Innehåll

Version: 2.0

Date: 29-05-16

Author: Group 10

1 Introduction

Learning is one of the most valuable things you can do in the world. Without it we wouldn't be able to become engineers, doctors or even be able to speak different languages with each other. In modern civilisation things develop quickly and everything changes but one thing that is consistent is learning. We will always have to learn to live. Life is about learning. That is why our project is main focused in learning. We will develop an application for android that will help you study and therefore learn. It will be a flashcard application because that is one of the easiest way to learn small facts.

1.1 Purpose of application

Learning is an integral part of a lot of people's lives so we should strive to make learning as effective as possible. As there is a lot of downtime in most people's lives now, whether it's waiting for the bus or waiting for the TV commercials, a lot of time that could be spent in a better way is wasted. The goal of this application is to make it possible for users to study with the help of flashcards in their mobile device whenever they want, including those moments of downtime.

1.2 General characteristics of application

The application will consist of several decks, each of them holding a certain amount of flashcards with a question on the front and its respective answer on the back. If the user decides to open a deck the program will open a view which presents an overview of the deck and if the user presses the "Play" button a card is randomly chosen from within the deck and the card's question is presented to the user. If the user swipes or presses a button the card will flip and show the answer. The user will then have the choice of assigning a difficulty to the card which will influence how often the card is presented to the user. If the user swipes or presses the button again the card will once again flip but at this point it will move on to another card. This process repeats until a certain amount of cards have been presented. Once finished the user is returned to the main menu.

1.3 Scope of application

The goal is to have the application running on all the most common platforms. The gist of the idea is that a desktop version of the application can be used to easily create decks and a multitude of cards with different properties and then with the mobile version so that users can handily rehearse their cards wherever they are. It would also be possible practice on the desktop version or add cards on the mobile version even if usability might be slightly hampered. Due to time constraints development will be exclusively for the Android version.

1.4 Objectives and success criteria of the project

1. The user should be able to create a deck and add cards to it
2. They should also be able to add audio and images to the cards
3. The user should be able to open a deck and go through their cards one by one until the end of the deck
4. The user should be able to share decks with other users through the internet.

1.5 Definitions, acronyms and abbreviations

- **GUI**, short for Graphical User Interface
- A **deck** is a collection of flashcards.
- A **flashcard** is a card with a question on the front, and its respective answer on the back.
- **SQLite** is a database management system used to store and retrieve data in a database.
- **JSON** is a standardized representation of a JavaScript object.

2 Requirements

2.1 Functional requirements

1. The user should be able to install the application on their chosen environment and be able to quickly get started on building a deck or downloading another user's deck.
2. The user should be able to create or download another person's deck.
3. The user should be able to open a deck and go through its cards.
4. The user should be able to share their deck with other users.

2.2 Non-functional requirements

2.2.1 Usability

Since this is an application that is supposed to be used regularly and sometimes for short durations it is important that the UI is easy to understand and quick to navigate. Because of this some time will be spent on the UI but it won't be the main focus due to time constraints. There will also be some sort of tutorial to get users started with the application in case they do not understand the flow intuitively.

2.2.2 Reliability

N/A

2.2.3 Performance

As this application is not very computationally complex or dependent on stable network connections, for the most part, performance should not be an issue. Of course this does not mean we should just ignore all optimization as, especially for mobile systems, the less energy spent is better.

The only thing that could cause performance issues would be the database. Since the data from the database is read in a synchronous fashion on start-up the application would appear to have frozen as it is reading the data if the database is too large.
As it is unlikely that any user's database would become so exceedingly large to the point that it takes several seconds to read all the data no precautions were taken against this. If this did end up a real problem for some users it would be easy to implement an asynchronous method to read the database.

2.2.4 Supportability

As previously stated, the plan is to in the future have support for all the major environments, most notably Windows and OS X for desktop and Android and iOS for mobile. There would also be a website where users can upload their decks and share them with users all over the world. Users would then be able to rate the decks that the other users have uploaded.

2.2.5 Implementation

The focus lies mainly on the Android version since the idea was that the mobile versions would be used the most and due to the fact that Android uses Java unlike iOS. The only requirement for the application to run is using Android whose firmware is over 4.0.3. An internet connection is also required so that the application can be downloaded from the Play Store.

2.2.6 Packaging and installation

The installation will be simple and quick to install through Google's Play Store.

2.2.7 Legal

The user has to follow the Terms of Service provided by Google and us.

2.3 Application models

2.3.1 Use case model

See APPENDIX for UML diagram and textual descriptions.

2.3.2 Use cases priority

1. Create a deck
2. Create a card in a deck
3. Start a deck
4. Play through a deck
5. Remove a card from a deck
6. Edit a card in a deck
7. Share a deck
8. Statistics

2.3.3 Domain model

We are well aware that the statistics are separated from the rest of the model. We simply did not have time to get this fixed because the members of our team worked on different modules.

See APPENDIX for UML diagram

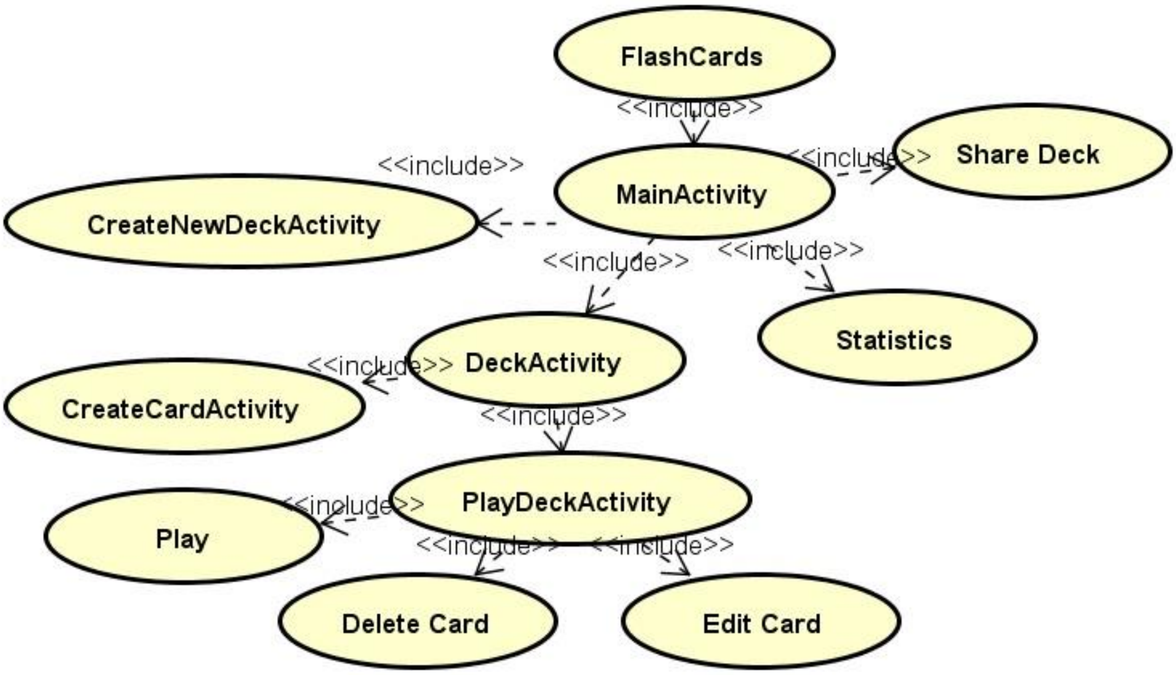
2.3.4 User interface

As the application is developed for Android, which does most of the resizing by itself, there was no reason to think about the resolution of the application. Android even uses different resolutions on any eventual images as long as they are provided and marked with which resolution they are supposed to be used with.

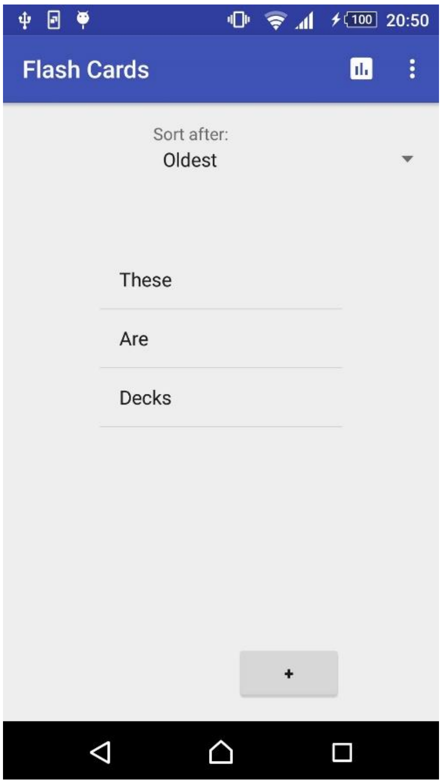
2.4 References

APPENDIX

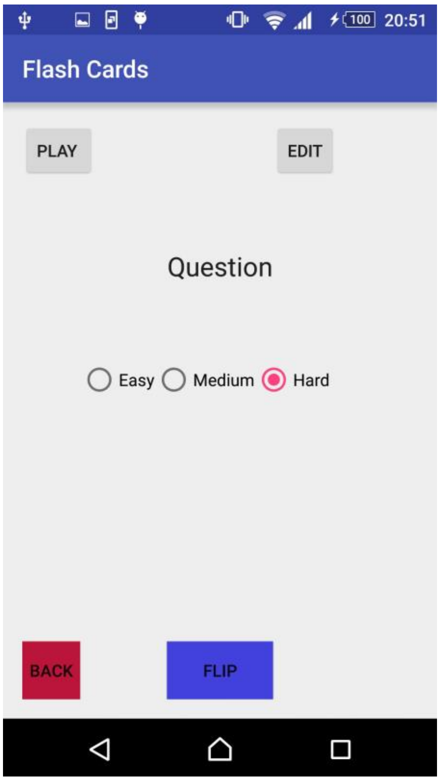
Use case model



GUI

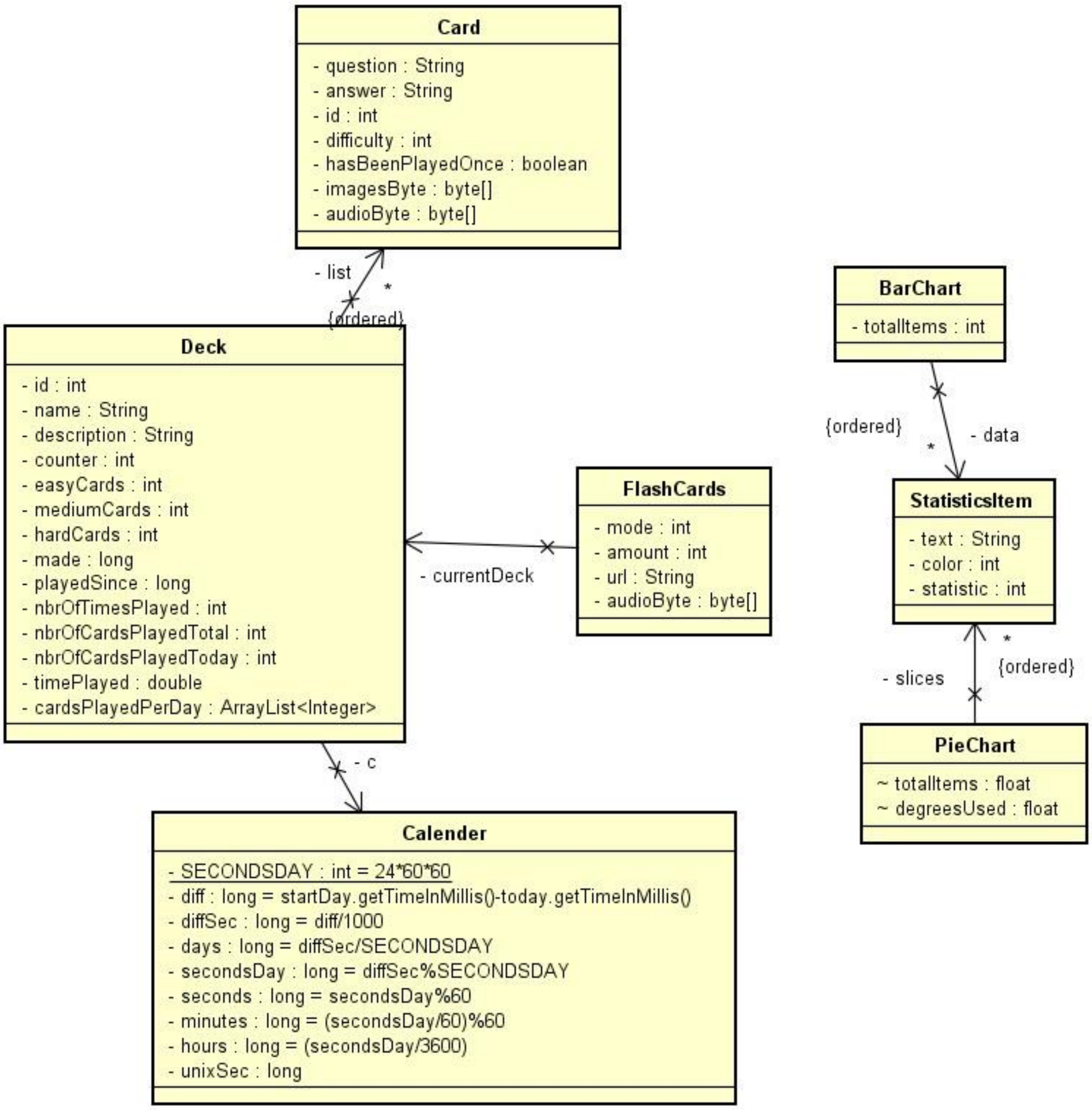


Sort after:
Oldest



○ Easy ○ Medium ● Hard

Domain model


Card
- question : String
- answer : String
- id : int
- difficulty : int
- hasBeenPlayedOnce : boolean
- imagesByte : byte[]
- audioByte : byte[]
- list
*
{ordered}
Deck
- id : int
- name : String
- description : String
- counter : int
- easyCards : int
- mediumCards : int
- hardCards : int
- made : long
- playedSince : long
- nbrOfTimesPlayed : int
- nbrOfCardsPlayedTotal : int
- nbrOfCardsPlayedToday : int
- timePlayed : double
- cardsPlayedPerDay : ArrayList<Integer>
- currentDeck
FlashCards
- mode : int
- amount : int
- url : String
- audioByte : byte[]
- c
Calender
- SECONDSDAY : int = 24*60*60
- diff : long = startDay.getTimeInMillis()-today.getTimeInMillis()
- diffSec : long = diff/1000
- days : long = diffSec/SECONDSDAY
- secondsDay : long = diffSec%SECONDSDAY
- seconds : long = secondsDay%60
- minutes : long = (secondsDay/60)%60
- hours : long = (secondsDay/3600)
- unixSec : long
BarChart
- totalItems : int
{ordered}
*
- data
StatisticsItem
- text : String
- color : int
- statistic : int
*
{ordered}
- slices
PieChart
~ totalItems : float
~ degreesUsed : float