MARMARA UNIVERSITY

FACULTY of ENGINEERING

DEPARTMENT of COMPUTER ENGINEERING

CSE3044 – Software Requirements Specification

“TravelBuddy”

Group Members

150116065 – Deniz Arda Gürzihin

150115002 – Ömer Cem Ağaoğlu

170214013 – Mehmet Can Yüney

1. Introduction

Discovering rarely known natural and cultural places is one of the biggest problems while visiting a foreign country or city. Although popular cultural places such as museums are easy to find, finding a natural places or rarely known cultural places is challenging.

Smart phones and its' applications one of the most popular terms in 21th century, so developers and investors pay attention to these topics. With the help of these technologies, individuals' desire for fast and easy solutions increased rapidly.

The present project concerns an application for helping people while they are discovering new countries or even their own country not by just popular cultural places but also with its' natural beauties. Hence this project is expected to manage the issues while visiting a foreign country and not being able to see its natural beauties. Also this project will be able to help people to find places that contain specific objects such as trees and rivers.

1.1. Purpose

This project aims to help people while they are discovering new places but not just with known popular places but also its' natural beauties.

We think this project is worthwhile because we are aware that finding popular cultural places are relatively easy when it comes to find rarely known places and natural beauties. We believe this project will help the users while they are travelling and wanting to visit natural places such as a relaxing place near a river or a place that is convenient for picnic.

1.2. Scope

The TravelBuddy application will provide users the ability to access information about rarely known places from a mobile device using the Android platform. Platform independency is not in our scope, our app will only run in Android platform.

The application will provide lots of options for the user. After user enters his/her account, he/she will be able to use every option that the system provides. However without login, users still will be able to find a place but scoring or messaging the other users will not be available without login.

The auto-tagger system will be available while individuals want to post a photo for a place. However auto-tagger system will only show known objects so it might not show every object in the photo.

Users will be able to log in to the application with their own mail addresses. Users' personal information and their GPS are not shown to other users.

1.3. Definitions, Acronyms and Abbreviations

- TravelBuddy Application – the product that is described in the document
- User - someone who uses a product, machine, or service
- GPS - global positioning system
- Operating System (OS) - a set of programs that control the way a computer system works, especially how its memory is used and how different programs work together
- Google Map - Google Maps is a web mapping service developed by Google [3]

2. Design Considerations

2.1. Operating Environment

TravelBuddy will be operated on Android operating system devices so assumes that users have qualified skills with using android devices. The database part of TravelBuddy is intended to create on SQLite.

2.2. Fault Tolerant Design

Exceptions will be handled by using general exception handlers of Java, Android SDK and SQLite database. Some of the examples that could be handled are connection loss of WIFI or mobile data while signing in to software or making search on software, connection loss of database between software. Therefore users must have an internet connection like Wi-Fi or mobile data.

2.3. Design Conventions

Travel Buddy is an Object Oriented Software so general Object Oriented Programming methodology will be used for this application. Since Travel Buddy is Object Oriented software "Waterfall Software Development" is more suitable and has been used to develop.

2.4. User Interface

2.4.1. General Information

When you open the TravelBuddy, users will be face with the map of Turkey, and then if they want, they can create or login to their account that has already been created via their email addresses. Non-logged in users can only make search about locations, saw the post that already shared or can look what is close to them. Logged in users can additionally share pictures, message with someone or comment on any other post.

2.4.2. Expected Input

All users can obtain data with the following inputs (If the user logged in or not),

- City name
- Place name
- Tag
- Current location of the user however this requires GPS and its usage verification

Sliding on screen is an input for software and the rating will also be considered as an input.

If the user chooses to be logged in, TravelBuddy can provide more options for them however this will require more input such as:

- A valid e-mail account to sign-up
- An image or a comment about a place

2.4.3. Expected Output

Inputs taken for the search operation will display output as some location shown with position symbol on the map. If a place cannot found with the specific input, the program will display an error message like “The places you have been searched could not find.” If any of input is not supplied, TravelBuddy think that NULL character has been choosed for that input.

If the user logged in to the systems to use more operation then they can upload a picture to a certain place or they can make comment to any picture or they can message to other users.

Sliding and scaling on map will be the output for finger action. If we slide or scale, the map will be changed it is shape. Additionally TravelBuddy will have a button which will show the closest possible place to visit and spend time there.

3. System Design

3.1. System Decomposition

The figure that is given in the below, represents the whole system as decomposed version. System will have mainly 4 components. Each component has back and home option so it does not show in the figure.

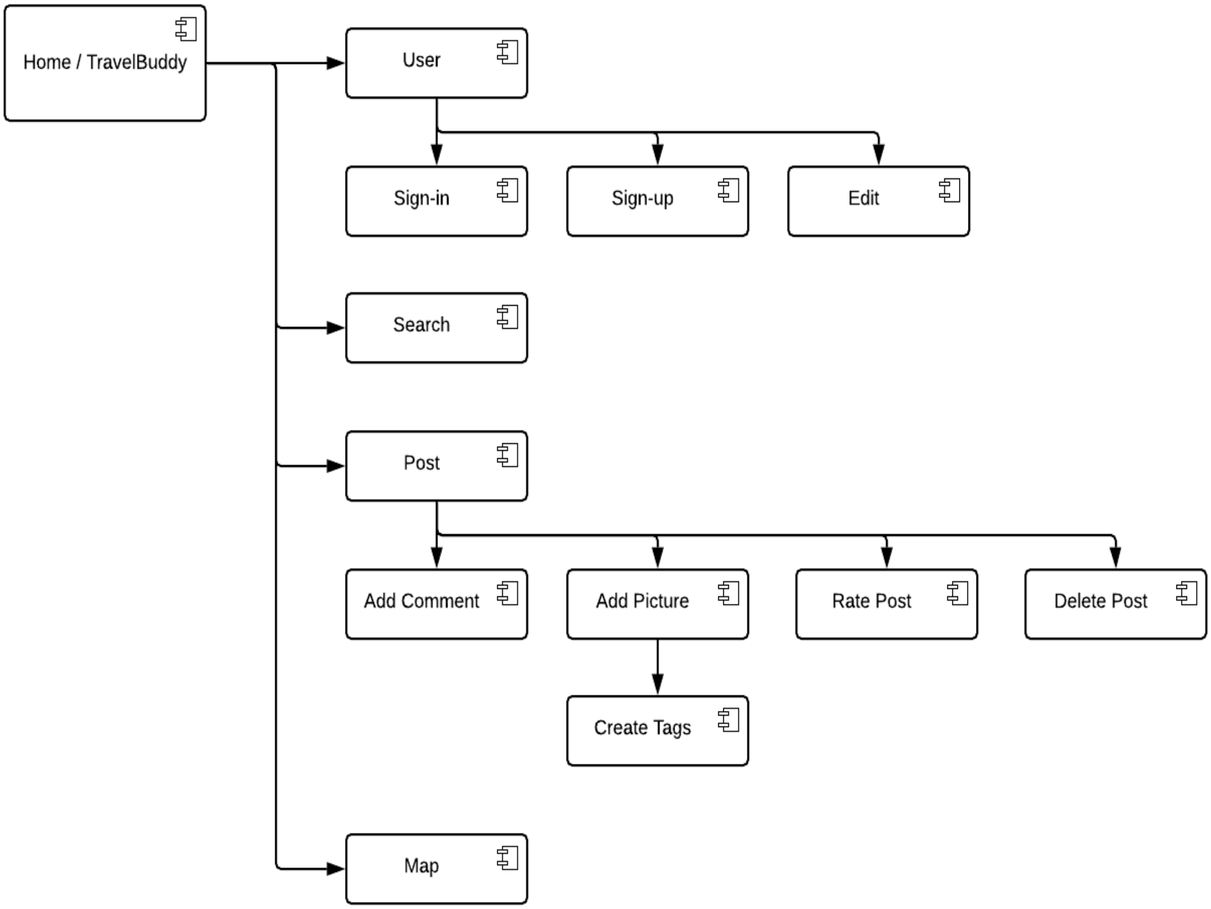

Figure 1 – Decomposition Diagram of TravelBuddy

3.2. Database Design

TravelBuddy will have database for pictures, comments, rating and also users. System will be initialized with few places, however initially there will be no normal users but 3 admin. For this reason, we will use SQLite database because it is easier to handle via Android Operating System.

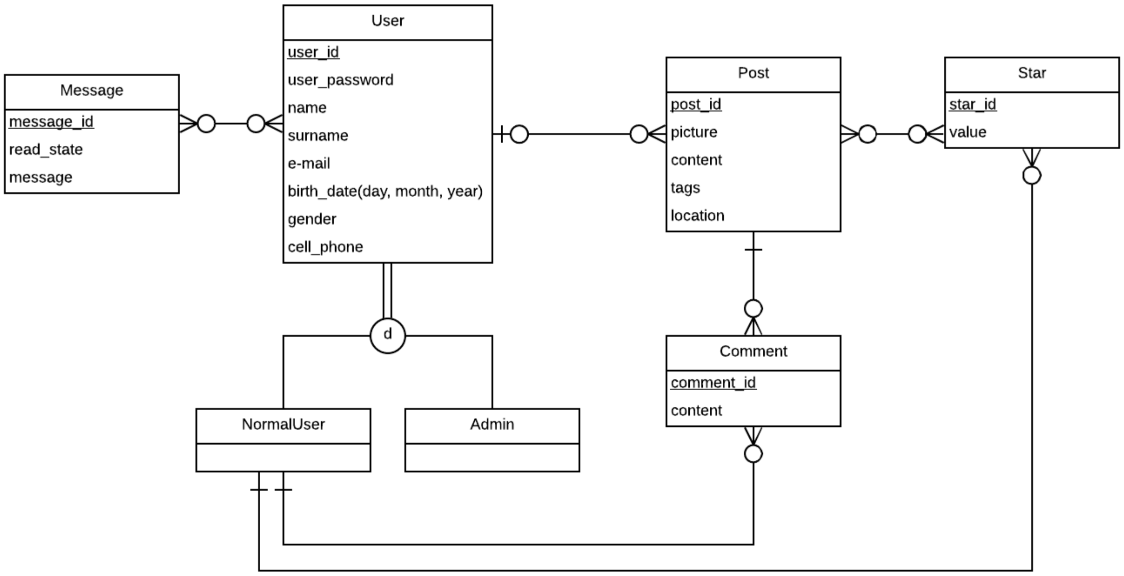

Figure 2 – ER Diagram of TravelBuddy

3.3. Class Diagram

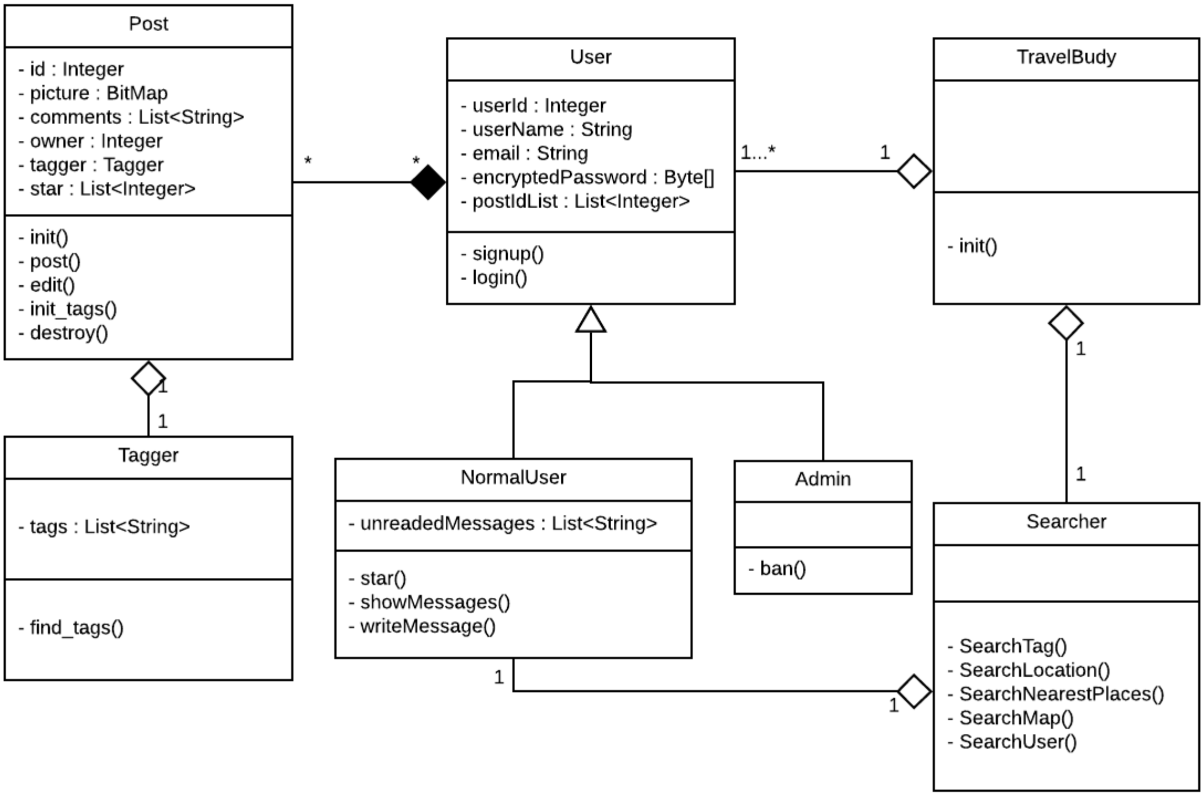

Figure 3 – UML Class Diagram

As shown in the Figure 3, tagger will be abstracted from the system. Whenever the users want to publish/post a picture about a place, depends on the users desire tagger will tag the objects that are detected via object detection algorithm provided by YOLO (You Only Look Once) [4] or with an external library called IMAGGA [5].

3.4. Activity Diagram

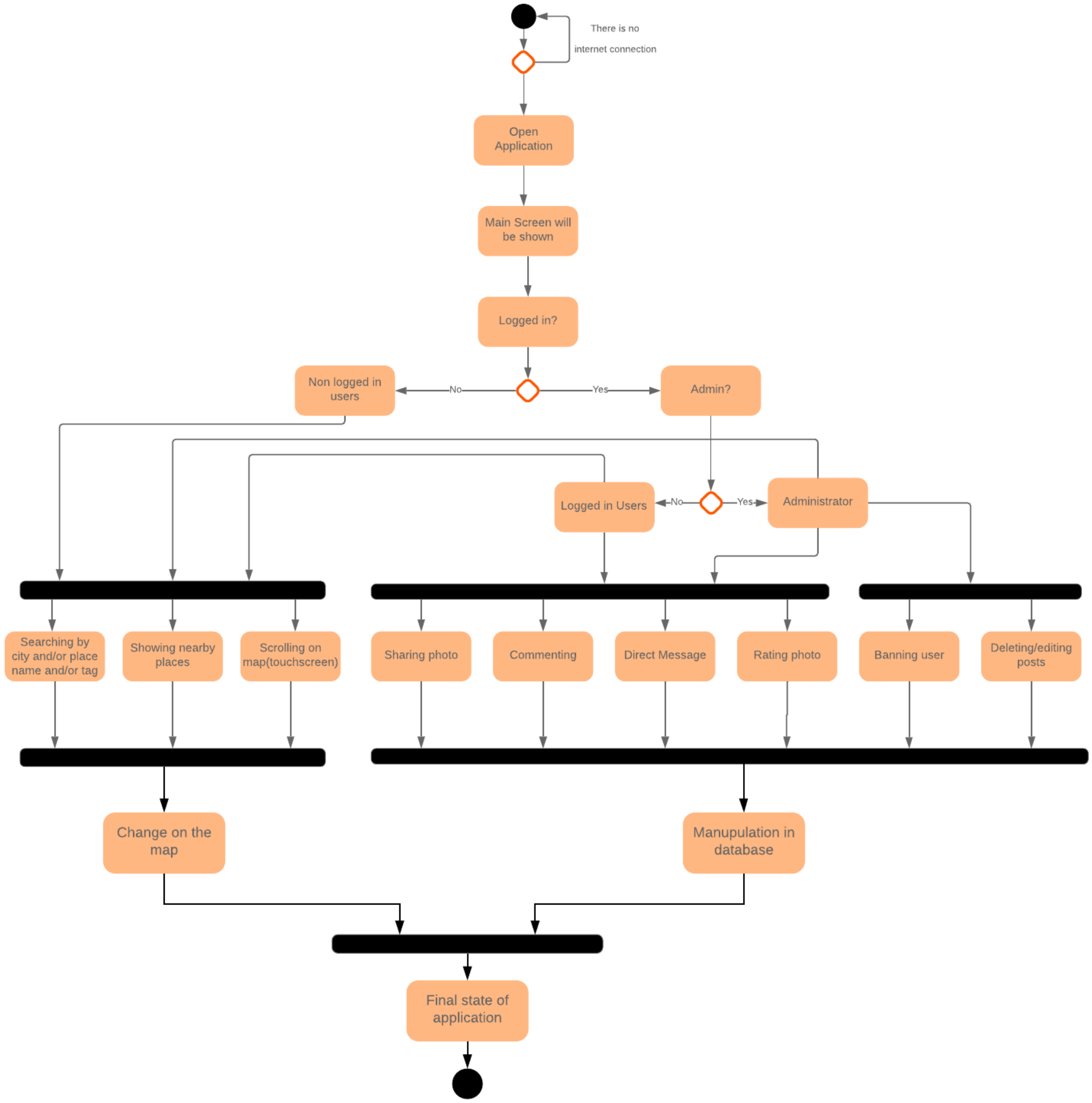

Figure 4 – Activity Diagram

3.5. Use Case Diagram

Following diagram shows the main case diagram of the system.

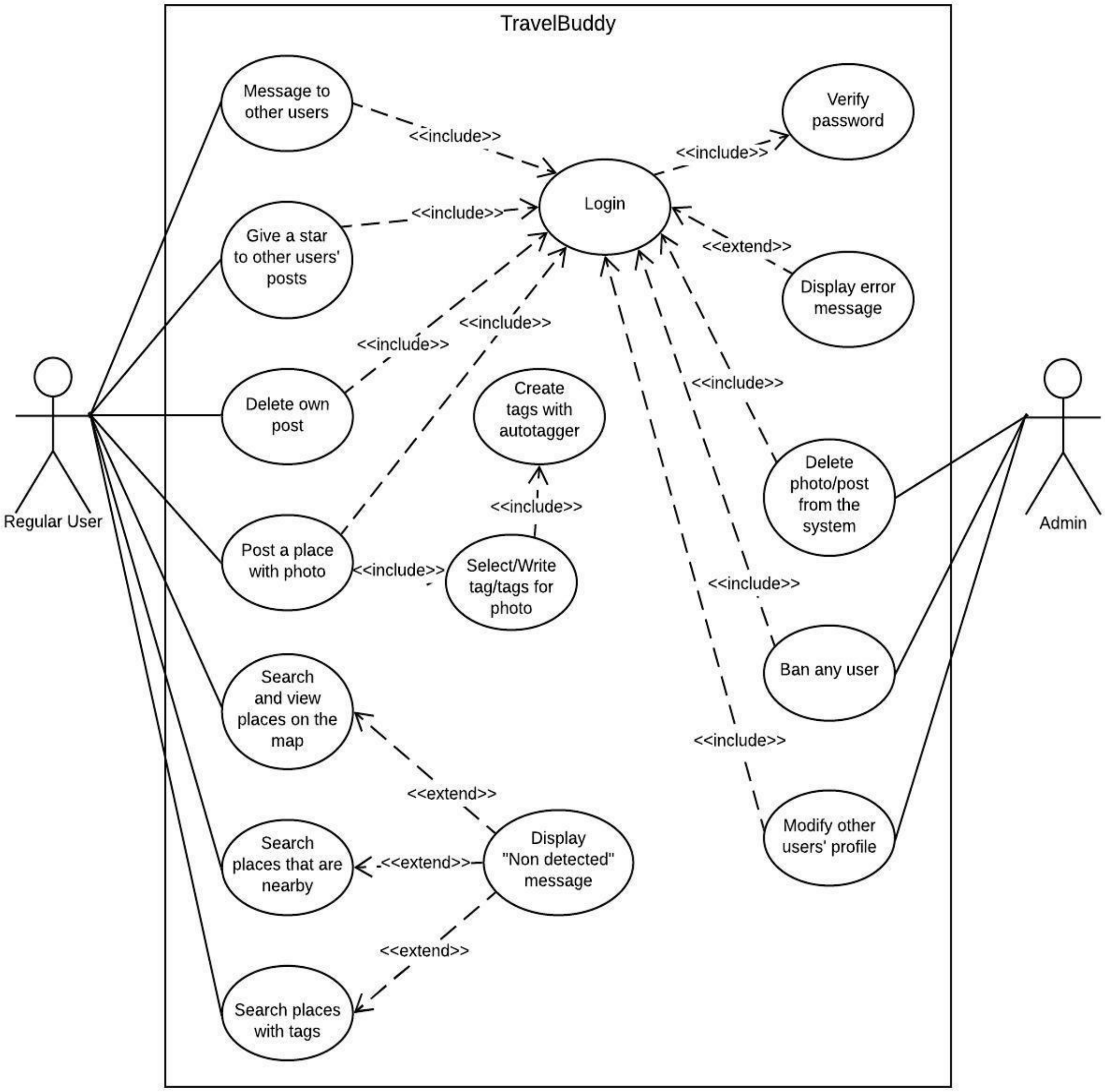
TravelBuddy
Regular User
Admin
Message to other users
Give a star to other users' posts
Delete own post
Post a place with photo
Search and view places on the map
Search places that are nearby
Search places with tags
Login
Verify password
Display error message
Create tags with autotagger
Select/Write tag/tags for photo
Display "Non detected" message
Delete photo/post from the system
Ban any user
Modify other users' profile
<<include>>
<<extend>>

Figure 5 – Use Case Diagram

4. Division of Responsibilities

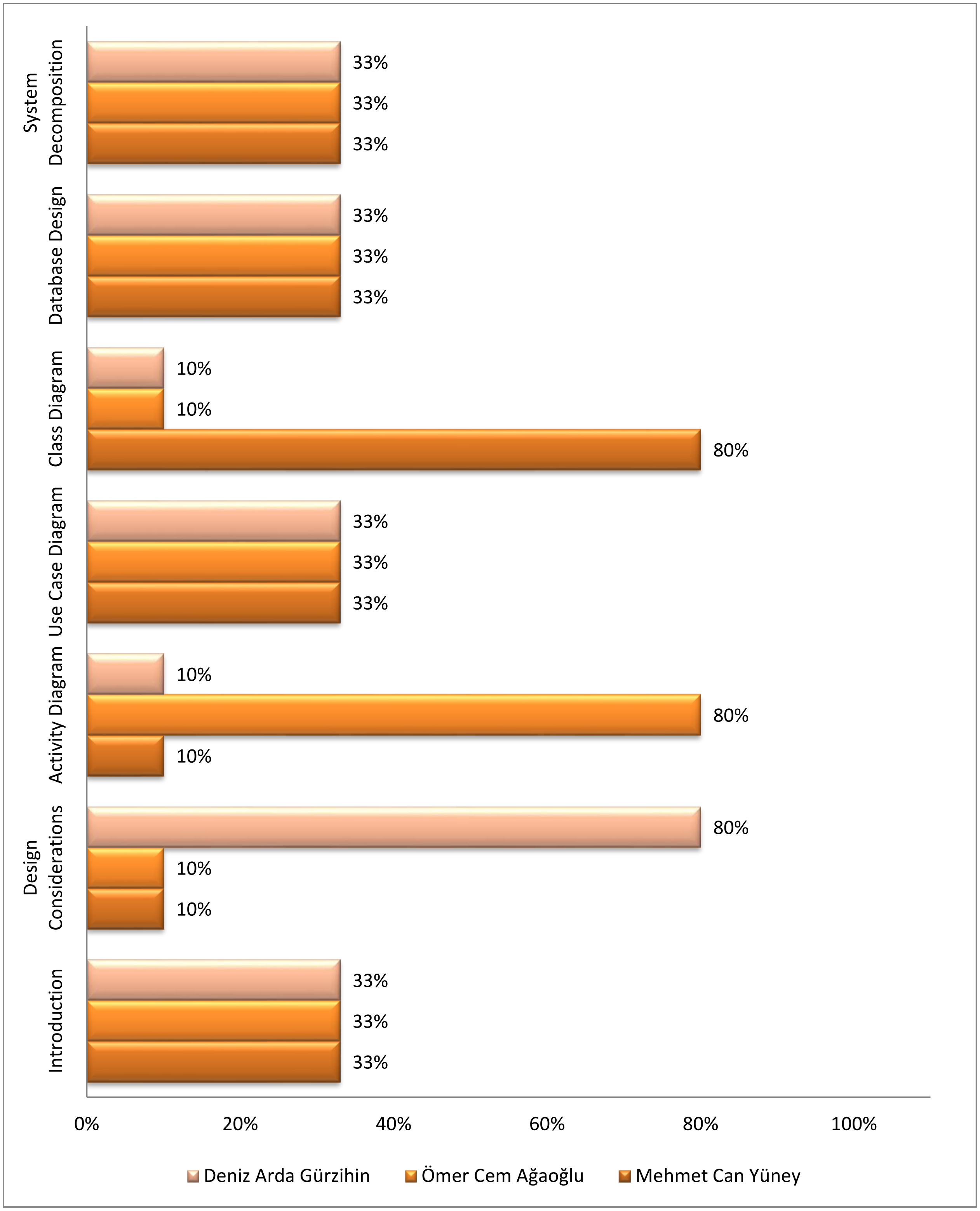

Figure 6 – Work Sharing Chart

References

[1]Medya Merkezi. (n.d.).Retrieved March 07, 2019, from https://tripadvisor.mediaroom.com/trabout-us

[2] TripAdvisor03. (n.d.). Retrieved March 7, 2019, from https://www.tripadvisor.com.tr/

[3] Geo-location APIs | Google Maps Platform | Google Cloud. (n.d.). Retrieved March 14, 2019, from https://cloud.google.com/maps-platform/

[4] REDMON, Joseph; FARHADI, Ali. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.

[5] Imagga Technologies Ltd. (n.d.). Auto-Tagging API. Retrieved April 12, 2019, from https://imagga.com/solutions/auto-tagging.html