler . . . . .	18
3.1.5	DBHandler . . . . .	18
3.1.6	View klasser . . . . .	18
3.1.7	ViewModel klasser . . . . .	18
<b>4</b>	<b>References</b>	<b>19</b>

# 1 Introduction

La Luna är en budget Applikation som hjälper till användaren att ha koll på hur mycket pengar den spenderar varje månad. Idén går till på att det ska finnas olika kategorier där användaren lägger en gräns på hur mycket hen tänker spendera i en specifik kategori. När man närmar sig att överstiga gränsen får man en typ av pliancy för att dra uppmärksamhet att användaren behöver spendera mindre i denna kategori för att nå spenderingsmål. Defaulta kategorier ska finnas redan när man kör applikationen för första gången men användaren har möjlighet att lägga till sina egna kategorier för att det ska passa sin spenderingsstil.

Applikationen ger användaren möjlighet att ha koll på sin spending i tidigare månader genom en analys som visas på ett modernt sätt som är enkelt att följa. Applikationen är ganska flexibel då man har möjlighet att modifiera kategorier och utgifter när man råkar mata in felaktiga information. Man kan även filtrera utgifter så att bara utgifterna i en specifik kategori visas. Det ska finnas en "sida" i applikationen där man kan få några rekommendationer för att spendera mindre.

## 1.1 Definitions, acronyms, and abbreviations

- Primary key: En kolumn som måste innehåller ett unikt värde på varje rad i en tabell
- Foreign key: En kolumn i en databastabell som pekar mot ett Primary key i en annan tabell
- Activity: Ett fönster som användaren ser i en android applikation.
- Fragment: Ett återanvändbar gränssnitt, man kan skapa den och använda den i olika Activities.
- User Story: En övergripande beskrivning av en funktion i applikationen ur ett användarvänligt perspektiv. Vanligen skriven i ett informellt språk.
- Navigationbar: Ett grafiskt komponent som vanligen placeras i botten av mobilskärmen och innehåller några ikonbeskrivande och/eller textbaserade knappar. Var och en av dessa knappar leder till olika destinationer i applikationen.
- Kategori: En viss gruppering som kan skapas av användaren och som hjälper denne att gruppera in sina betalningar.
- Pliancy: Ett omfattande koncept inom interaktionsdesign och som visar användaren vad hen kan göra i applikationen genom att de olika grafiska komponenten i applikationen bland annat ändrar sina färger om de blir påpekade av musen.
- Visningslista(listview): En vy som grupperar flera identiska element för att sedan visa upp dem i en oftast vertikal skrollbar lista.

## 2 Requirements

### 2.1 User Stories

#### 2.1.1 Smidig navigation på applikationsnivå

##### *Description*

"Som användare vill jag en navigationbar för att navigera enkelt mellan samtliga sidor (Activities, Fragments, etc)"

##### *Confirmation*

##### *Functional:*

- Kan jag gå tillbaka till hemsidan?
- Kan jag se på vilken sida jag befinner mig i (Pliancy)?

##### *Non-functional:*

- Är det smidigt att navigera mellan applikationens sidor?

#### 2.1.2 Tilläggning av nya utgifter

**OBS! ej implementerad än.**

##### *Description*

"Som användare vill jag ha möjlighet att lägga till utgifter själv så att jag har kontroll över min budget samt vilka utgifter som ska ingå i min ekonomiska planering på applikationen."

##### *Confirmation*

##### *Functional:*

- Kan jag lägga till en utgift när som helst?
- Kan jag välja vilken kategori denna utgift kommer att tillhöra?

##### *Non-functional:*

- Kan jag se alla mina utgifter när som helst?

#### 2.1.3 Lista över tidigare utgifter

##### *Description*

"Som användare vill jag kunna ha en lista med tidigare registrerade utgifter så att jag kan veta var jag har spenderat mina pengar."

##### *Confirmation*

##### *Functional:*

- Kan jag öppna en viss registrerad utgift för att få detaljerad information om den?
- Kan jag redigera denna list?
- Kan jag radera en tidigare registrerad utgift från denna lista?

##### *Non-functional:*

- Innehåller denna lista endast utgifter till den aktuella månaden?

-Hur ser denna lista ut när den är tom?

#### **2.1.4 Kvarstående pengar**

*Description*

"Som användare vill jag veta hur mycket pengar som finns kvar av budgeten så att jag kan se hur mycket jag har spenderat i helhet."

*Confirmation*

*Non-functional:*

-Står de kvarstående pengarna tydligt i samtliga sidor där jag förväntas vilja ta reda på dem?

#### **2.1.5 Tilläggning av nya kategorier**

**OBS! ej implementerad än.**

*Description*

"Som användare vill jag kunna lägga till en ny kategori ifall det inte finns en default kategori som överensstämmer med det jag har köpt."

*Confirmation*

*Functional:* -Kan jag lägga till en ny kategori?

-Kan jag välja en budget till den nya kategorin?

*Non-functional:*

-Kan jag när som helst lägga till en ny kategori?

-Finns det en gräns till antal kategorier som jag kan ha/lägga till?

#### **2.1.6 Spendering för aktuell månad**

*Description*

"Som användare vill jag kunna se hur mycket jag har spenderat inom varje kategori under månaden så att jag kan ha koll på hur min spendering går till."

*Confirmation*

*Functional:*

-Kan jag se hur mycket jag har spenderat inom varje kategori (månadsvis)?

-Kan jag veta hur mycket budget som är kvar för varje kategori?

*Non-functional:*

-Kan jag se hur mycket budget som är kvar för varje kategori när som helst ?

#### **2.1.7 Statistisk månadspendering**

*Description*

"Jag vill kunna se hur mycket jag har spenderat inom varje kategori från tidigare månader så att jag som användare ska ha koll på hur mina utgifter varjerar under året"

*Confirmation*

*Functional:* -Kan jag klicka på Analys i navigationsbar?

-Kan jag se en lista med alla kategorier i form av en grid (Grid of Equals)?

-Kan jag se budgetten på varje kategori?

-Kan jag se hur mycket jag har spenderat inom varje kategori (årligen)?

### **2.1.8 Begränsning av kategoribudgeter**

***OBS! ej implementerad än.***

*Description*

"Som användare vill jag kunna begränsa budgeten av varje kategori för att det ska bli lättare att kontrollera hur mycket pengar jag tänker att spendera på respektive kategori."

*Confirmation*

*Functional:*

-Kan jag välja vilket belopp som helst?

*Non-functional:*

-Är det möjligt att ändra på budgeten när som helst?

### **2.1.9 Spendering på kategorinivå**

***OBS! ej implementerad än.***

*Description*

"Som användare vill jag kunna se en lista med de senaste registrerade betalningar som tillhör en viss kategori så att jag kan lätt komma till mina betalningar inom den och ändra dem."

*Confirmation*

*Functional:*

-Får jag en ny sida som visar en lista med betalningar om jag klickar på en kategori?

-Kan jag klicka på en betalning och ändra den?

-Kan jag gå tillbaka till alla kategorier?

*Non-functional:*

-Kan jag öppna de senaste betalningarna inom en kategori när som helst?

### **2.1.10 Kategorisida**

***OBS! ej implementerad än.***

*Description*

"Som användare vill jag ha en sida till kategorier så att jag kan modifiera dem på ett smidigt sätt."

*Confirmation*

*Functional:*

- Kan jag klicka på kategorier i navigationsbar?
- Kan jag se en lista med samtliga kategorier när jag klickar i navigationsbaren?
- Kan jag se relevant information om varje kategori (budget, namn, osv.....)
- Kan jag klicka på "Edit" och ändra kategorins information?

### **2.1.11 Redigering av registrerade utgifter**

***OBS! ej implementerad än.***

*Description*

"Som användare vill jag kunna redigera en utgifts information ifall jag har råkat skriva felaktig information t.ex. (pengar, kategori, etc)"

*Confirmation*

*Functional:*

- Kan jag klicka på en utgift och få en sida med ändringar?
- Kan jag klicka på spara ändringar och se att ändringarna sparades?
- Kan jag klicka avbryt och gå tillbaka utan att ändra?

*Non-functional:*

- Funkar det närsomhelst så länge jag har utgifter?
- Kan jag gå tillbaka utan att ändra i fall jag klickade fel?

### **2.1.12 Borttagning av registrerade utgifter**

***OBS! ej implementerad än.***

*Description*

"Som användare vill jag kunna ta bort en registrerad utgift för att det finns en risk att jag bland annat råkar skriva in felaktig information."

*Confirmation*

*Functional:*

- Kan jag radera en utgift?

*Non-functional:*

- Kan jag ta bort tidigare tillagda utgifter när som helst?

## **2.2 Definition of Done**

För att en användarberättelse, User story, ska definieras som fullständigt kompletterad ska följande krav ha uppfyllts:

### *Kompletterade uppdrag*

Samtliga uppdrag inom en användarberättelse ska vara färdigimplementerade.

### *Exekverbar kod*

Det måste finnas en fungerande kod som kompileras och exekveras utan några tekniska problem.

### *Testning*

Testning inledas med tillräckligt många visuella exempel för att sedan en regelbunden testning ska påbörjas, så som enhetstest (Unit test). Denna fas är väldigt viktig inom mjukvaruutveckling eftersom den bekräftar att koden beter sig som förväntat, vilket i sin tur leder till hög mjukvarukvalitet.

## 2.3 User interface

### 2.3.1 Färg

Applikationen har en enhetlig visuell struktur där alla sidorna inte skiljer sig mycket från varandra. Färgerna som används i applikationen är färger som gör applikationens utseende något sportig, vilket ger applikationen en vacker utsikt och större önskan att användas av yngre målgrupp. De mest använda färgerna i applikationen börjar från vänster på färgpaletten vilket åskådliggörs i Figur 1.

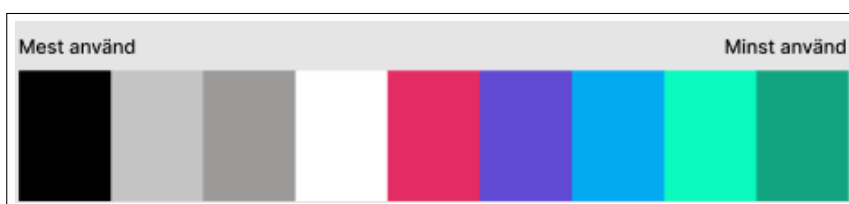


Figure 1: Färgerna som används genomgående i applikationen.

### 2.3.2 Övergripande

Applikationen består av fyra huvudsidor som användaren kan navigera mellan dem när som helst med hjälp av den globala menyn längst ner på skärmen, se figur 2.

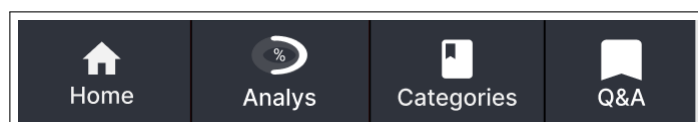


Figure 2: Den globala menyn.



Denna globala meny innehåller en pliancy som reagerar med användarens reaktion. Till exempel, om användaren klickar på hemsidan (Home), ändras hemsidans bakgrund i den globala menyn till ljusgrått. Därtill är interaktiva komponenter förtydliga med hjälp av ikoner. Denna meny är designat för att göra navigeringen mellan sidorna enklare och applikationen mer överskådligt.

### 2.3.3 Hemsidan (Home page)

Hemsidan är den första sidan som dyker upp när användaren öppnar applikationen varje gång. Den består av en cirkel som visar användaren spenderade beloppet från budgeten och justerar dess färg enligt förbrukade pengar. Dessutom finns det text längst upp till vänster på skärmen som visar användaren hela budgeten, se figur 3.

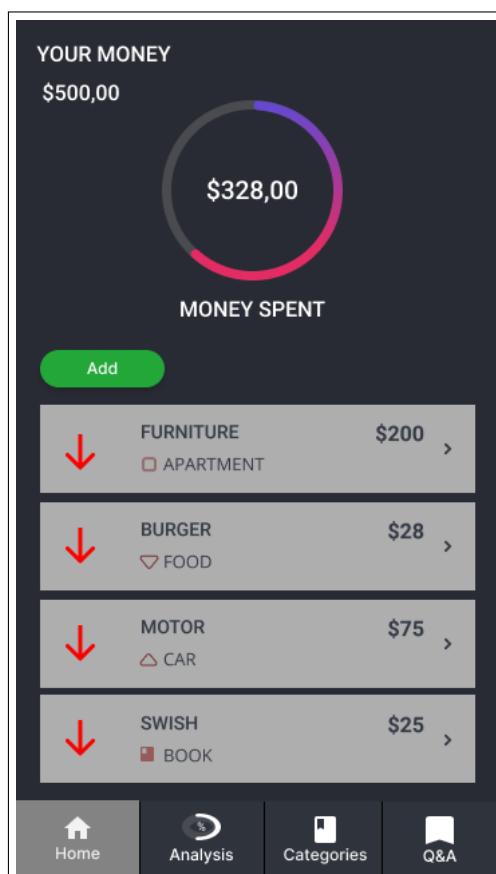


Figure 3: Hemsidan.

I den nedre halvan av gränssnittet ligger en rullningsbar visningslista (listview) som visar användaren alla utgifter som har lagts till. Utgifterna placeras kronologiskt efter

datum i visningslistan, dvs. senast först.

Därtill finns det också en grön knapp så att användaren kan lägga till sina utgifter. Genom att trycka på knappen visas ett nytt gränssnitt där användaren måste ange information om den utgiften som ska läggas till, vilket åskådliggörs i Figur 4.

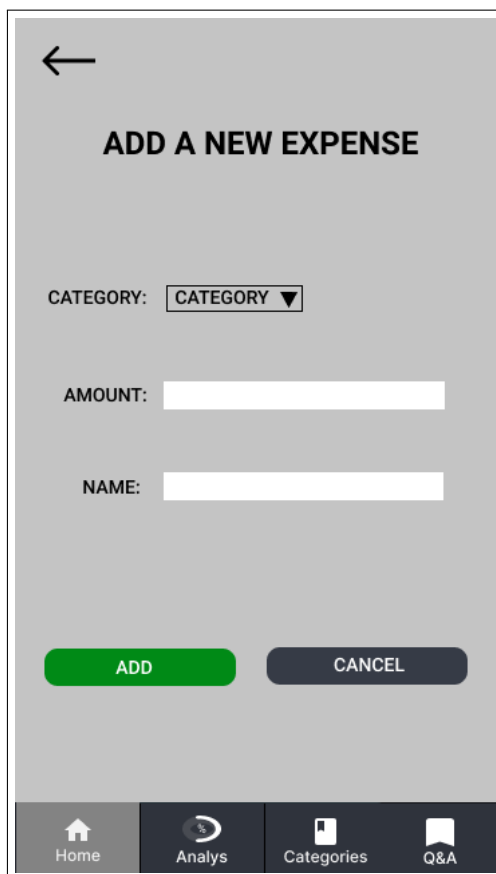


Figure 4: Lägga till en utgift sidan.

Dessutom har användaren också möjlighet att radera eller redigera en utgift genom att klicka på utgiften i visningslistan. Detta öppnar ett nytt gränssnitt med detaljerad information om utgiften och två knappar, en för radering och en för redigering. Genom att klicka på redigeringsknappen öppnas en ny sida där användaren kan redigera och spara redigeringen. För att radera behövs användaren bara klicka på ta bort knappen, se figur 5.

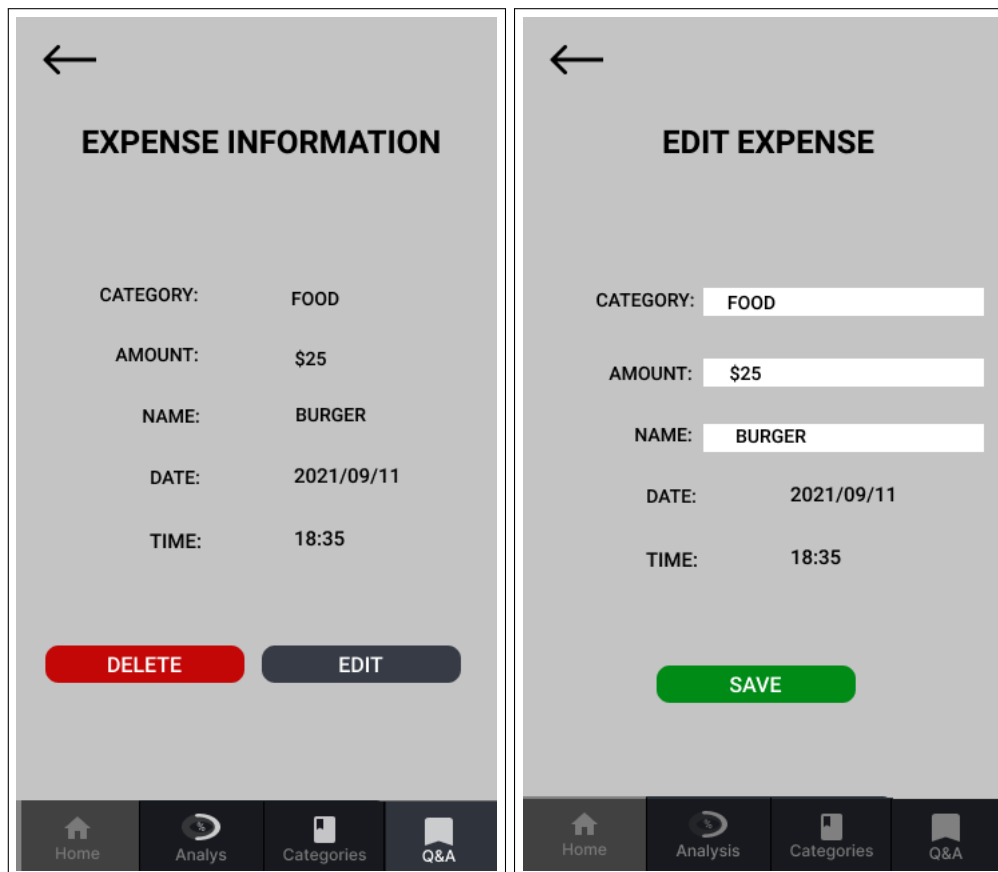


Figure 5: Två sidor som beskriver processen att radera eller redigera en utgift.

Alla sidorna som öppnas från hemsidan har möjlighet att gå tillbaka till föregående sida genom pilen längst upp till vänster som visas i figur 4 och 5.

### 2.3.4 Analyssidan (Analysis Page)

I den övre halvan av gränssnittet finns samma komponenter som i den övre halvan av hemsidan. Ett litet tillägg som finns i analyssidan är två små pilar för att ta användaren till föregående analysidor. Utöver detta finns det små cirkelar som liknar den stora cirkeln i toppen, men istället för att visa användaren hur mycket av hela budgeten som har spenderats, visas bara hur mycket som har spenderats av budgeten för varje kategori, se figur 6.

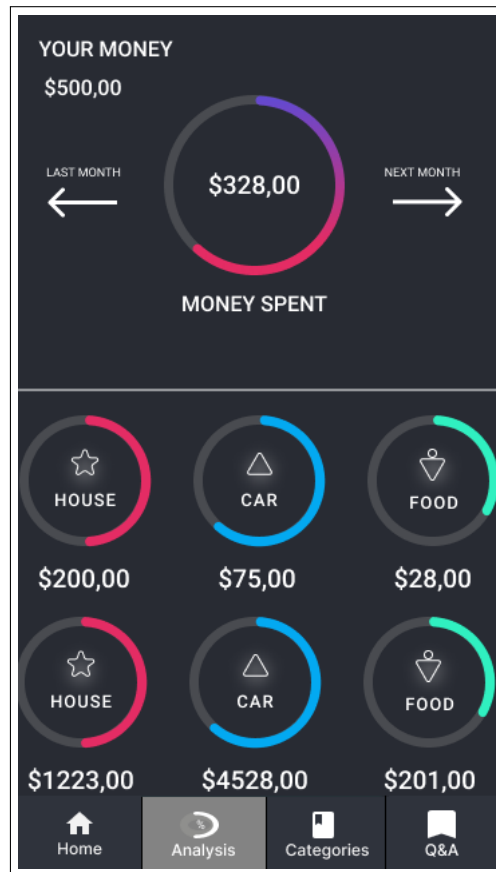


Figure 6: Analyssidan.

Genom att klicka på en cirkel öppnas ett nytt gränssnitt där visas en lista över specialutgifter som tidigare lagts till för den valda kategorin. Till exempel, om användaren bara vill se utgifterna för Matkategorin "Food Category", kan hen klicka i den lilla gröna cirkeln, för klargörande se figur 7.

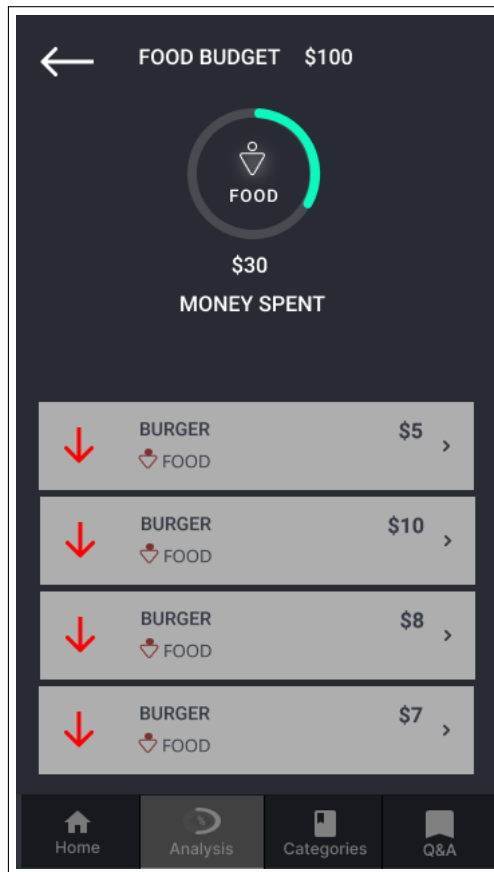


Figure 7: En lista av utgifter som tillhör till Mat kategori.

### 2.3.5 Kategorisidan (Categories page)

Kategorisidan innehåller kategorierna som nästan täcker hela gränssnittet i form av en rullningsbar visningslista. Dessa kategorier är standardkategorier och kategorier som användaren tidigare har lagt till. Dessutom finns det en knapp längst upp till höger för att lägga till kategorier vilket åskådliggörs i Figur 8.

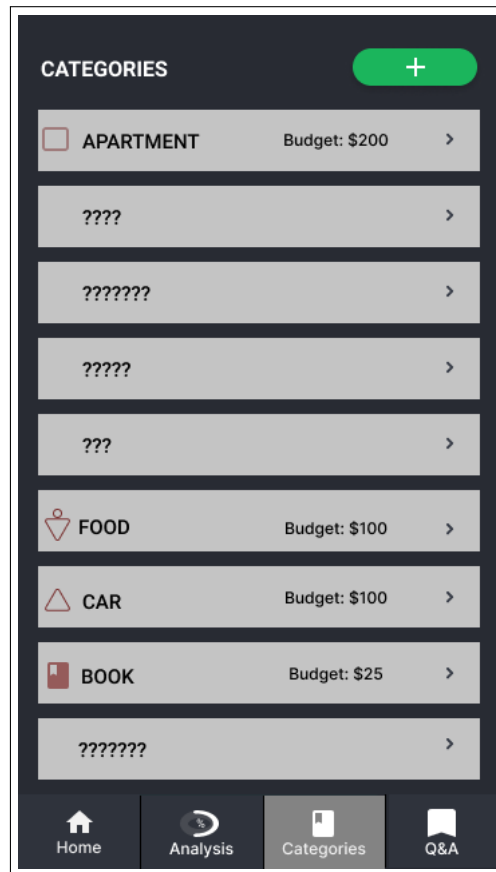


Figure 8: Kategorisidan.

Processen för att lägga till en ny kategori, läsa kategorisinformation, radera eller redigera en kategori, fungerar på liknande sätt som för utgifter vilka ligger under rubriken hemsidan. Men för mer tydlighet se figur 9 och 10.

A mobile app screen titled "ADD A NEW CATEGORY" with a back arrow in the top left. It features two input fields: "BUDGET:" and "NAME:". At the bottom, there are two buttons: a green "ADD" button and a dark grey "CANCEL" button. A bottom navigation bar contains four icons: Home, Analys, Categories (highlighted), and Q&A.

Figure 9: Processen för att lägga till en ny kategori.

Two side-by-side mobile app screens. The left screen is titled "CATEGORY INFORMATION" and shows "CATEGORY NAME: FOOD" and "BUDGET: \$25". It has a red "DELETE" button and a dark grey "EDIT" button. The right screen is titled "EDIT CATEGORY" and shows "CATEGORY NAME: FOOD" and "BUDGET: \$25" in input fields, with a green "SAVE" button below. Both screens have a bottom navigation bar with icons for Home, Analys, Categories (highlighted), and Q&A.

Figure 10: processen för att läsa information om en specifik kategori samt radera eller redigera en kategori.

### 2.3.6 Rekommendationer/Hjälp-sidan (QA Page)

Denna sida består bara av en dragspel-menü (accordion) som täcker hela gränssnittet. I dragspel-menyn innehåller frågor som användaren sannolikt kommer att stöta på när de använder applikationen, se figur 11.



Figure 11: Rekommendation/hjälp-sidan.



### 3 Domain model

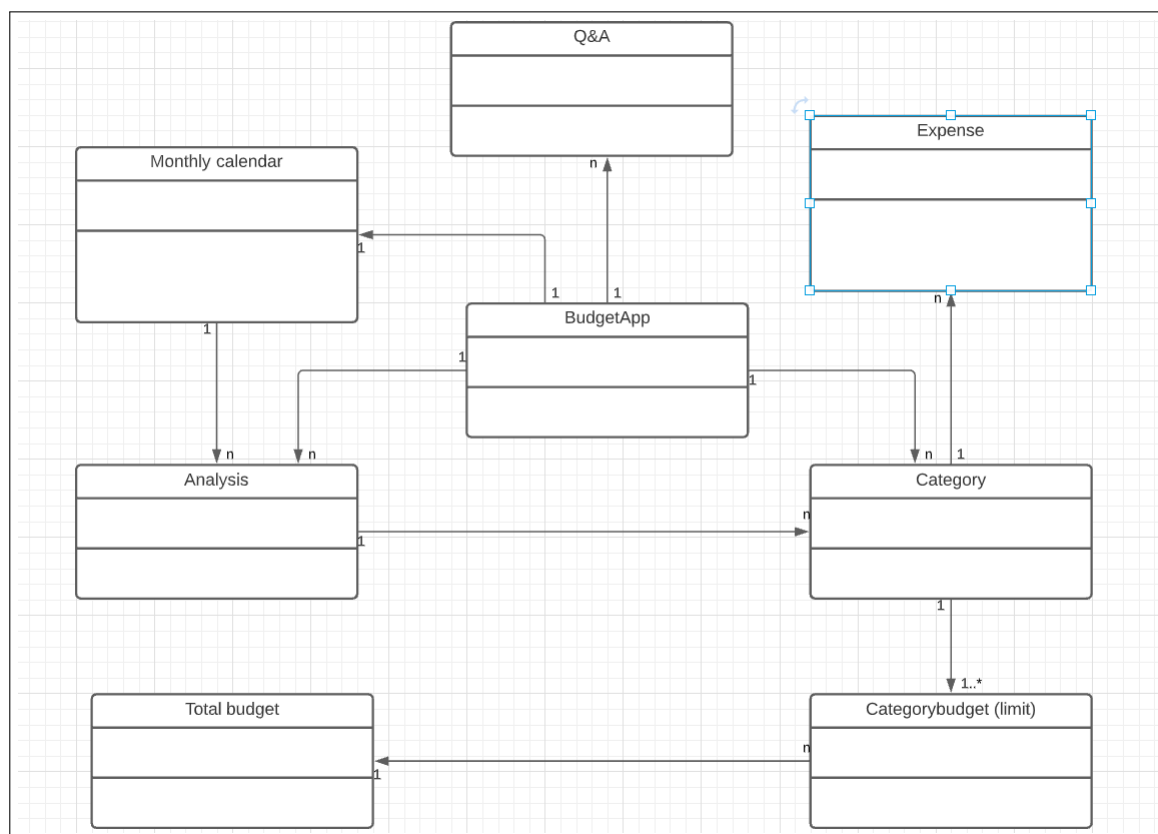


Figure 12: Domain model.

#### 3.1 Class responsibilities

Här är en lista med de klasser som ska användas i applikationen och deras ansvar. De sista två punkter är en beskrivning av en samling klasser som uppfyller liknande funktioner.

##### 3.1.1 Category

klassen representerar en kategori i databasen och den innehåller allt info som en kategori ska ha i form av attributes till exempel (id, namn, gräns, startdatum, slutdatum). Varje objekt av Category klassen ska representera en rad i tabllen Categories i databasen och den ska användas av applikationens gränssnitt.

##### 3.1.2 Expense

klassen representerar en utgift, samma som kategori klassen den innehåller info om utgiften (namn, värde, osv) och varje objekt av den ska representera en rad i tabellen

Expenses. Databastabellen som klassen representerar innehåller en (Foreign key) så klassen innehåller en attribute av typen Category som pekar mot ett kategori., Den kommer att användas i applikationens gränssnitt också.

### **3.1.3 IDatabaseHandler**

En interface, varje klass som implementerar av denna interface ska vara ansvarig för hantering av databasen alltså den ska innehålla metoder som är ansvariga för hämtning och inmatning av information från databasen.

### **3.1.4 SqliteHandler**

Klassen är ansvarig för hanteringen av sqlite databasen som ska finnas i mobilen. Den implementerar interfacet IDatabaseHandler och innehåller metoder för hämtning och inmatning av information från databasen. Den innehåller metoder för hämtning av data till exempel getCategories som ger tillbaka alla kategorier i databasen och metoder för att mata in värde i databasen till exempel addExpense som lägger till en utgift i databasen. Klassen är också ansvarig för att skapa databasen och skapa tabellerna som finns i den eftersom den ärver från abstrakt klassen SQLiteOpenHelper som är uppbyggd inom android. Databasen skapas i konstruktorn och tabellerna skapas i metoden onCreate.

### **3.1.5 DBHandler**

Klassen är ansvarig för all kommunikation mellan databasen och resten av programmet. Den innehåller ett objekt som har den dynamiska datatypen SqliteHandler och den statiska data typen IDatabaseHandler. Den använder sig av denna objekt för hanteringen av databasen och den underlättar för bytet av databasen också och gör att resten av programmet är oberoende av vilken databas som användas i bakgrunden.

### **3.1.6 View klasser**

Design mönstret MVVM kommer att användas för att bygga applikationen så varje Activity och Fragment som ska finnas i applikationen kommer att innehålla en Java klass som är ansvarig för att hantera gränssnittet

### **3.1.7 ViewModel klasser**

Det behövs viewmodel klasser i mvvm design pattern som kommer att fungera som en länk mellan model klasser (Category, expense, DBHandler) och view klasser. ViewModel klasser kommer att ärva från BaseObservable klass som är uppbyggd inom android och fungerar som hjälpmedel för data utbyte mellan view klasser och viewmodel klasser

## 4 References

List all references to external tools, platforms, libraries, papers, etc.