DENSE VS FOCUSED ON DQN FOR REINFORCEMENT LEARNING

Object Design Document Version 1.0

15.05.2020

Ersin Çebi

Prepared for ENGR4901



# Table of Contents

1. [Introduction 3](#_TOC_250013)
   1. [Object Design Trade-offs 3](#_TOC_250012)
   2. [Interface Documentation Guidelines 4](#_TOC_250011)
   3. [Definitions, Acronyms, and Abbreviations 4](#_TOC_250010)
2. [Packages 5](#_TOC_250009)
   1. [Models 5](#_TOC_250008)
   2. [Operations 5](#_TOC_250007)
   3. [Libs 6](#_TOC_250006)
   4. [Layout 7](#_TOC_250005)
   5. [Menu 8](#_TOC_250004)
3. [Class Interfaces 9](#_TOC_250003)
   1. [Class Diagram 9](#_TOC_250002)
   2. [Database Diagram 10](#_TOC_250001)
   3. Method Explanations 10
4. [References 20](#_TOC_250000)

**OBJECT DESIGN DOCUMENT**

# Introduction

This Object Design Document (ODD) defines the object-level design of the app to be developed. Proposed in Python data analysis development strategy.

# Object Design Trade-offs

FMWAY Işık is a transportation application exclusively for Işık University students. In FMWAY Işık, there are several trade-offs which editing trip as admin, adding/editing trip as driver, sign up to use the functions, rating and upload issues to admins.

Drivers can add their own trip and save it to the system. However, when an edition on a trip is done by admin, it is automatically approved by system and started to show in trip list without any approval. So, we can say that, admin has the over control on everything in the trip system. Users can add a trip, but admins can alter or even delete it without ask the users.

Drivers also can add or edit their own trips without any control. They can define their trip information, and simply add or delete them with a click. However, in case of report on a trip information by any user, the trip will be deleted immediately by admins and that user may be banned from the system.

In the FMWAY Işık system, signing up to the system is a must for joining a trip. All users are signed up as passengers in the beginning and can only join trips, not create one. The users who upload their license and get approved by the system, are defined as drivers, who can both join and create trips

Another trade-off is rating. Every user can rate their experience on the trip with the other person, such as driver can rate the passenger as “+” or “-“, and the passenger can rate the driver in an numeric rate system.

Uploading issue is another trade-off of FMWAY Işık system. Every registered user can send their issue to the system admins to tell his/her problems. Every admin can check report messages.

Other trade-offs we have made are:

* B uy vs Build : Our software project is built in android studio, and we are coding and developing the code together with our team.
* M emory space vs response time : Response time focused to be not take a very long

time, and more memory space can be used for faster response time.

-Internet Connection Required: The project need internet connection in order to complete the application task such as login, add-delete-update information on tables and database.

# Interface Documentation Guidelines

There are a list of rules we have applied in our developer team so we can understand and develop the code written (individually) together easier, and make it more readable.

These rules are:

* + - Buttons are labeled as “<ActivityName>Button” for onClick events
    - Classes are named with function name for instance; AddTripActivity, SignUpActivity etc.
    - Methods are named with verb phrases, fields, and parameters with noun phrases.
    - Start and End of certain functions are shown in comment lines.
    - Since we worked with Parse, every pushes ands posts to Parse Server are came with some callback functions including exceptions.

-The functions and the classes will be written based on object oriented development, not hard- coding.

Login page is the main page for all of the users. After the login, with related to user roles, different kind of main pages are served to the users. With respect to user roles, different kind of functions will be served to the users, such as, add trip for driver, join trip for passenger and so on.

FMWAY Işık has simple log-in and sign-up views. In Sign-in page, system asks to new users to enter the needed information. In login page, there are just e-mail and password field.

# Definitions, Acronyms, and Abbreviations

MVVM- Model View ViewModel Admin: Actor of FMWay Işık System.

Database: The collection of large data set/stack. Feedback: The answer of the system for the operation.

GitHub: A web-based hosting service, mostly used for computer code. Passenger: Actor of FMWay-Işık System.

Driver: Actor of FMWay-Işık System.

Trips: The specific travels that drivers create and passengers attend to.

Parse Server: Open source Backend-as-a-Service(BaaS) framework initially developed by Facebook.

Java 8: A programming language. Template: A guide of making patterns.

Android Studio: An Integrated Development Environment (IDE) developed by IntelliJ.

# Packages

FMWAY Işık is designed and implemented as an Android application. There are a lot of Android system packages, but there are 5 designed and implemented packages by us which are models, operations, libs, layout and menu.

Our project classes include Java,XML,Parse Query commands. XML is used for creating and designing the “view” part of the application. Java classes and activities are used for developing the “UI”, which is the “controller” side of this application. Java Android Library is used on the application development. The “model” side is the database, which is imported in the project, “build.gradle Scripts”. PARSE server database system is used for the project and the tables of the project exists on the PARSE Server “[www.back4app.com](http://www.back4app.com/)”.

# Models

In this package, the system object classes are stored.

# Operations

In this package, all operations of the application are stored.

# Libs

In this package, common variables are defined

# Layout

All layouts that are used by activities are stored in this package.

# Menu

This package contains all menu layouts for all user types.

# Class Interfaces

# Class Diagram

# Database Diagram

* 1. **JavaDoc Export**

# References

**1.** Bruegge B. & Dutoit A.H.. (2010). Object-Oriented Software Engineering Using UML, Patterns, and Java, Prentice Hall, 3rd ed.