Chalmers University of Technology

OOP Projekt

System design document (SDD)

Rent A Car - Stulb

Albert W, Hannes T, Jamal M, Johan S, Josef N
24/10-2021
v.2.0

---

1. Introduction

Car Rental Application is based on a concept to book a car and taxi online. Here at first the user has to login or sign up to get access. Then the user can list, search for cars according to their needs, check each car's description, book prices and book easily with help of various payment methods. All the available cars can be rated by the users too. The user can filter, search or sort cars by price for searching cars.

The application also displays a reserved car's list. With the help of this application, online car booking has become easier for the customers. It can be used in several android gadgets such as smartphones, tablets, television. This project is easy to operate and understood by the user.

1.1 Design goals

The goal is to have a loosely coupled and modular application that is easy to extend.It should be simple to add new accounts. Android is an open source so that developers find it easy to establish and expand new features.

1.2 Definitions, acronyms, and abbreviations

GUI ”Graphical User Interface” also ”User Interface”, the part of the program that the user sees and interacts with.

User Story short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system.

API ”Application Programming Interface”, an interface or communication proto-col between a client and a server intended to simplify the building of client-side software.

Hi-Fi ”High fidelity”, a sketch with much detail.

UML ”Unified Modeling Language”, A visual language for mapping and repre-senting systems. Can be used for modelling och class diagrams, sequence diagrams and more. In this project used for Domain and design model.

MVC “Model, View and Controller”

NoSQL NoSQL originally referring to "non-SQL" or "non-relational” [1]. It is a database which provides a mechanism for data storage and retrieval of data. That is not modeled with tabular relations used in relational databases.

JSON (JavaScript Object Notation). Is a data-interchange format used because it is both easy for humans to read and write and furthermore machines can easily parse and generate it.

Firebase Realtime Database The Firebase Realtime Database is a cloud-hosted NoSQL database that lets you store and sync data between your users in real time.

2. System architecture

The overall structure of the application is following the MVC-pattern. It requires only one android phone or android emulator on a computer to function. The MVC-pattern is a software design pattern commonly used when designing GUIs. The pattern is divided into three different parts, model, view and controller. The first part is the model that is responsible for the datan and the logic and it should not know anything about the part who is not a part of the model . The second part is the view, the view is responsible for the graphics of the software. Finally is the controller which is responsible for user interactions, processing events that can result in changes in both the view and the model. The software application starts by building the gradle files and then running the executable file and stops by closing the emulator.

2.1 Model

The Model Package holds classes representing data as well as the core logic of the program. The model handles data about the cars stored in the database. It also handles dates for the reservations and the user's payment details.

2.2 View

The view package holds classes that are used for the visual representation of elements from the model package and takes use of the library **.xml** in order to accomplish this.
The view is together with the controller responsible for rendering the user interface. Since we are using xml files.

2.3 Controller

The controller package holds classes and interfaces that are used to manage views in the view package and to handle user input such as key presses and then it passes the action on to the model who then manages the action.
The controller package is divided into three different packages, home, browse and profile.

2.4 Singleton-pattern

Singleton-pattern is used in the Database class. It ensures that there is only one one instance of the class and it also provides a global access point to the only instance.

2.4 MVC-pattern

Model View Controller-pattern the system architecture follows a MVC-pattern. The application is divided into three different packages, the model-package, the view-package and the controller-package.

2.5 resources

The resource package holds resources or data that can be accessed by the code. Images, icons, fonts and **XML** files can all be found in the resources package.

2.6 Communication at a high level

When the program is started an asynchronous Firebase query is sent to retrieve the data from the Firebase database. When the query is finished the program will load the database into its memory. As this is happening the program is launching the views and view controllers to display the user interface. When a view requests information from the database the model is called from a controller and the Database class that exists in the application memory hands the controller the information which in turn hands the information over to the view. When the application is closed the Database class sends the information to the Firebase database and the updated information is saved.

3. System design

3.1 Model View Controller

Since the program has a graphical user interface representation, a **MVC** pattern was used to separate the different components of the program. The program is based on **XML** graphical components.

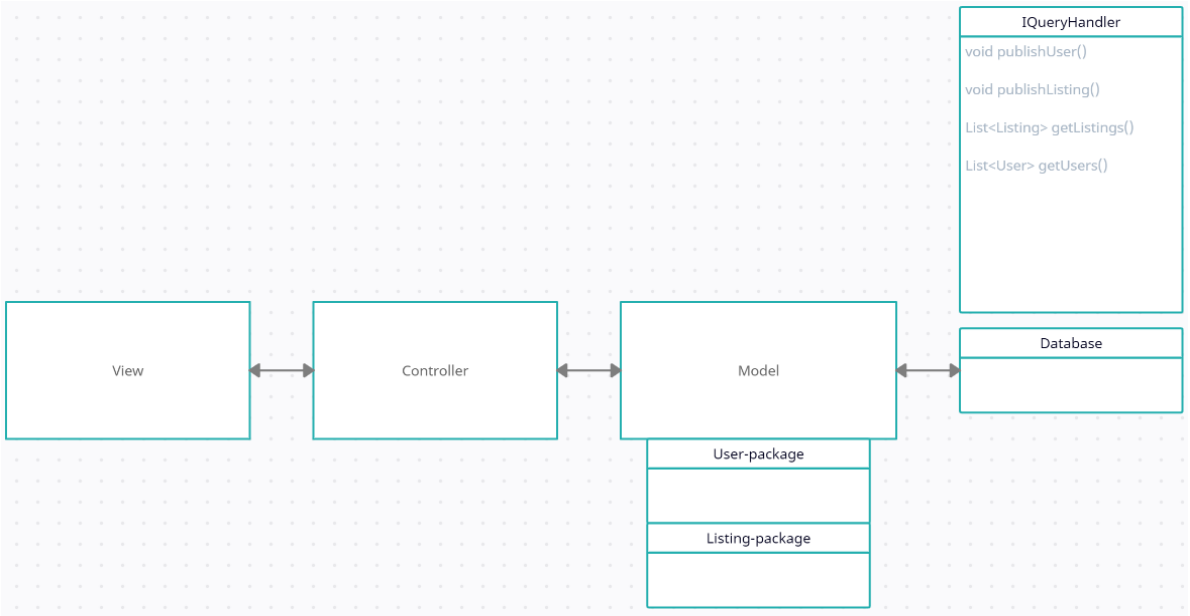

Figure 1: Blocks representing the design model of the program, a clear MVC pattern

The structure follows **MVC** which first and foremost requires that the model package is not dependent on any other part of the program. This makes the code modular and creates the possibility of using different views and controllers for the same model.

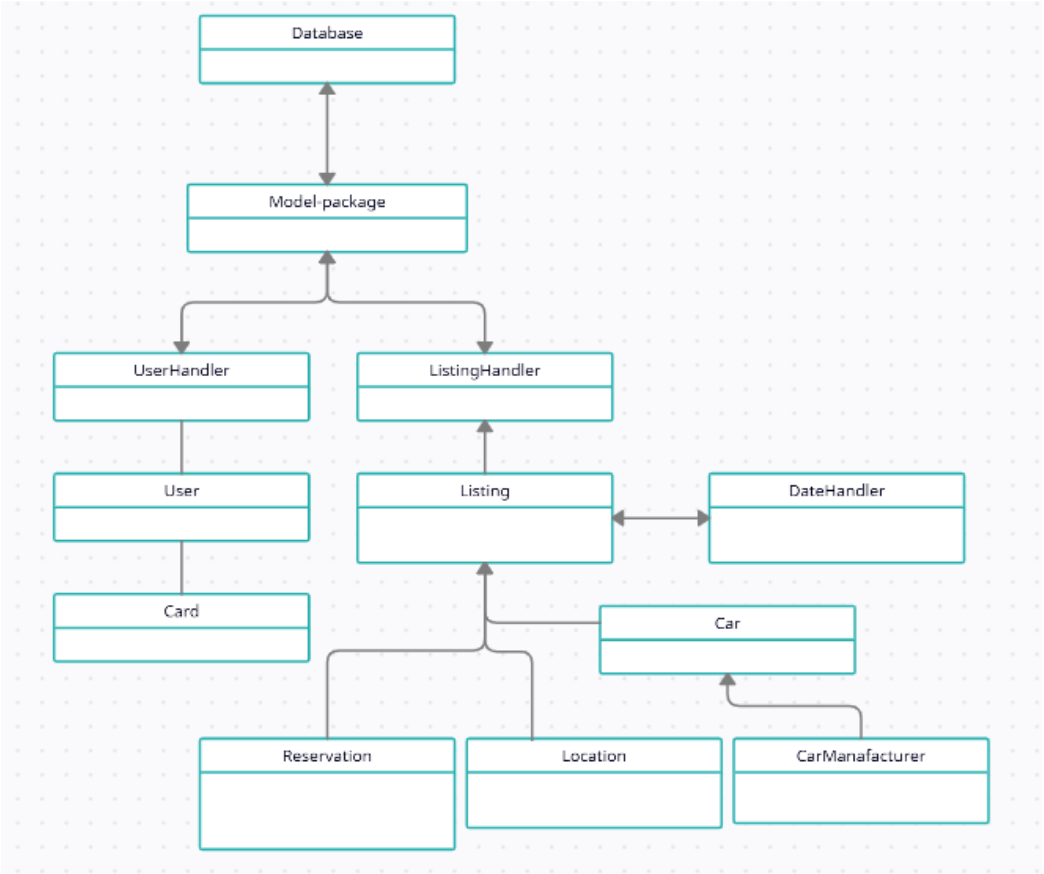

Figure 2: The model block

3.2 Relation between domain model and design model

The domain model and the design model were closely related, however during our process we have changed our design model as we received feedback from our teacher and another group. Our domain model shows the application from a module perspective while the design model shows the application from a class perspective and how the classes are related to each other. Our design model was redesigned to better suit our application needs and for this reason the original domain model is not very close to the implemented design model.

4. Persistent data management

To store data we use the Google product Firebase Realtime Database which is a NoSQL cloud database used to store and sync data. From the Firebase database the data can be synced at any time and across different clients such as, the web, android and IOS. The data is stored in JSON format which updates in real time with the connected client.

There are a lot of advantages of using Firebase Realtime Database with the main advantage being that the data is updated in real-time. Therefore there is no need to make requests for the data changes or updates. With the database use of data synchronization when data changes as every time there is a change it will reflect the connected user instantly. Another advantage of Firebase is that the apps you create will remain responsive even if the connection to the database is lost. When the connection is established again the user will receive the data changes from the database. Finally the data in the database is easily accessible through the web and it's possible to manage your database from both PC and mobile devices.

As we developed our program we noticed that a realtime database came with its difficulties. Every query or request to the database was handled using asynchronous functions which we decided we were not capable of managing in the time frame of this project. For this reason we used a temporary solution to only sync with the Firebase Database on program

startup and when it is exited. Note that program exited means when you leave the Android home screen.

5. Quality

In order to test the program the testing framework **JUnit** [2] was used. This made it easy to test functionality without a graphical interface initially. The tests are located in a separate folder test.

Analysis result from PMD and Find bugs:

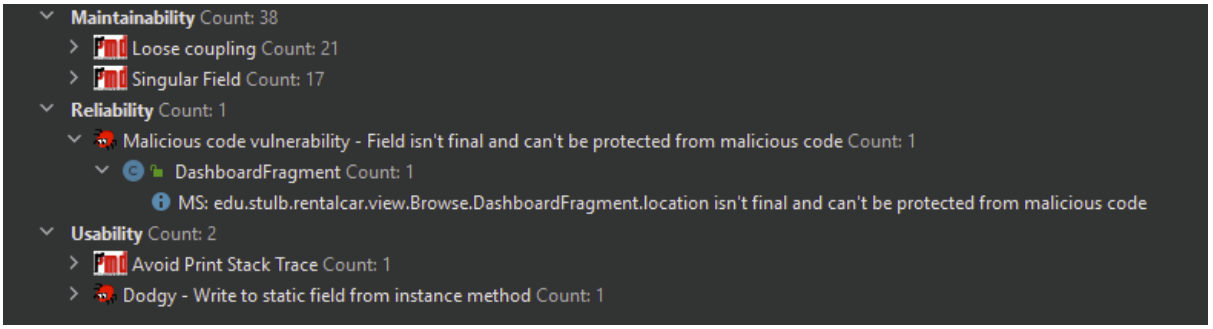

Figure 1: PMD and Find bugs result [3][4].

Analysis result from Intellij code analyzer dependency matrix :

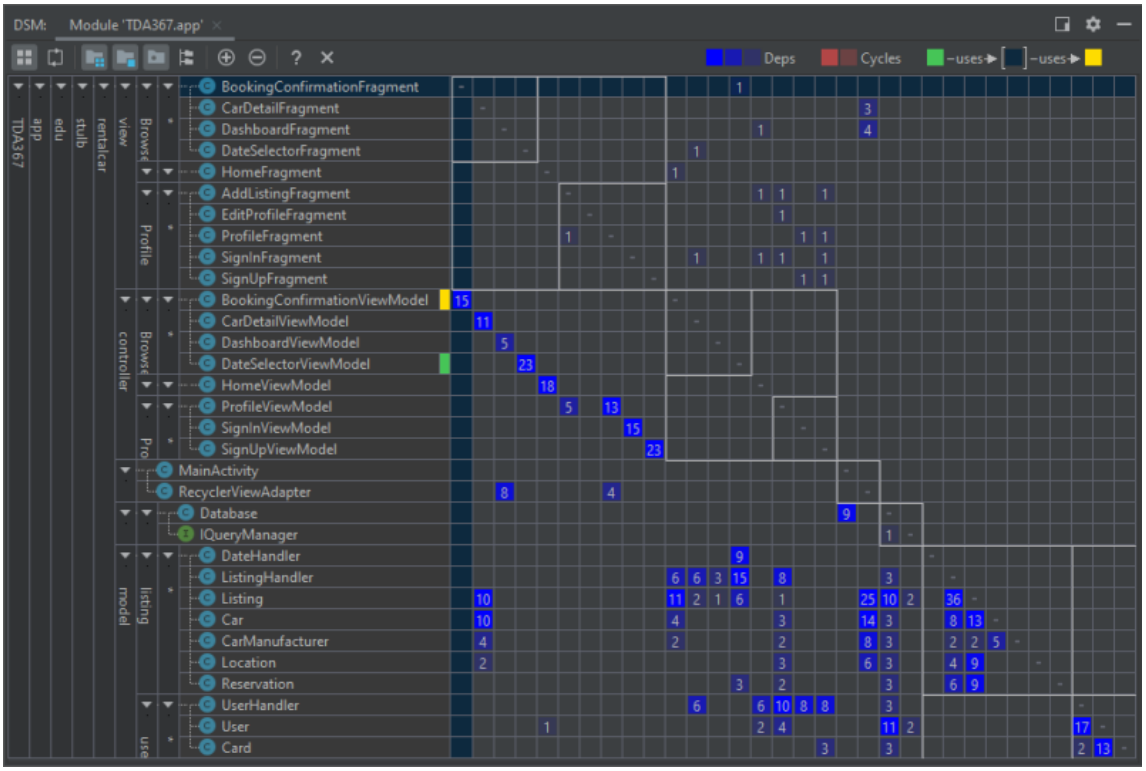

Figure 2: Intellij code analyzer dependency matrix result [5].

5.1 Access control and security

Rent A Car is an application which can be used by anyone. All registered and signed in users can use all the functionality of the app. The only other “role” apart from a signed in user is a user that is not signed in. When the user is not signed in they can not use all of the apps functionality they won't be able to list their own ad or book a car. However the non-signed user can use the search function and scroll through the different views like “home” and “dashboard”.

References

[1] Google product Firebase Realtime Database- https://firebase.google.com/
[2] JUnit. [Online]. Available: https://junit.org/junit5/
[3] Find bugs http://findbugs.sourceforge.net
[4] pmd https://pmd.github.io
[5] Dependency analysis matrix java.
https://www.jetbrains.com/help/idea/dsm-analysis.html