

**ÇANKAYA UNIVERSITY FACULTY OF
ENGINEERING
COMPUTER ENGINEERING DEPARTMENT**



**PROJECT REPORTCENG 408
INOVATITE SYSTEM DESIGN AND
DEVELOPMENT II
RECOMMENDATION SYSTEM FOR
TRAVELERS**

PROJECT ID: 2022-05

TAHA KOYUTÜRK	201811044
HÜSEYİN DİKBASAN	201611014
PELİN SU GÖK	202011405
ABDULLAH ÖMÜR ŞENOCAK	201811403
CEREN EROĞLU	201811029

ADVISOR: PROF. DR. AHMET COŞAR

Contents

Abstract	9
Özet	9
1. Introduction	10
1.1 Motivation	10
1.2 Problem Statement	10
1.3 Solution Statement	10
2. Literature Review	11
Abstract	11
2.1 Introduction	11
2.1.1 Personalized Recommendation Systems	12
2.1.1.1 Content-Based Filtering.....	12
2.1.1.2 Collaborative Filtering.....	12
2.1.1.3 Hybrid Filtering.....	12
2.1.2 Recommender System Feedback Techniques	12
2.1.2.1 Explicit Feedback Technique	12
2.1.2.2 Implicit Feedback Technique	12
2.1.2.3 Hybrid Feedback Technique.....	13
2.2 Machine Learning.....	13
2.2.1 Supervised Learning.....	13
2.2.1.1 Regression	13
2.2.1.1.1 Linear Regression.....	13
2.2.1.1.2 Logistic Regression	13
2.2.1.2 Naïve Bayes.....	14
2.2.1.3 KNN (K-Nearest Neighbor Algorithm).....	14
2.2.2 Unsupervised Learning.....	14
2.2.2.1 Clustering	15
2.3.2.1.1 K-Means Clustering.....	15
2.3.2.1.2 Hierarchical Clustering.....	15
2.3.2.1.3 KNN Clustering.....	15
2.3.2.2 DBSCAN.....	15
2.3.3 Data Analysis	15
2.3.3.1 PCA	15
2.3.3.2 EDA.....	16
2.3 Mobile Application Development	16
2.3.4.1 Java.....	16
2.3.4.2 Kotlin.....	16

2.3.4.3 Dart.....	17
2.3.4.4 Objective-C	17
2.3.4.5 Swift	17
2.4 Database	17
2.4.1 SQL	17
2.4.1.1 MySQL.....	17
2.4.2 NoSQL.....	18
2.4.2.1 Firebase	18
2.4.2.2 MongoDB.....	18
2.4.3 Differences Between SQL and NoSQL.....	18
2.6 Related Work Overview	18
2.6.1 Google Maps	18
2.6.2 Travel Shop Turkey.....	19
2.6.3 City Mapper.....	19
2.6.4 LiveTrekker	19
2.6.5 Time Out	20
2.6.6 Zomato	20
2.5 Conclusion.....	20
3. Software Requirements Specification (SRS).....	20
3.1 Introduction	20
3.1.1 Purpose	20
3.1.2 Scope of Project.....	21
3.1.3 Glossary.....	21
3.2 Overall Description	21
3.2.1 Product Perspective	21
3.2.1.1 Development Methodology	21
3.2.2 User Characteristics.....	22
3.2.2.1 User	22
3.2.2.2 Admin.....	22
3.3 Requirements Specification.....	22
3.3.1 External Interface Requirements	22
3.3.1.1 User Interface	22
3.3.1.2 Hardware Interfaces.....	22
3.3.1.3 Software Interfaces.....	22
3.3.1.4 Communications Interfaces	22
3.3.2 Functional Requirements.....	22
3.3.2.1 Open Page.....	22

3.3.2.2	Register Page	23
3.3.2.3	User Login Page	24
3.3.2.4	Admin Login Page.....	25
3.3.2.5	Setting Profile.....	26
3.3.2.6	Recent Places.....	27
3.3.2.7	Favorites	28
3.3.2.8	Map.....	29
3.3.2.9	Commenting and Rating.....	31
3.3.2.10	Deleting Comments and Ratings	32
3.3.2.11	Adding/Removing Picture	33
3.3.2.12	Delete Inappropriate Content	35
3.3.2.13	Blacklist User	36
3.2.14	Recommendation System	37
3.3.3	Software System Attributes.....	38
3.3.3.1	Portability	38
3.3.3.2	Usability	38
3.3.3.3	Reliability	38
3.3.3.4	Availability	38
3.3.3.5	Security.....	38
4.	Software Design Document (SDD).....	38
4.1	Introduction	38
4.1.1	Purpose	38
4.1.2	Overview	38
4.1.3	Definitions and Acronyms Abbreviations	38
4.2	Architecture Design.....	39
4.2.1	Approach	39
4.2.2	Class Diagram	40
4.2.2.1	Android.....	40
4.2.2.1.1	User	40
4.2.2.1.2	User Provider.....	41
4.2.2.1.3	Post	41
4.2.2.1.4	Widget Tree	42
4.2.2.1.5	Splash Page.....	42
4.2.2.1.6	Introduction Page.....	42
4.2.2.1.7	Auth Methods	42
4.2.2.1.8	Auth Page	43
4.2.2.1.9	Firestore Methods	43

4.2.2.1.10	Storage Methods.....	44
4.2.2.1.11	Routes	44
4.2.2.1.12	Main Home Page.....	44
4.2.2.1.13	Wish List	45
4.2.2.1.14	Photos	45
4.2.2.1.15	Recent Places.....	45
4.2.2.1.16	Maps	45
4.2.2.1.17	Map Service.....	46
4.2.2.1.18	Auto-Complete Result	47
4.2.2.1.19	Place Result	47
4.2.2.1.20	Activity Page	47
4.2.2.1.21	Discover Page.....	47
4.2.2.1.22	Post Page	47
4.2.2.1.23	Profile Page	47
4.2.2.2	iOS.....	48
4.2.2.2.1	Landmark.....	48
4.2.2.2.2	Local Search Service	49
4.2.2.2.3	Location Manager-Extension	49
4.2.2.2.4	Location Manager-Class.....	50
4.2.2.2.5	MK Coordinate Region	50
4.2.2.2.6	Map View	50
4.2.2.2.7	Landmark List View.....	51
4.2.2.2.8	Comment	51
4.2.2.2.9	Comment View Model	52
4.2.2.2.10	Comments View	52
4.2.2.2.11	Comments Cell.....	53
4.2.2.2.12	Custom Input View.....	53
4.2.2.2.13	Post	53
4.2.2.2.14	Feed View Model-Class	54
4.2.2.2.15	Feed View.....	54
4.2.2.2.16	Feed Cell View Model.....	55
4.2.2.2.17	Feed Cell.....	55
4.2.2.2.18	Post Grid View Model.....	56
4.2.2.2.19	Post Grid View	56
4.2.2.2.20	Search View Model	57
4.2.2.2.21	Search View	57
4.2.2.2.22	Search Bar	58

4.2.2.2.23	User List View.....	58
4.2.2.2.24	User Cell.....	59
4.2.2.2.25	User	59
4.2.2.2.26	Profile View Model	60
4.2.2.2.27	Profile Action Button View	60
4.2.2.2.28	Profile View	60
4.2.2.2.29	Profile Header View	61
4.2.2.2.30	User Stat View.....	61
4.2.2.2.31	Auth View Model	62
4.2.2.2.32	Register View-Struct	63
4.2.2.2.33	Custom Text Field	63
4.2.2.2.34	Register View-Extension.....	64
4.2.2.2.35	Custom Secure Field.....	64
4.2.2.2.36	Image Picker.....	65
4.2.2.2.37	Coordinator.....	65
4.2.2.2.38	Image Uploader	65
4.2.2.2.39	Reset Password View	66
4.2.2.2.40	Content View.....	66
4.2.2.2.41	Login View.....	67
4.2.2.2.42	Main Tab View	67
4.2.2.2.43	Upload Post View Model	68
4.2.2.2.44	Upload Post View-Struct.....	68
4.2.2.2.45	Upload Post View-Extension	69
4.2.2.2.46	Text Area	69
4.2.2.3	Machine Learning.....	70
4.2.2.3.1	Google Review Rating Dataset	70
4.2.2.3.2	User Class.....	71
4.2.2.3.3	Evaluation Class	71
4.2.2.3.4	Cluster Class.....	72
4.2.2.3.5	Places Class	72
4.2.2.3.6	Prediction Class	73
4.2.2.3.7	Recommendation Class	74
4.2.3	Sequence Diagrams	75
4.2.3.1	Register Page.....	75
4.2.3.2	Login Page.....	76
4.2.4	Activity Diagram.....	77
4.2.4.1	Admin.....	77

4.2.4.2	Profile	78
4.2.4.3	Visit Place.....	79
4.3	User Interface Design.....	80
4.3.1	Android.....	80
4.3.1.1	Open Page.....	80
4.3.1.2	Login Page.....	80
4.3.1.3	Sign Up Page	81
4.3.1.4	Main Page.....	81
4.3.1.5	New Post Page.....	82
4.3.1.6	Map Page.....	82
4.3.1.7	Profile Page	83
4.3.2	Apple	83
4.3.2.1	Login Page.....	83
4.3.2.2	Sign Up Page	84
4.3.2.3	Feed	84
4.3.2.4	Explore Page.....	85
4.3.2.5	New Post Page.....	85
4.3.2.6	Map Page.....	86
4.3.2.7	Profile Page	86
4.3.2.8	Search Page	87
4.3.2.9	Another Users Profile Page	87
4.4	Constraints.....	88
4.4.1	Time.....	88
4.4.2	Performance.....	88
4.4.3	Application Constraints.....	88
5	Test Plan Document	88
5.1	Introduction	88
5.1.1	Overview	88
5.1.2	Scope	88
5.1.3	Glossary.....	88
5.2	Features to be Tested.....	88
5.2.1	Sign Up (SU)	88
5.2.2	Sign In (SI)	89
5.2.3	Sign Out (SO).....	89
5.2.4	Google Maps (GM)	89
5.2.5	Google Maps Recommendation	89
5.2.6	Main Page.....	89

5.2.7	Profile Page	89
5.3	Features not to be Tested.....	89
5.4	Pass/Fail Criteria	89
5.5	Exit Criteria	89
5.6	Test Design Specification.....	89
5.6.1	Sign In Part.....	89
5.6.2	Sign Up Part	90
5.6.3	Maps Part.....	90
5.6.4	Social Media Part	90
5.7	Detailed Test Cases	91
5.7.1	SIUS01	91
5.7.2	SIUS02	91
5.7.3	SIUS03	92
5.7.4	SIUS04	92
5.7.5	SUUS01.....	93
5.7.6	SUUS02.....	93
5.7.7	SUUS03.....	94
5.7.8	SUUS04.....	94
5.7.9	SUUS05.....	95
5.7.10	MAP01	95
5.7.11	MAP02	96
5.7.12	MAP03	96
5.7.13	MAP04	97
5.7.14	MAP05	97
5.7.15	MAP06	98
5.7.16	MAP07	98
5.7.17	MAP08	99
5.7.18	MAP09	99
5.7.19	MAP10	100
5.7.20	MAP11	100
5.7.21	SM01	101
5.7.22	SM02	101
5.7.23	SM03	102
5.7.24	SM04	102
6	Test Results	103
6.1	Individual Test Results	103
6.2	Summary of Test Results.....	104

6.3	Exit Criteria	104
7	User Manual	104
7.1	Login Page.....	104
7.2	Sign Up Page	104
7.3	New Post Page.....	105
7.4	Map Page.....	106
7.5	Profile Page	106
8	Project Work Plan	107
8.1	Project Work Plan	107
8.2	Project Work Plan-Detailed.....	108
9	Conclusion.....	108
10	References	109

Abstract

In general, people like to travel, see new places, explore and share the memories they have spent there with people by photographing them. We see this a lot on social media. Based on this idea, we thought to collect and develop these activities together thanks to our mobile application Dec. Users using this application can recommend the places they visit to other users by voting, sharing photos, commenting. They can photograph their memories of these places and share them with people on their profile. In addition, our application is similar to the places that users visit, it can recommend places that the user may like depending on their location and, if desired by the user, it can draw a road map so that October can easily go to the proposed location. In this way, users in the system can see more places, discover more different places and also share the locations they have visited with users.

Özet

İnsanlar genellikle gezmeyi, yeni yerler görmeyi, keşfetmeyi ve oralarda geçirdikleri anıları fotoğraflayarak insanlarla paylaşmayı severler. Buna sosyal medyada da çok gez görmektediriz. Bu düşünceden yola çıkarak bu aktiviteleri mobil uygulamamız sayesinde bir araya toplayıp geliştirmeyi düşündük. Bu uygulamayı kullanan kullanıcılar gezdiği yerleri oylayıp, fotoğraf paylaşp, yorum yaparak diğer kullanıcılara önerebilirler. Bu yerlerdeki anılarını fotoğraflayıp profilinde insanlarla paylaşabilirler. Ek olarak uygulamamız kullanıcıların gezdiği yerlere benzer, kullanıcının beğenebileceği yerleri konumlarına bağlı olarak önerebilir ve eğer kullanıcı tarafından istenirse önerilen konuma rahatlıkla gidebilmesi için bir yol haritası çizebilir. Bu sayede sistemdeki kullanıcılar daha fazla yer görebilir, daha farklı yerler keşfedebilir ve ayrıyeten ziyaret ettikleri konumları kullanıcılarla da paylaşabilirler.

1. Introduction

1.1 Motivation

Over the years, sociability among people, recommendations on the internet and interest and sharing in places evaluated, the more widespread social media, travel planning have increased a lot. In this regard, there is no mobile application that combines all these features and more. This situation made us think about the mobile application we made during the project process.

1.2 Problem Statement

Due to the lack of a mobile application that is widely used today, which contains most of the requirements, the users cannot make their travel plans suitable for them, and the places they go afterwards can be liked with a correct and appropriate planning, but they are not liked. In addition, situations caused by the inactivity of user communication, the uncertainty of the good or bad features of that place, due to the fact that other users' opinions about the places they have been to before are not stated. Finally, due to the lack of a mobile application that contains most of the requirements, the lack of a fully efficient and fast service to the users, therefore, the users may not notice the places suitable for them in a particular region, causing them to search for places suitable for them for a longer period of time, causing many users to give up on travel. may cause. As a result, it may indirectly cause a decrease in travel rates. As an example of some commonly used applications; Travel Shop Turkey, Google Maps, City Mapper, Live Trekker, Time Out, Zomato.

1.3 Solution Statement

In the common applications we have designed and similar, it will maintain the existence of all necessary features, sociability and communication, as well as make recommendations according to the preferences and tastes of the users, make the users happier from the place they travel, express their thoughts about the places visited within the framework of certain etiquette

rules and go there. By making an application that will give ideas to other users who want, in short, that will both increase sociality and help plan the travel better and more comfortable than users, users can be used faster and more actively.

2. Literature Review

Abstract

People have always had an innate need to travel. These days travel and adventure has become the most trending entertainment as well. With the development of information technology and social media, there are numerous possibilities and opportunities in fetching suited information that can yield to setting up an appropriate travel plan and hence enhance the quality of travel. Over the past few years, a Recommender System (RS) has become increasingly important. It helps users to discover information and settle on choices that they prefer. By filtering individualized information based on the user's preferences from a big quantity of data, it is a software tool and approach that offers a suggestion based on the customer's taste to find a new appropriate product for them. To help travelers prevent information overload and provide travel tips, recommendation systems (RS) have been frequently used.^[1] By recommending rich digital information, perspective services, and fellow users' opinions and ratings, recommendation systems for the tourism industry aspire to improve the experience of visitors. This survey reviews mobile tourism recommendation systems and provides insights into their offered services. We reviewed each method and technique in detail and tracked their development through time in the travel and travel industry before comparing them to our project.

Keywords: Recommendation system, Types of the recommendation system, Travel, and Tourism

2.1 Introduction

The issue of information selection and search has become more difficult as a result of the rapid increase in internet environments; users are now faced with a confusing variety of options that they may not have the time or knowledge to analyze. According to Google statistics, about 80 percent of people seek information from the web for planning their upcoming holidays. Another analysis uncovers that 52% remain positive while 48% reexamine their travel arrangements based on reviews on online platforms.^[2] Users of recommender systems (RSs) can utilize them to manage their information overload when they are online.^[3] The use of cell phones as the main platform for information access is a relatively recent development in recommendation systems. The unique qualities of mobile travel offer possibilities and difficulties for the development of reducing customized services. Many industries have used recommender systems, especially Facebook and other social media platforms, the entertainment industry (Netflix and YouTube), e-commerce (Amazon and Flipkart), and the travel industry (Tripadvisor and Trivago).^[4] Recommendation Systems

Recommendation systems have expanded over the past few years. It aids the customer in gathering knowledge and making decisions. It is a software tool and approach that offers suggestions based on the customer's taste to discover new relevant items for them by filtering customized information based on the user's preferences from a huge amount of data. Because they assist users in finding products they might not have otherwise found, recommender systems are a helpful alternative to search algorithms.^[5] It's important to note that recommender systems are frequently built utilizing search engines that index unconventional data. In 1992, Goldberg, Nichols, and Oki&Terry created The First RS. A recommendation system is a solution to the problem of giving the consumer the right items aside from looking through many different options. Although people's likes differ from

one another, there are some patterns that they all share. There are multiple types of recommendation systems.

2.1.1 Personalized Recommendation Systems

Personalization is the process of matching the appropriate users with the appropriate services, goods, or content. When done properly, it enhances user engagement, which refers to any interaction users have with a service, website, app, etc. There are 3 types of filtering when personalizing a recommendation system.^[6]

2.1.1.1 Content-Based Filtering

By identifying similarities, the content-based filtering (CBF) algorithm makes suggestions based on particular item features. Based on description data, which could include user or item attributes, these systems create data profiles. The profiles that have been built are then utilized to suggest products that are comparable to those that the user has previously enjoyed, purchased, watched, or listened to.

2.1.1.2 Collaborative Filtering

Collaborative filtering (CF), the algorithm that is used the most frequently, generates suitable recommendations based on how various users interact with the target items. These recommender systems collect data on prior user activity and mine it to determine which products to show to other active users who have similar likes.^[7] This may include things like the music they've listened to, items they've added to their shopping cart, adverts they've clicked on, movies they've reviewed in the past, etc. Such a system aims to anticipate how a person will respond to things they haven't yet engaged with.

2.1.1.3 Hybrid Filtering

To address the drawbacks and shortcomings of pure recommendation system models, hybrid filtering was developed. The hybrid models combine numerous recommendation strategies under one roof to produce recommendations with a higher degree of accuracy and fewer drawbacks than any single one.^[8] Collaborative filtering is typically combined with other techniques to address the cold start issue. However, this is not a requirement since techniques can be integrated into various ways.

2.1.2 Recommender System Feedback Techniques

Information feedback is the fundamental component of a recommendation system since it provides the information the RS need to make appropriate recommendations to the customers based on their preferences. Generally speaking, there are three categories of feedback techniques.

2.1.2.1 Explicit Feedback Technique

The system must prompt users to rate products to gather explicit feedback from them. The system determines how relevant or similar an item is to users' choices after gathering feedback. Although this enables the recommender to understand the user's actual viewpoint, it is frequently difficult to gather because it involves direct user interaction.

2.1.2.2 Implicit Feedback Technique

Contrary to explicit feedback, gathering implicit input doesn't require user involvement. By observing activities taken by users, such as which things they viewed, where they clicked, what they bought, or how long they spent on a web page, the system automatically analyzes users' preferences.^[9]

2.1.2.3 Hybrid Feedback Technique

To improve prediction quality, hybrid feedback employs both explicit and implicit feedback. The system must be able to gather both explicit and implicit feedback from users to employ the hybrid method.

2.2 Machine Learning

ML is an artificial intelligence system that uses mathematical models that learn according to the data they receive and improve their performance. Machine learning algorithms are developed to recognize patterns and correlations in big data sets and to find the optimum algorithm fit for this analysis rather than being explicitly programmed. Artificial intelligence means a machine aimed at producing systems that imitate human intelligence and it is a very comprehensive field, it can be said that machine learning is a subfield of artificial intelligence.^[10]

Machine learning algorithms are designed to classify events or data, find examples, predict results, and make decisions based on predicted results. Machine learning could be performed in a number of methods. They include learning to learn, reinforcement learning, supervised learning, unsupervised learning, semi-supervised learning, and unsupervised learning.^[11]

2.2.1 Supervised Learning

While the algorithm is being developed, the person who develops the algorithm needs to determine the strict rules and limits that the algorithm should follow while working. In order to anticipate events that might occur in the future, a supervised algorithm designed in a supervised manner can apply what has been learned in the past to new data by using labeled examples. The principle of operation is based on defining a set of input data and results, the algorithm is considered successful when it finds a match and a near-correct prediction as a result. The main purpose of supervised machine learning is to predict the target in the closest possible way using a function defined over a series of independent variables.^[12] Examples of supervised machine learning include linear and logistic regression, CART (Classification and Regression Trees), Naïve Bayesian classification, and the KNN (K-Nearest Neighbor) algorithm.

2.2.1.1 Regression

Regression is a statistical measurement that determines the strength of the relationship between a dependent variable and other independent variables. In the case of changes in independent variables, it reveals how dependent variables change. In this way, the cause-and-effect relationship between the variables can be estimated. Regression can be used to understand the fit of data to equations.^[13]

2.2.1.1.1 Linear Regression

It examines the linear relationship between two variables and creates a model that allows estimating the value of one variable from the other variable based on this relationship. Although it is often confused with correlation, they are different things. Correlation shows how much two variables are related.^[14] The purpose of using linear regression is to find the most appropriate straight line or hyper equation for a set of points, using the most appropriate regression line, establishes a relationship between the dependent variable and one or more independent variables. The most appropriate regression line is used to decipher the relationship between the dependent variable and one or more independent variables.

2.2.1.1.2 Logistic Regression

Although the word regression is mentioned in its name, it is a classification algorithm. It is the most suitable theorem for binary classification. The biggest difference between linear regression and the line that will decouple the two classes is how the fit is applied. Linear regression uses the Least

Squares method to find the optimal line, while logistic regression uses the Maximum Likelihood method.^[15] The Sigmoid function is used to make the classification, this function is an S-shaped curve. The Sigmoid function is a function that is used to compress data between 0 and 1. The sigmoid function is a function that decrypts data between 0 and 1. Advantages; it is quite easy to implement and interpret and is less prone to overfitting at the same time, it performs very well when the data can be separated linearly. Disadvantages are; if the data set cannot be separated linearly or if the number of observations is less than the number of features, logistic regression cannot be applied to that data set.

2.2.1.2 Naïve Bayes

Bayes' theorem is a computational formula found by Thomas Bayes in 1812, it is used to calculate conditional probability. It is studied in probability theory, this theorem shows the deceleration between conditional probabilities and marginal probabilities for a random variable. A classifier based on the Bayes theory was developed. When calculating the probability of an event occurring, it takes into account the occurrence of an event at the same time.^[16] It is a lazy learning algorithm. One of the biggest advantages is that it can work on unstable data sets and does not need a lot of training data. The way the algorithm works calculates the probability of each situation for an element and classifies the data according to the highest probability value.

2.2.1.3 KNN (K-Nearest Neighbor Algorithm)

Unlike other algorithms, the entire data set is used as a training set, in other algorithms, the data set is divided into training and test sets. KNN estimates the class of the data based on which class the vector formed by the values of the independent variables to be estimated in its data is concentrated to its nearest neighbors.^[17]

The K-Nearest Neighbors algorithm works by using two basic values. These are the distance and the number of neighbors(k). The distance is the distance between the point to be estimated and the other points, it is found by the Minkowski distance calculation method. If the number of neighbors(k) is the number of neighbors with whom the calculation will be made, it has a direct impact on the result. If the value of K is 1, overfitting can occur, if it is too large, very general results will occur, so the value of k must be carefully selected and selected by the algorithm developer, the main problem of the algorithm is to determine the value of K. Three different indicators can be used to measure KNN success.

1-Jaccard Index takes values between 1 and 0, 1 is the most successful stat. It is found by proportioning the intersection set of the correct set of estimates and the real set of values, and the combination set of the correct set and the real set of values.

2-F1-Score is calculated using the confusion matrix. While calculating, precision and recall values are considered. It takes a value between 1 and 0, 1 is the most successful state.

3-LogLoss, like other indicators, takes a value between 1 and 0, but unlike other indicators, 0 indicates the most successful state. It is calculated from the logistic regression result.

2.2.2 Unsupervised Learning

When developing the algorithm, the information used to train the algorithm does not need to be classified or labeled in any way, the algorithm finds out how systems can extract a function to explain a hidden structure from their unlabeled data. If the system does not find the correct output, it continues to search the data and tries to find the correct output by examining the data sets. Unsupervised learning is usually used if the algorithm developer does not have information about the labels of the data to be given to the algorithm. Since the data given to the algorithm are unlabeled,

no definite comments can be made about the accuracy of the resulting output. For example: association(Apriori algorithm), clustering(K-means, hierarchical, KNN)

2.2.2.1 Clustering

Clustering is one of the most important concepts for the unsupervised learning algorithm. It deconstructs the data into clusters or groups by analyzing the similarities and differences between the data. This is done by taking data points in the same group and dividing the data into groups by their similarity and difference with other data points in their group or with data in another data group. There are different clustering algorithms.^[18]

2.3.2.1.1. K-Means Clustering

It is a recursive clustering algorithm. How many clusters the data will be divided into is determined by the k value, the k value is usually determined by the algorithm developer, and the more iterations of the k value, the higher the value will be found.^[18] According to the k value selected by the algorithm developer, the data is combined by the algorithm into k sets. As long as there is a similarity between the combined data, the other data are clustered, and the algorithm ends when the maximum similarity is reached.

2.3.2.1.2 Hierarchical Clustering

The principle of this algorithm is that the data in the data set are examined and similar data are clustered based on the similarity between the data, groups in which similar objects coexist are called clusters. Each cluster is different from the others, but the data within the clusters are highly similar to each other. In other words, a hierarchy has been created for each of the clusters.^[19] Differences between K-means and Hierarch Clustering techniques: k means consumes less memory than hierarchical and usually compute time is $O(n)$, but K-means results are much more sensitive to random initialization than hierarchical. the number of clusters to be formed in the k-means clustering method is predetermined and its use in categorical variables does not give very accurate results. Hierarchical is a deterministic algorithm, unlike k-means, the number of clusters is not predetermined, the fact that the number k is not fixed gives clustering flexibility, but memory also takes up a lot of space according to k-means.

2.3.2.1.3 KNN Clustering

It is one of the most commonly used clustering methods. There are two types of data for K-NN, one of which is data obtained in the past and the other is new data. K-NN decouples by looking at the distance between the past data and the new data. K represents the number of neighbors, and the distance between neighbors up to the number of k is examined, as a result, the data is included in the most expectable class.

2.3.2.2 DBSCAN

Density-Based Spatial Clustering of Applications with Noise. It is often used for data containing clusters of similar density, finds high-density core samples, and expands clusters from these samples. Unlike K-Means, DBSCAN evaluates clusters instead of gaps in density, thanks to which it can successfully detect separated clusters.

2.3.3 Data Analysis

2.3.3.1 PCA

Principal Component Analysis. It is a statistical technique used in the fields of recognition, classification, and image compression. Its main purpose is to keep the data set with the highest variance in high-dimensional data, but in doing so it tries to achieve size reduction, but while reducing the data size at this time, care should be taken that the lost properties are not too

sensitive or important for the data.^[20] As a result of PCA, an artificial set of variables called "basic components" is created, this set includes highly correlated variables. PCA reduces the number of features, but it is not a feature selection method. Its advantages are that it destroys correlated columns, improves algorithm performance and visualization capability, and reduces overfitting. Disadvantages are that there should be a loss of information and data normalization should be performed. It becomes more difficult to interpret the independent variables. Steps; continuous and initial data must be standardized, then the covariance matrix is found to find the correlation. The created covariance matrix contains the eigenvector and eigenvalue, so the principal components are accessed. to decide which principal concepts are useful, a new feature vector is created and the data is recast.

2.3.3.2 EDA

Exploratory Data Analysis. EDA is one of the most critical stages in machine learning. It is usually used to analyze data using visual methods. It provides deciphering the data whole, features, and the relationship between data and features.^[21] EDA briefly extracts the summary of the data set. EDA can be divided into 4 main stages. These are general picture extraction, univariate analysis, bivariate analysis, and multivariate analysis). General image extraction allows for obtaining information about the entire data set. In univariate analysis, information is obtained about features one by one, for example, missing values, outliers, and abnormal values can be given as examples. Bivariate analysis decodes the relationship between features. Multivariate analysis, on the other hand, has correlations between three or more features.

2.3 Mobile Application Development

A mobile application is a type of application software designed specifically to run on mobile devices like smartphones and tablets, and is more commonly referred to as "an app." Similar services to those accessed on PCs are routinely made available to consumers through mobile applications.^[22] Mobile devices have grown in popularity in recent years thanks to their cheaper prices and compact, stylish design. Developers now face additional challenges as a result of the rise in popularity of mobile devices, including how to handle increases in bandwidth and how to adapt old codebases to run on a screen that is ten times smaller than it was intended to.^[23]

2.3.4.1 Java

Java is a network-centric, object-oriented, and multi-platform language of programming. It is a high-security and fast programming language that can be used while developing applications such as mobile and web applications, servers, enterprise software, big data, game development, and artificial intelligence. Java has an API and a virtual machine. Java defines the programming language and its semantics and defines the rules needed to develop a program. Java APIs are pre-written and packaged Java programs, used to perform certain operations.^[24] A Java Virtual Machine (JVM) is a layer between software and machine hardware. It is a java code compiler, all java codes are compiled first in bytecode, and these bytecodes only work in JVM. The JVM interprets the application according to the operating system on which the application is running.

2.3.4.2 Kotlin

It first appeared in 2011. It was statically written by JetBrains company to be fully compatible with Java and was fully accepted by Google in 2017. It is free and open source with an Apache 2.0 license, and can also be used together with JavaScript and in HTML pages. Java and Kotlin can be defined as programming languages that complement each other, the biggest difference and advantage from Java is that the probability of receiving a null warning is close to impossible.^[25] Although it can be used in server and client-side developments, the most widely used area is to develop Android applications.

2.3.4.3 Dart

It is a programming language developed by Google in 2011 and has become standardized by ECMA for the development of mobile applications, web applications, servers, and IoT devices.^[26] It is largely similar to C as a syntax, it can be compiled using machine code or JavaScript, along with supporting features such as type inference, interface, etc. Dart is a programming language that forms the basis of Flutter. Flutter is an SDK (Software Development Kit) developed by Google to help with mobile application development. The biggest advantage of Flutter is that it allows you to develop a single application instead of developing separate applications for both operating systems (Android and iOS).

2.3.4.4 Objective-C

Objective-C is an object-oriented programming language written based on C that began development in 1983 and was continued to be developed by Apple in 2007. It was developed by Brad Cox and Tom Love under the Stepstone company. Its syntax is almost identical to the C programming language, but it does not support custom libraries. Objective-C was the preferred programming language utilized, supported, and mandated by Apple for the development of macOS and iOS applications until the release of Swift in 2014.^[27]

2.3.4.5 Swift

Swift is a multi-paradigm, general-purpose programming language that was created by Apple at the year of 2014. Swift is an open-source, object-oriented programming language used to develop software on Apple platforms.^[27] Swift was created as a replacement for Apple's previous programming language Objective-C, which had been mostly untouched since the early 1980s and lacked modern language capabilities. Swift is preferred more than Objective-C due to its fast, powerful, and understandable. the open-source community. In addition to writing applications for apple products (devices with macOS, iOS, iPadOS, watchOS, and tvOS installed) using Swift, Linux-based applications can also be developed, although they are not very preferred.

2.4 Database

The concept of a database emerged in the 1960s. It is the collection and recording of data or complex information records in a specific area. A database is an area where data that is usually related to each other is stored, it is controlled by a management system and has its management system.^[28] Database Management systems are systems that make the data stored in an electronic environment easily accessible, by using database management systems, tables can be created for any desired information, access restrictions can be placed on these tables, and queries can be made in tables. There are multiple types of databases, they are relational (SQL) databases, object-oriented databases, non-relational(NoSQL) databases, OLTP databases, distributed databases, graphical databases, and cloud databases. The two most popular types of databases are:

2.4.1 SQL

The relational data model served as the foundation for SQL, a database management system that holds a variety of data with varying sizes and purposes. SQL was developed in the 1970s. It is not a programming language, it can be defined as a database sublanguage. It is the basis of database management, and its purpose is to model data and datasets for the job to be done. It allows operations such as data saving, data update, data query or search, and data deletion. It is powerful, ensures data security, is scalable, and has high performance, at the same time it is user-friendly.^[28]

2.4.1.1 MySQL

MySQL is launched in 1995, it is a relational database(RDBMS-Relational Database Management System) that is most popularly used for web design. One of the biggest advantages is that it is free and open source, high-performance, and simple to use. In addition to providing easy management

and availability to its users, it provides scalability, data security, and low cost. MySQL and SQL are different things from each other, they should not be confused. SQL is used for database execution, database query, and database management purposes. MySQL is a database software language.

2.4.2 NoSQL

NoSQL databases came out in the 2000s, they are a non-table structure and non-relational database type, which is not suitable for relational models, it is preferred for processing big data. Stores data as a key/value pair in a schemeless way. Calls are made with key values. It is used to store data in columns and use it in analytical applications.^[28]

2.4.2.1 *Firebase*

Firebase developed by Google provides login authorization and data storage in a real-time/synchronous way for the programmer's web and mobile applications to be run on the server side. Firebase is a real-time database, real-time database provides connection via WebSocket instead of HTTP, and the database is synchronized simultaneously in case of a data update in the application. Firebase can also be used for file storage purposes, allows controlling the data requested to be uploaded to the server, and provides data security. One of the most important features of Firebase is that it provides authentication.

2.4.2.2 *MongoDB*

MongoDB is developed in 2009 as, NoSQL, an open-source database. It is among the most popular NoSQL databases. All data and records are stored as documents in JSON format. The table structure is called a collection, the row structure is called a document, and the column structure is called a field. It is based on a document, key/value, graph, and column. In general, it is used in applications where performance is important. It can create multiple copies of the original data and thus prevents data loss. As well as allowing the processing of big data, it can also perform collection operations and provides driver support for most of the current programming languages.

2.4.3 Differences Between SQL and NoSQL

While SQL database is used in the relational and structured query language, NoSQL database is not relational and has dynamic schemas for structured data. SQL databases are used vertically, as tables, and in multi-row operations, while NoSQL databases are used horizontally to perform operations with documents, keys/values, graphs, or large columns and documents.^[29] The advantages of NoSQL are that it processes data at high speed, can store various types of data, and easily updates schemas and fields. The disadvantages are that standardization is not possible, and database backup and consistency are low. The advantages of SQL are that it is easier to use than NoSQL, it is secure, scalable, and at the same time, they are versatile.

2.6 Related Work Overview

2.6.1 Google Maps

Navigation is the first thing that comes to mind when it comes to route planning, Google Maps is an indispensable part of our travels. Features such as adding your destination to the favorites on the application, and adding multiple stops to your route are our favorite aspects. You can also use Google Maps as an offline map. If there is no internet at your destination, you download the map of this place where there is internet and you can use this map offline while you are there.^[29] Google Maps is excellent, but there are no downsides. Sometimes he can take you off the beautiful route and take you to the mountain, to the forest. This can also allow you to encounter unusual things. At this point, it is suggested to look at the general route in general while saying the route is OK. And sometimes he tries to change the route itself, so don't accidentally choose and take a different route. Finally, you can look at the photos and comments of the places you

will go on the application. Especially in the comments, you can find very useful things. You can also sort comments by date. Google Maps is the most important application to have on a phone. Apple and Yandex maps are also other alternatives. As stated, the application is good, but you should not completely give up control over the route, especially when it comes to the application. When we created our project we were inspired by Google Maps. However, we wanted to design our application by developing Google maps in slightly different directions. That's why Google maps, is one of the applications that most fit our original logic. We will talk about the extra features we have added here; First, user comments, route determination, an indication of how long it will take to reach this route, and an indication of which vehicles can be used in our application, as well as in Google Maps. We have added interaction with platforms such as Facebook and Instagram to increase socialization to have socialization. Additionally, by selecting a specific range, the display of sights and historical monuments at a distance will again be selected. However, here Google maps only show all petrol offices, shopping malls, etc., including all the places in the region.

2.6.2 Travel Shop Turkey

Three alternatives are presented here, aiming for a straightforward cultural trip without confusing people too much. Firstly, tours are being organized to historical monuments located in various regions of Turkey (at a distance that is more well-known) and places where there is a lot of interest. Secondly, cabin trips are provided at the designated locations separately from the tours. Thirdly, it determines the routes that can be traveled by high-speed train in advance and then leaves it to the preference of passengers who want to attend these routes.^[30] The difference from our project is; since we do not want a situation where people will be confused here, we thought of making an application so that they can easily make their preferences and not be undecided about issues such as where what is there or which places are more popular. Thus, after the application users have determined the desired range, they will be informed about many more places such as restaurants, historical places, and entertainment venues located within that range, and will also make recommendations from them, including the closest to the user's age and favorite things. Additionally, it can be linked to applications such as Facebook and Instagram to increase sociability. Finally, we have designed the application interface as convenient and understandable as possible for users to be satisfied with the application.

2.6.3 City Mapper

City Mapper is a must-have application for city transportation. The application, which has a very large database in which all vehicles used in public transportation are connected and synchronized, also cooperates with many other applications such as Google, Foursquare, and Uber. It both takes you from point A to point B with the directions it offers and supports the processes from places to your means of transportation. You can find out your journey time by vehicle or how far you will walk, what alternatives there are by bus or metro on the route, and their fares. There are both android and iOS-supported versions. In our project, it is progressing in the same process, only different, it can be thought of as a slightly higher model. While city mapper provides access only to certain entered points, many places are determined according to the range chosen in the application, depending on the person. Then it is ranked according to its popularity.

2.6.4 LiveTrekker

The purpose of the LiveTrekker application is to record the route you took during your trip, then show where you can go for how long on the interactive map and share it on social networks. You can also create a travel diary by uploading photos, notes, audio recordings, and video content to the application. The application we will develop is one of the most similar applications among the applications. The route is determined by travel time, similarly, it is used

in an integrated way with platforms such as Instagram and Facebook in case you want to share it on social networks. In the application mentioned here, many features such as photos, notes, voice recordings, etc. Can be uploaded inside the application.

2.6.5 Time Out

Time Out, an application that lists restaurants, cafes, and entertainment venues located on the route you designate, also calculates the distance by showing the address information of the listed places on the map. In addition, you can easily make plans with the Time Out application, which also shows current cultural, art, and music events. This application is one of the closest applications to the application we will make. However, unlike our application, it can interact with platforms such as Instagram and Facebook to ensure sociability. However, although determining places via a specific route is similar to its features, such as showing the popularity of these places, addition, our application also determines historical places.

2.6.6 Zomato

In a city where you are going for the first time, you don't know where to eat and how much to pay. Zomato, which comes to the rescue of users in this regard, saves you from a big burden by listing the businesses closest to your location, their concepts, menus, and other user experiences. Which also offers the ability to make reservations, and stands out as a very useful application. On the other hand, we see that this application answers more questions such as what is on the menu and where we should I eat, but our application it is a little different in this regard. There is no routing based on the location of the user in this application. And also, Zomato doesn't integrate with any type of social networking application.

2.5 Conclusion

Our goal in this project is to plan and develop the popular recommendation that systems examine our positive aspects are included in the practice, while other applications on the negative aspects and think about how we can fix it based on user feedback, and develop software for developing the negative aspects. Especially to provide personalization, unlike other recommendation systems. While developing our application, we will use hybrid filtering, and explicit feedback. As a programming language, we will use flutter for Android devices, and Swift for Apple devices. For the database of our mobile application, we will use a NoSQL database, Firebase.

3. Software Requirements Specification (SRS)

3.1 Introduction

A recommendation system for travelers is a mobile application that learns the location of the user, is located in the area between the distance values entered by the user, shows and informs the location of the place or places that the travelers want to go, and if desired, guides the user to the place accepted in the system. This system includes Live Location Tracking and Navigator. Additionally, the system uses Google API.^{[29] [31]}

3.1.1 Purpose

The purpose of the document is to give a detailed description of the requirements for the “Recommendation System for Travelers” project. This application aims that users who will use the application called “Travelers” can easily find the places they can visit, share and evaluate from the people who are interested in traveling. This document describes the requirement of the project in detail.

3.1.2 Scope of Project

“Recommendation system for Travelers” is a GPS-Based mobile application that helps users to find and discover new historical places or natural areas based on the user's current GPS location and recommending places to go. There are other features such as the application drawing a route for the specified location and users can upload and share photos to the system. This information will form the basis for the recommended results displayed to the user. The administrator also uses this application to manage the system and limit inappropriate or prohibited posts by the social platform and delete users from the platform when necessary. In addition, if there is a malfunction in the application structure, the administrator can directly handle and manage the situation. System information is stored in a web-based data store. System information is stored in a web-based data store. The stored data is filtered and transferred to the user interface and used. In addition, the GPS of the device used by the user must be turned on for the user's instant location to be known by the system. While the GPS shows the instant location, the system suggests and shows the details of the nearby areas by taking the instant location data from the user according to the radius range determined by the user.

3.1.3 Glossary

- API: API stands for Application Programming Interface.
- API Key: This is the key required to use the API.
- Geolocation: The process or technique of identifying the geographical location of a person or device using digital information processed via the internet.
- Machine Learning: ML is a subset of AI (artificial intelligence) that focuses on creating systems based on data they have learned.^[10]
- User: A person who uses or operates something.
- iOS: A mobile operating system created by Apple.
- Android: A mobile operating system created by America; Google.
- Flutter: Flutter is an Open-Source UI SDK developed by Google.^[32]
- Dart: Dart is an Open-Source, client-side programming language.^[26]
- Swift: A programming language created by Apple to develop applications for iOS, Mac, Apple TV, and Apple Watch.^[27]
- SwiftUI: SwiftUI is a framework designed to be used with Swift.
- Emulator: An emulator is a special software that allows you to use computers like a tablet or a phone.
- Provider: It provides that the created components can be accessed by the contexts and that the rendering process is performed again when these contexts are updated.
- Firebase: Provides detailed documentation and cross-platform SDKs to help you build and ship apps on Android and iOS.^[28]

3.2 Overall Description

3.2.1 Product Perspective

Recommendation System for Travelers project that has the purpose of recommending places to visit such as historical places, natural environments, etc. This section will give an overview of the system. The system is made to specify and organize the details to be done in the project.

3.2.1.1 Development Methodology

For developing the project, we have planned to use the Waterfall software development methodology, as known as the Waterfall model.^[33] The waterfall model uses a logical progression, similar to the direction water flows over the edge of a cliff. We planned to divide the project into equal parts in our group and to unite it after everyone did the task given in their field according to the determined requirements.

3.2.2 User Characteristics

Two types of users interact with the system: admin and user.

3.2.2.1 User

- Users must have internet access and any mobile device to use the application.
- The user must activate GPS for giving the system to the current location.

3.2.2.2 Admin

- Admin should know the system well and be able to intervene when necessary.
- Admin must know how to use a computer and phone.
- Admins must have internet access and any mobile device to use the application.

3.3 Requirements Specification

3.3.1 External Interface Requirements

3.3.1.1 User Interface

The user interface must be easy-to-use and clear because the app's target users are general users, and for that reason, the possible errors should be minimized. In case of an error, error messages should be clear and understandable. The user interface should be designed in such a way that user can clearly show their current location on the map and the places they want to go or are recommended by the app. The target user of this app are people who are indecisive about where to see when traveling.

Flexibility and efficiency, Aesthetic and minimalist design, Help and documentation, as well as a design for the user to better use the given design, are being applied.

3.3.1.2 Hardware Interfaces

The project is a mobile-based application. The client-side application will be developed as an Android and iOS application. To use the application user must have an Android or iOS-based operating system device. Since the application will work on the map, location permission must be given from the d used. Also, if the user wants to add a photo of the places they visited, camera and gallery access are needed.

3.3.1.3 Software Interfaces

To develop the application, Flutter is the main programming language for Android devices, and for iOS devices, Swift programming language will be used. The application will support Android and iOS. To develop this application, Google's database Google API will be used. Also, we are using GPS for location services.

3.3.1.4 Communications Interfaces

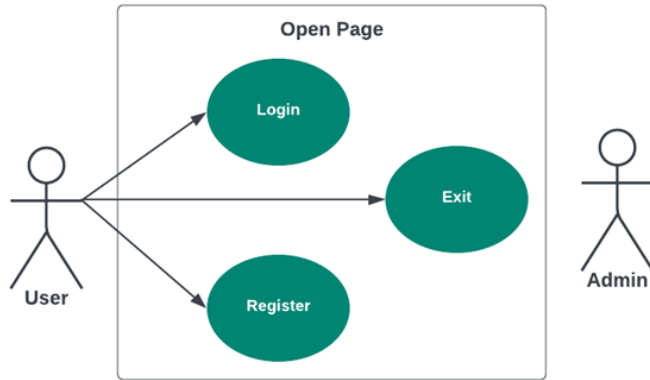
To send the user's instant location to the system, the user's internet and GPS must be enabled.

3.3.2 Functional Requirements

3.3.2.1 Open Page

- Login
- Register
- Exit

Use Case Diagram:



Use Case Name: Open Page

Use Case Number: UC-1

Actors: User

Overview:

Users will use this use case to log in to the page.

Pre-Conditions:

- An Internet connection is required to open the page.

Post-Conditions:

- Users must log in to use the app.

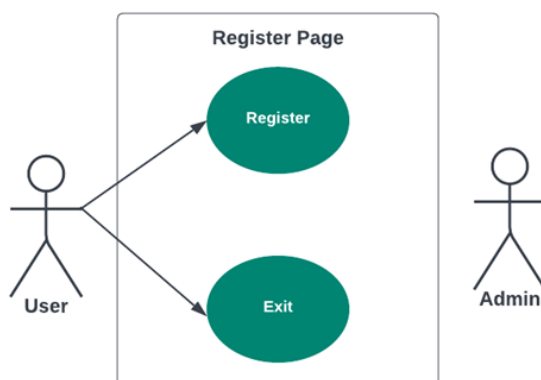
Flow Description:

- The user clicks the “Register” button to register to the application.
- When the user clicks the “Login” button, they are redirected to the login screen.

3.3.2.2 Register Page

- Register
- Exit

Use Case Diagram:



Use Case Name: Register Page

Use Case Number: UC – 2

Actors: User

Overview:

In this use case, users register to the system.

Related Use Cases: UC – 1

Pre-Conditions:

- Users must log in to the register page.
- Users should not be registered in the system before.

Post-Conditions:

- When users fill in the required information correctly, they are successfully registered in the system.

Flow Description:

- The system expects users to enter their “Username”, “Email” and “Password” correctly.
- The system checks whether the users have this information in the database.
- If there is an error, it is indicated on the screen. Re-enters user information.

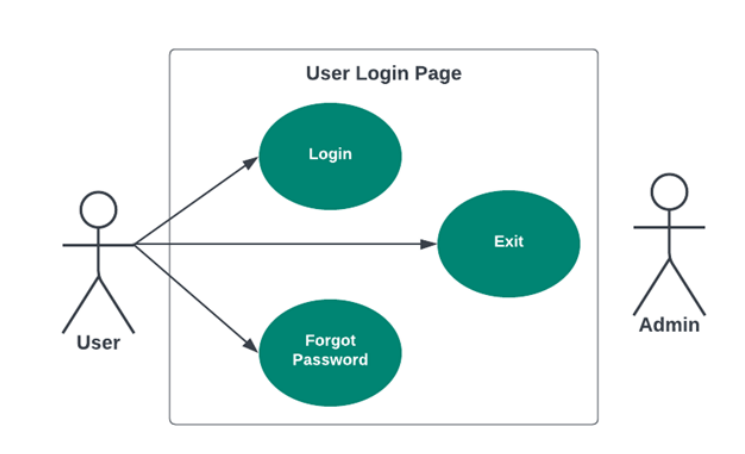
Alternative Flow Description

- If the user has registered in the system before, the application gives a warning and is redirected to the login page.

3.3.2.3 User Login Page

- Login
- Forgot My Password
- Exit

Use Case Diagram:



Use Case Name: User Login Page

Use Case Number: UC – 3

Actors: User

Overview:

In this use case, users log into the system.

Related Use Cases: UC – 1

Pre-Conditions:

- Users must enter user information correctly.

Post-Conditions:

- If the user has entered the correct “Username” and “Password”, she/he logs into the system.

Flow Description:

- Users fill in the username and password section on the screen after pressing the login button.
- The system checks the information entered by the users.
- If there is an error, it is indicated on the screen. Re-enters user information.

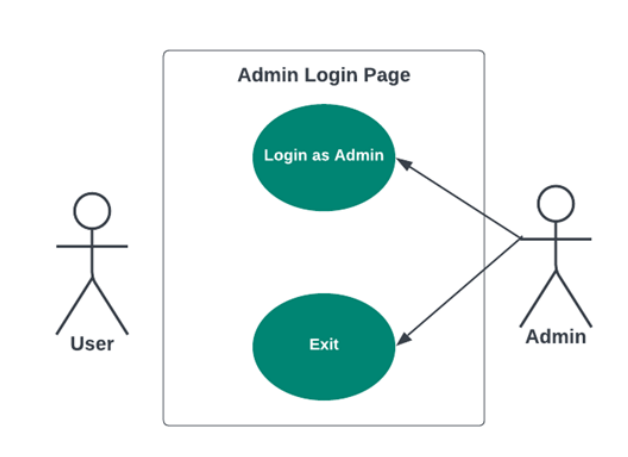
Alternative Flow Description:

- If the user has forgotten his password, click the “Forgot My Password” button.
- The system prompts the user to enter the e-mail registered to the system.
- The system sends a password change mail to the user’s e-mail.
- The user can reset his password in line with the incoming mail.

3.3.2.4 Admin Login Page

- Login as Admin
- Exit

Use Case Diagram:



Use Case Name: Admin Login Page

Actors: Admin

Overview:

In this use case, admins log into the system.

Related Use Cases: UC-1

Pre-Conditions: -Admins must have an internet connection.

Post-Conditions: -If the admin information is entered correctly, it logs into the admin page.

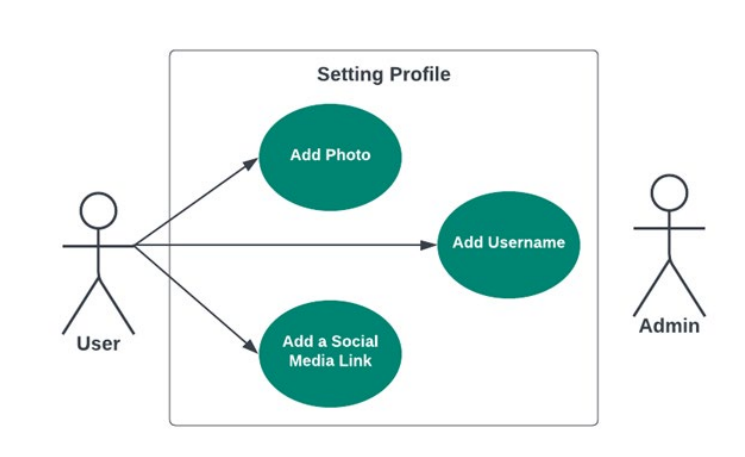
Flow Description:

- Admin must click “Login as Admin” on the login screen
- Admin enters “Username” and “Password” correctly.
- If there is an error, it is indicated on the screen. Re-enters admin information.

3.3.2.5 Setting Profile

- Add Photo
- Add Username
- Add a Social Media Link

Use Case Diagram:



Use Case Name: Setting Profile

Use Case Number: UC-5

Actors: User

Overview:

In this use case, users are redirected to the “Settings Profile” page, where users can share their photos, add a nickname for themselves and share links to other social media platforms.

Related Use Case: UC-1, UC-3

Pre-Conditions:

- Users must be registered in the system.

Post-Conditions:

- When the users click the setting profile button, they are directed to the setting profile page.

Flow Description:

- User presses the setting profile button.
- If the user wants to add a photo, he/she clicks the “add photo” button.
- If the user wants to change their username, they click the “change username” button.
- When the users make changes to their profile or add something new, they should press the save button.

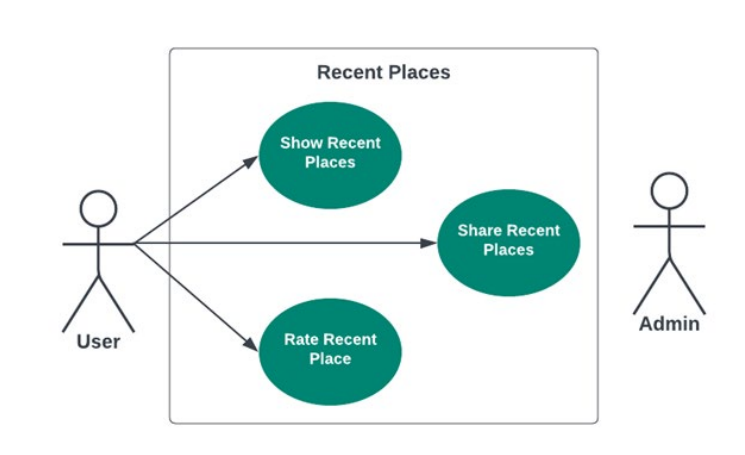
Alternative Flow Description

- If the user does not press the “save” button on the settings they have changed, the system will continue to use the user’s default settings.

3.3.2.6 Recent Places

- Show Recent Places
- Share Recent Place
- Rate Recent Place

Use Case Diagram:



Use Case Name: Recent Places

Use Case Number: UC-6

Actors: User, System

Overview:

In this use case, users can see the recent places visited by other users or themselves.

Related Use Cases: UC-1, UC-3

Pre-Conditions:

- Users must be registered in the system.

- Users must have an internet connection.
- Users must have the GPS turned on, on their phones.

Post-Conditions:

- When the users click “Recent Places”, they can see the last places they have visited of the user.
- When the users click “Share Recent Place”, they can share that the place they have visited.
- When the user clicks “Rate Recent Place”, the user can rate the place.

Flow Description:

- The user should enter their own or another user’s profile and click the “Recent Places” button.
- The user can like the recent places he/she has been to by pressing the like button under the photo or location.
- The user can write his/her comment in the comment box about this place and share it with the share button.
- If users want, they can stop sharing with the close “Share Recent Places” button in the profile settings section.

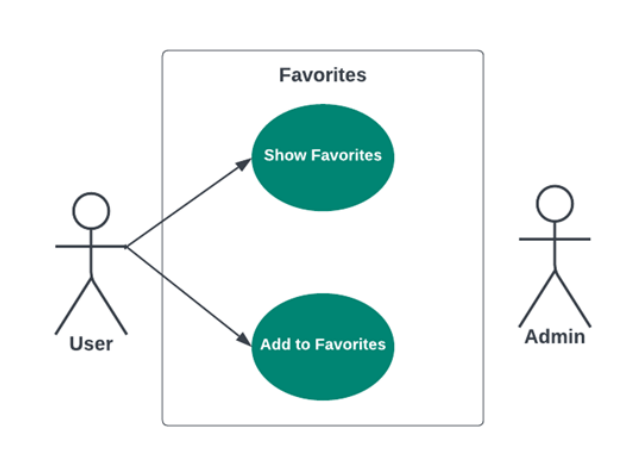
Alternative Flow Description:

- Users cannot see the last visited GPS feature and the internet connection on their phones is not turned on.

3.3.2.7 Favorites

- Show Favorites
- Add to Favorites

Use Case Diagram:



Use Case Name: Favorites

Use Case Number: UC-7

Actors: User

Overview:

In this use case, users can create and share collages on the “Add Favorites to Visit” page.

Related Use Cases: UC-1, UC-3

Pre-Conditions:

- Users must be registered in the system.
- Users must have an internet connection.
- Users must have the GPS turned on, on their phones.

Post-Conditions:

- When the users click “My Favorite Places”, they can see the added favorite places.

Flow Description:

- Users should click on the “Recent Places” button.
- Users should click on the heart icon above the place they like.
- When the user clicks on the heart symbol, the system automatically creates a “My Favorites Places” folder in the user’s profile.

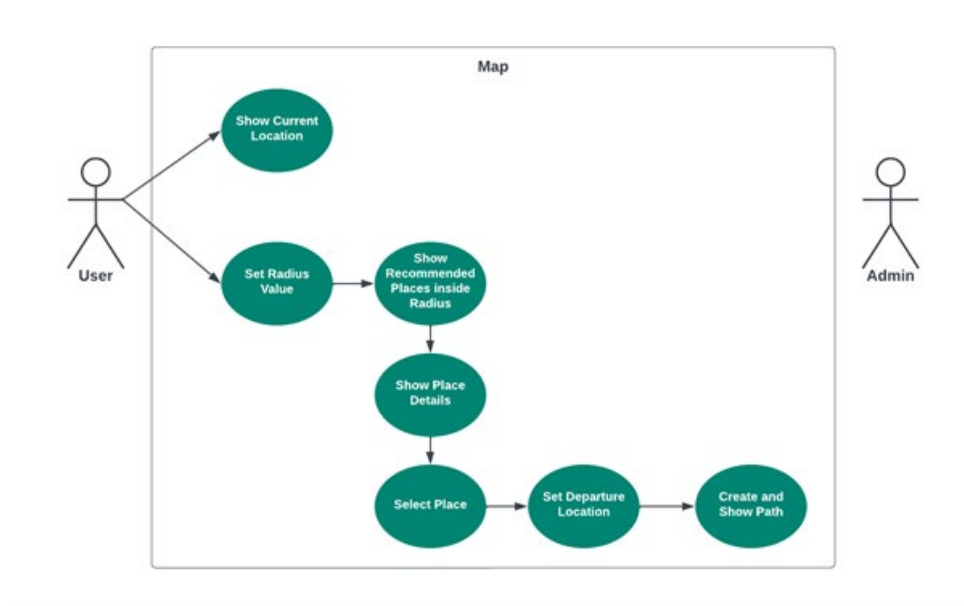
Alternative Flow Description:

- Users cannot see the last visited GPS feature and the internet connection on their phones is not turned on.

3.3.2.8 *Map*

- Current Location
- Set Radius Value
- Show Recommended Places Inside Radius
- Show Place Details
- Select Place
- Set Departure Location
- Create and Show Path

Use Case Diagram:



Use Case Name: Map

Use Case Number: UC-8

Actors: User

Overview: In this use case, after logging into the application, users can find their location on the map on the main screen with the GPS feature on their phones. The user can set how many kilometers in diameter they want to see places around them. The system recommends places to users according to the places they visit. The system creates the route of the place where the user wants to go on the map used by the application most shortly.

Related Use Cases: UC-1, UC-3

Pre-Conditions:

- Users must be registered in the system.
- Users must have an internet connection.
- Users must have the GPS turned on, on their phones.

Post-Conditions:

- When the user clicks on the “show my current location” button on the map, it shows the current location and its surroundings.
- When the user clicks on the “Set Radius Value” button on the map, they enter the desired value, and the surrounding places are shown according to the entered value.
- The system of the application learns by machine learning by keeping the places visited by the user in its database.
- The system sends warning messages for the location of similar places on the map used by the application, according to the type of places the user visits the most.
- The user should select the place from the list where s/he wants to go on the map used by the application and click the “Create Path” button.

Flow Description:

- After the user logs into the application, the map appears on the main screen.
- The users press the “show my location” button on the map to find their location on the map.
- After pressing the button, the users can see their location and surroundings on the map.
- The user chooses settings and “Set Radius Value”, and enters the desired Radius value.
- The application recommends the location of the places s/he may like to the user in the form of a list.
- After the user selects a place, they also have to select a departure location from the map or they can choose their current location as the departure location.
- When the locations are entered correctly the “Create Path” button should be clicked.
- After pressing the button, the system creates the shortest route from the user’s current location to the location they want to go and displays it on the map.
- When the user reaches the place s/he wants to go, the message that s/he has reached that place.

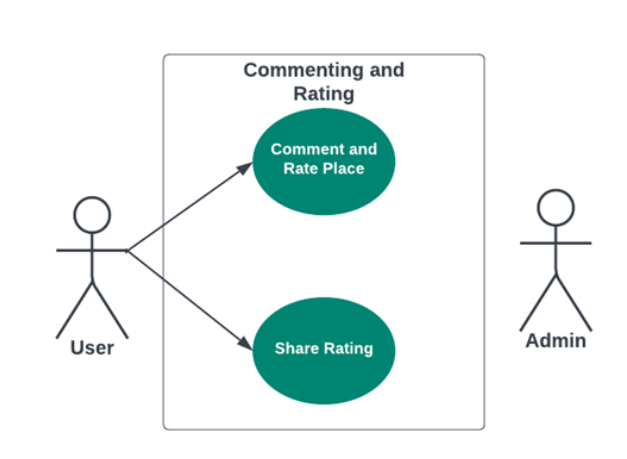
Alternative Flow Description:

- If the user does not have an internet connection, they cannot see their location on the map.
- If the GPS feature of the users is not turned on, they cannot see their location.
- If there is no place to show within the selected radius, a warning message is shown to the user to select a wider radius.

3.3.2.9 Commenting and Rating

- Comment & Rate
- Share Rating

Use Case Diagram:



Use Case Name: Commenting and Rating

Use Case Number: UC-9

Actors: User

Overview:

In this use case, the user sets how many kilometers in diameter they want to see places around them.

Related Use Cases: UC-1, UC-3

Pre-Conditions:

- Users must be registered in the system
- Users must have an internet connection.
- GPS service must be turned on.
- Users must have gone to the place to comment and rating.

Post-Conditions:

- If was in location update database, refresh is called and exit activity.
- If it wasn't in location send an error message and exit activity.

Flow Description:

- If the user wants to comment and rate the place s/he went, s/he should press the “Comment & Rate” button.
- After pressing the button, the user can write a comment in the box that appears.
- The user can rate the place one to five stars.
- When the user presses the “Publish” button, their comment and rating are published so that everyone can see them.

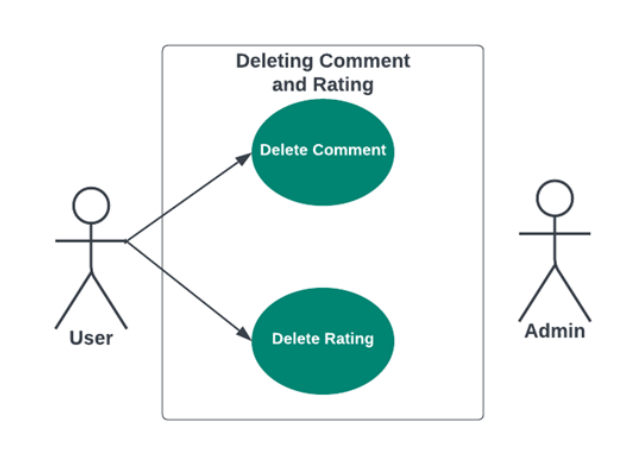
Alternative Flow Description:

- If there is no place to show within the selected radius, a warning message is shown to the user to select a wider radius.

3.3.2.10 Deleting Comments and Ratings

- Delete Comment
- Delete Rating

Use Case Diagram:



Use Case Name: Deleting Comments and Ratings

Use Case Number: UC-9

Actors: User

Overview:

In this use case, after the user opens the application and logs in, the GPS service activation warning message appears on the system screen.

Related Use Cases: UC-1, UC-3, UC-8

Pre-Conditions:

- The user must be registered in the system.
- User must have an internet connection
- The user must have previously commented on that place or rated that place.

Post-Conditions:

- When the user presses the "Delete Comment" or "Delete Rating" button, the selected content will be deleted.

Flow Description:

- User clicks the "Delete Comment" or "Delete Rating" button.
- A message is shown to the user asking, "Are you sure you want to delete the content".
- The selected content will be deleted if the user confirms their choice.

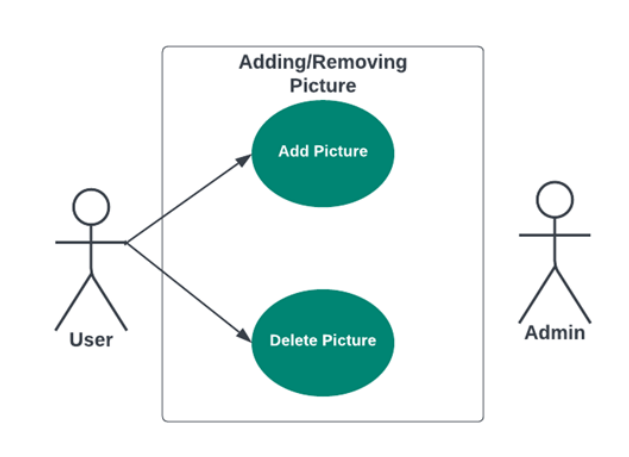
Alternative Flow Description:

- If the user's comment has been deleted by the admin, a message should be sent to the user stating this.

3.3.2.11 Adding/Removing Picture

- Add a Picture of the Place
- Delete Picture of the Place

Use Case Diagram:



Use Case Name: Adding/Removing Picture

Use Case Number: UC-11

Actors: Users

Overview:

In this use case, the user adds a photo to the location they have visited.

Related Use Cases: UC-1, UC-3, UC-7

Pre-Conditions:

- The user must be logged in to the system.
- The user must have an internet connection.
- The user must allow camera access.
- The user must have visited that place before adding a photo.

Post-Conditions:

- When the user clicks on the "Add Photo" button, the user can add the photo they have taken.
- If the user clicks "Add Photo", the selected photo will be uploaded and can be seen on the app.
- If the user clicks "Delete Photo", the selected photo will be deleted from the app.

Flow Description:

- User clicks the "Add New Photo" button.
- User either picks the photo from their gallery or they can take the picture from their camera.
- User uploads the photo to the system.
- The user clicks the "Delete Photo" button, and pictures get deleted from the application.

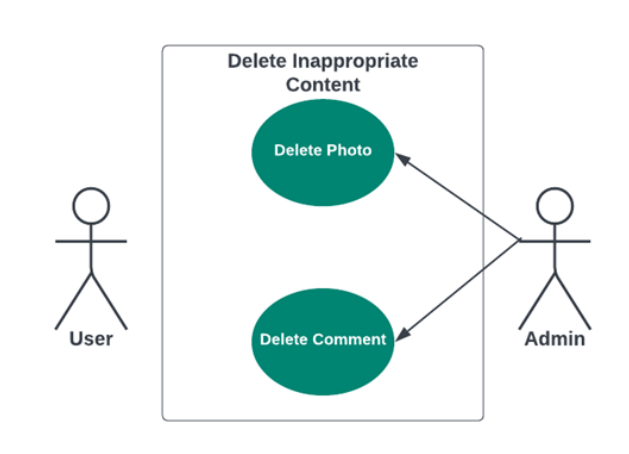
Alternative Flow Description:

- If the photo cannot be uploaded due to internet connection problems or the resolution of the photo an error message will appear on the screen.

3.3.2.12 Delete Inappropriate Content

- Deleting inappropriate Content

Use Case Diagram:



Use Case Number: UC-12

Actors: Admin

Overview:

In this use case, the admin can delete inappropriate content that may disturb other users.

Related Use Cases: UC-1, UC-4

Pre-Conditions:

- Admin must be logged into the system.
- Admin must have an internet connection.
- When the admin clicks on the "Delete Comment" or "Delete Photo" button, the chosen content will be deleted.

Post-Condition:

- If the admin clicks the button, the selected comment or photo will be deleted, and the user will be added to the blacklist.

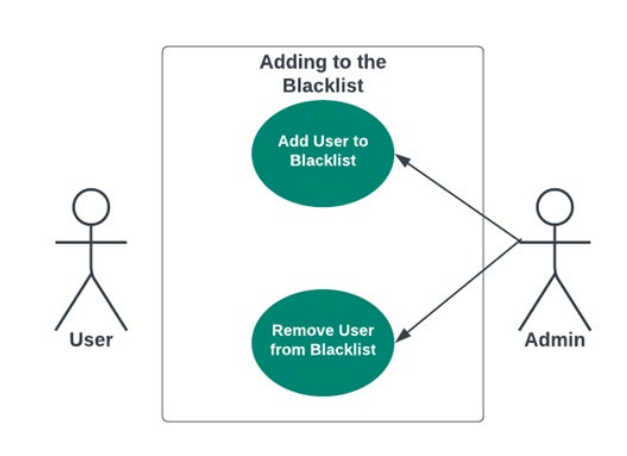
Flow Description:

- Admin clicks "Delete Comment/Photo"
- The selected comment or photo will be deleted from the system.
- The user will be added to the blacklist. Alternative Flow Description:
- If the user has already been blacklisted, the user's account will be deleted.

3.3.2.13 Blacklist User

- Add the user to the Blacklist
- Remove the user from Blacklist

Use Case Diagram:



Use Case Number: UC-13

Actors: Admin

Overview:

In this case, Admin can delete users if the user makes an inappropriate action on the system.

Related Use Cases: UC-1, UC-4

Pre-Conditions:

- Admin must be logged into the system.
- Admin must have an internet connection.
- The admin must have previously deleted the user's inappropriate content, or the user must have been reported to the admin.

Post-Condition:

- If the admin clicks the "Blacklist User" button, the user's account will be suspended.

Flow Description:

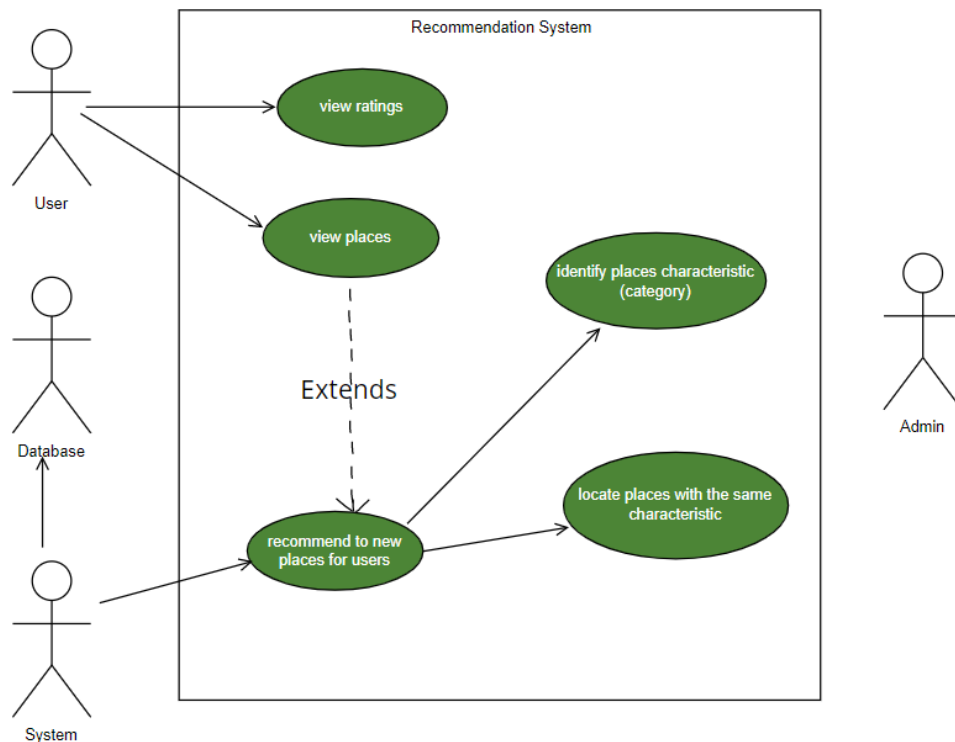
- Admin clicks the "Blacklist User" button.
- The admin specifies why the user's account wants to be blacklisted.
- The user's account is suspended.

Alternative Flow Description:

- If the user's account has already been suspended, the admin will be shown an error message indicating this.

3.2.14 Recommendation System

- Recommending the most suitable places to users.



Use Case Number: UC-14

Actors: User, System

Overview:

In this case, Users provide ratings and comments for the places they have visited, the system utilizes this feedback to generate personalized recommendations from its extensive database. These recommendations are tailored to the users' preferences based on the ratings and comments provided by other users.

Pre-Conditions:

- Admin must be logged into the system.
- Admin must have an internet connection.
- The user must have gone places to give compliments and ratings.

Post-Condition:

- If the user clicks the rating with stars button, the user will be recommend places are designed.

Flow Description:

- User clicks the rating with stars button.
- The system receives the high ratings and positive comments the user gives to certain places and classifications/categorizes the places according to these choices.

- The system will suggest different places for users, but these places will be similar to places with high ratings and positive comments from users' previous visits.

Alternative Flow Description:

- If the user is already making comments and ratings for the places they went. The recommendation system will start working for that user and the most appropriate selections will be shown to the user.

3.3.3 Software System Attributes

3.3.3.1 Portability

The system will be a mobile application, it will be able to be used on all mobile devices.

3.3.3.2 Usability

This application is user-friendly.

3.3.3.3 Reliability

Reliability is one of the key attributes of the system. Back-ups will be made regularly so that restoration with minimal data loss is possible in the event of unforeseen events. The system will also be thoroughly tested by all team members to ensure reliability.

3.3.3.4 Availability

This application will be available on Google Play Store and also AppStore.

3.3.3.5 Security

The system shall be designed with a level of security appropriate for the sensitivity of information enclosed in the database. More interaction is needed with the client about the volatility of the information. The only requirements that could be implemented are encrypting the database and/or making the database password-protected, by the user's request.

4. Software Design Document (SDD)

4.1 Introduction

A recommendation system for travelers is a mobile application that learns the location of the user, is located in the area between the distance values entered by the user, shows and informs the location of the place or places that the travelers want to go, and if desired, guides the user to the place accepted in the system. This system includes Live Location Tracking and Navigator.

4.1.1 Purpose

The purpose of this document is to determine the requirements of the project to be carried out, to determine and explain the internal and external design used in it, to define and design the layout of the system before the project has been completed.

4.1.2 Overview

In the introduction, information about the purpose of use of our product, what it contains, and its design is given. In the Approach section, the approach we will use while designing the product is emphasized. In the System Design section, the current class diagram, sequence diagram, and flowcharts of the product are designed.

4.1.3 Definitions and Acronyms Abbreviations

- Android: A mobile operating system based on a modified version of the Linux kernel and other open-source software.

- iOS: A mobile operating system created and developed by Apple.
- User: Any user character that is yet to register to the system.
- Administrator: The user character that supervised the system and a registered administration user.
- Firebase: Offers comprehensive documentation and cross-platform SDKs to make it easier for developers to create and publish apps for iOS and Android.^[28]
- GPS: The Global Positioning System is a satellite-based radio navigation system owned by the US government.^[34]
- Geolocation: The process or technique of identifying the geographical location of a person or device by means of digital information processed via the internet.
- Flutter: Mobile app development platform created by Google.^[32]
- Dart: Client-optimized language for developing fast apps on any platform.^[32]
- Swift- A programming language created by Apple to develop applications for iOS, Mac, Apple TV and Apple Watch.^[27]
- UI: UI design stands for "user interface".^[35]
- UX: UX design stands for "user experience".^[35]

4.2 Architecture Design

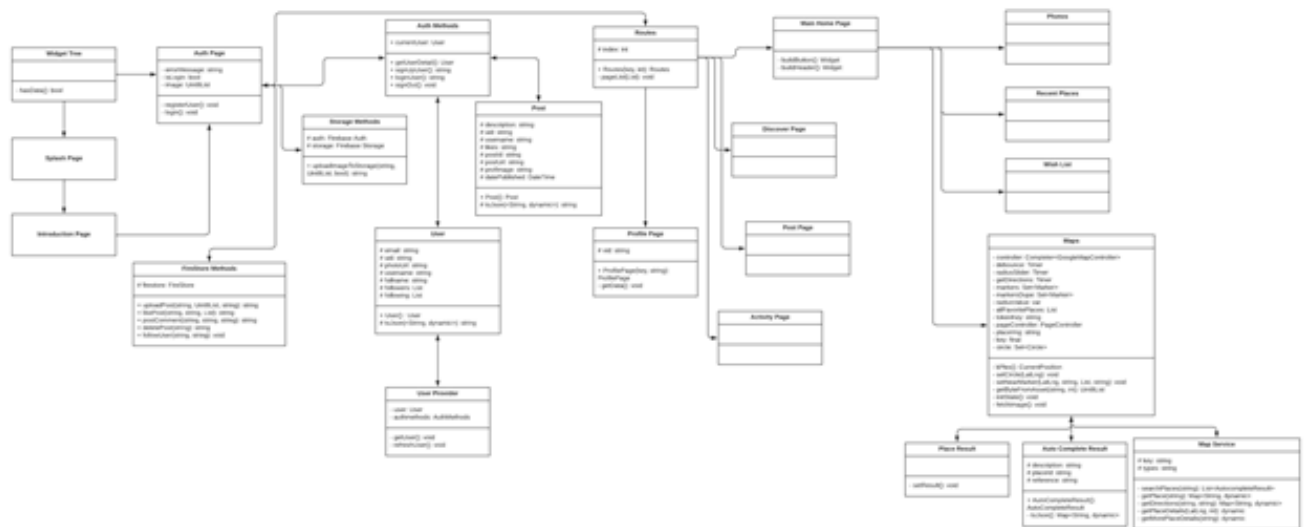
4.2.1 Approach

'Recommendation system for travelers'; It is a GPS based mobile application that helps the user find and explore new historical places or natural areas and suggest places to go based on their current GPS location. It also has other features such as the application that draws a route for the determined location and the ability for users to upload and share photos to the system. This information will form the basis of the recommended results displayed to the user. The administrator also uses this application to manage the system and limit inappropriate or prohibited posts by the social platform and delete users from the platform when necessary. In addition, the administrator can add and delete places and update places informations. In addition, in case of a malfunction in the application structure, the administrator can directly handle and manage the situation. System information is stored in a cloud-based data warehouse. The stored data is filtered and transferred to the user interface and used. In addition, as a result of logging into the system, it is possible for the user to write comments and evaluate the places he/she has actually visited. In addition, the GPS of the device used by the user must be turned on in order for the user's instant location to be known by the system. While the GPS shows the instant location, the system suggests and shows the details of the nearby areas by taking the instant location data from the user according to the radius range determined by the user.

Briefly:

- Admin can view/modify/delete User, upload places information.
- Admin can view all feedback posted by customers, and identify negative comments and users to be restricted.
- The user can log in to the system after entering the registration information.
- Users can search places by rating and place type.
- When the user is going somewhere, GPS must be turned on.
- The user can evaluate and write a comment about that place according to where he went.
- The user can post comments and reviews about a particular place. It can also provide photo sharing related to that place.

4.2.2.1 *Android*



4.2.2.1.1 User

User
<pre># email: string # uid: string # photoUrl: string # username: string # fullname: string # followers: List # following: List</pre>
<pre>+ User() : User # toJson(<String, dynamic>): string</pre>

Fields:

- email: E-mail of the user.
- uid: ID of the user.
- photoUrl: Url of the profile picture of the user.
- username: Username of the user.
- fullname: Name of the user.
- followers: Followers of the user.
- following: List of other users the user follows.

Functions:

- `User()`: Constructor of the User class.
- `toJson()`: Converts json to list.

4.2.2.1.2 User Provider

User Provider
- user: User - authmethods: AuthMethods
- getUser(): void - refreshUser(): void

Fields:

- user: Userprovider gets all the user data to there.
- authMethods: Authenticator checks the user data.

Functions:

- getUser(): This function is used for getting a user.
- refreshUser(): This function is used to refresh the user.

4.2.2.1.3 Post

Post
description: string # uid: string # username: string # likes: string # postId: string # postUrl: string # profImage: string # datePublished: DateTime
+ Post(): Post # toJson(<String, dynamic>): string

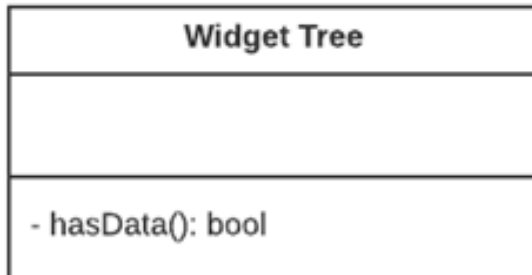
Fields:

- description: Post's description.
- uid: The ID of the user who shared the post.
- username: The username of the user who shared the post.
- likes: Number of likes of the post.
- postId: ID of the post.
- postUrl: Url of the post.
- profImage: The profile picture of the user who shared the post.
- datePublished: Date of the publication of the post.

Functions:

- Post(): Constructor of the Post class.
- toJson(): Converts Json file to the list for posts.

4.2.2.1.4 Widget Tree



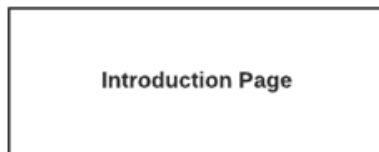
Functions:

- hasData(): If the user is entered the application.

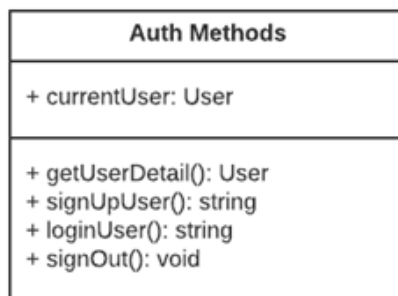
4.2.2.1.5 Splash Page



4.2.2.1.6 Introduction Page



4.2.2.1.7 Auth Methods



Fields:

- currentUser: The user object of the user who has logged in to the application.

Functions:

- getUserDetail(): This function is used for getting details of the user who has logged in to the application.
- signUpUser(): This function is used when a user registers to the system.
- loginUser(): This function is used when the user logs in to the system.
- signOut(): This function is used when the user logs out of the system.

4.2.2.1.8 Auth Page

Auth Page
- errorMessage: string - isLogin: bool - image: Uint8List
- registerUser(): void - login(): void

Fields:

- errorMessage: Error message displayed in case of an error during authentication.
- isLogin: The value that holds whether the user is logged in to the application or not.
- image: It gets the image input from the user and sends to the firebase.

Functions:

- registerUser(): This function is used when a user registers to the system.
- login(): This function is used for when the user logs in to the system.

4.2.2.1.9 Firestore Methods

Firestore Methods
firestore: Firestore
+ uploadPost(string, Uint8List, string): string + likePost(string, string, List): string + postComment(string, string, string): string + deletePost(string): string + followUser(string, string): void

Fields:

- firestore: Gets the firestore details.

Functions:

- uploadPost(): This function is used for uploading a post.
- likePost(): This function is used for liking a post.
- postComment(): This function is used for posting a comment on a post.
- deletePost(): This function is used for deleting a post.
- followUser(): This function is used for following an user.

4.2.2.1.10 Storage Methods

Storage Methods
auth: Firebase Auth # storage: Firebase Storage
+ uploadImageToStorage(string, Uint8List, bool): string

Fields:

- auth: Calls the auth methods to check whether there is a compatible user.
- storage: Firebase storage.

Functions:

- uploadImageToStorage(): This function is used for uploading an image to storage.

4.2.2.1.11 Routes

Routes
index: int
+ Routes(key, int): Routes - pageList(List): void

Fields:

- index: Gets the current page index.

Functions:

- Routes(): Constructor for Routes class.
- pageList(): Gets all the included pages.

4.2.2.1.12 Main Home Page

Main Home Page
- buildButton(): Widget - buildHeader(): Widget

Functions:

- buildButton(): This function is used for building buttons.

- buildHeader(): This function is used for building headers.

4.2.2.1.13 Wish List

4.2.2.1.14 Photos

4.2.2.1.15 Recent Places

4.2.2.1.16 Maps

Maps
<ul style="list-style-type: none"> - controller: Completer<GoogleMapController> - debounce: Timer - radiusSlider: Timer - getDirections: Timer - markers: Set<Marker> - markersDupe: Set<Marker> - radiusValue: var - allFavoritePlaces: List - tokenKey: string - pageController: PageController - placeImg: string - key: final - circle: Set<Circle>
<ul style="list-style-type: none"> - kPlex(): CurrentPosition - setCircle(LatLng): void - setNearMarker(LatLng, string, List, string): void - getByteFromAsset(string, int): Uint8List - initState(): void - fetchImage(): void

Fields:

- controller: Controlling Google API datas.
- debounce : Google API asynchron method control.
- radiusSlider: : Slider for setting radius value.
- getDirections: Drawing and controlling polylines for Google maps.
- markers: The icon for Google maps.
- markersDupe: The dupe of the marker.
- radiusValue: It determines how many meters of border places will be shown to the user.
- allFavoritePlaces: List of favorite places of the user.
- tokenKey: Allows us to take place API.
- pageController: Controller for PageController class.
- placeImage: Image of the place.
- key: The actual Google API key.

- circle: The shape that we use for getting scale.

Functions:

- kPlex(): Starting location function.
- setCircle(): This function is used for setting a circle based on radius value.
- setNearMarker(): The parsing place datas and getting details.
- label(): Label for map.
- getByteFromAsset(): Receiving images from Google API.
- initState(): Initial start function.
- fetchImage(): This function is used for getting the image of the place.

4.2.2.1.17 Map Service

Map Service
key: string # types: string
<ul style="list-style-type: none"> - searchPlaces(string): List<AutocompleteResult> - getPlace(string): Map<String, dynamic> - getDirections(string, string): Map<String, dynamic> - getPlaceDetails(LatLng, int): dynamic - getMorePlaceDetails(string): dynamic

Fields:

- key: The actual API key.
- types: Type of the place.

Functions:

- searchPlaces(): This function is used for searching places.
- getPlace(): This function is used for getting places.
- getDirections(): This function is used for getting directions.
- getPlaceDetails(): This function is used for getting place details.
- getMorePlaceDetails(): This function is used for getting extra information about the place.

4.2.2.1.18 Auto-Complete Result

Auto Complete Result
description: string # placeId: string # reference: string
+ AutoCompleteResult(): AutoCompleteResult - toJson(): Map<String, dynamic>

Fields:

- description: The description that gets from the Google API location.
- placeId: ID of the place.
- reference: The link used for receiving reference.

Functions:

- AutoCompleteResult(): Constructor for AutoCompleteResult class.
- toJson(): Converting function that json to list.

4.2.2.1.19 Place Result

Place Result
- setResult(): void

Functions:

- setResult(): This function is used for setting result.

4.2.2.1.20 Activity Page

4.2.2.1.21 Discover Page

4.2.2.1.22 Post Page

4.2.2.1.23 Profile Page

Profile Page
vid: string
+ ProfilePage(key, string): ProfilePage - getData(): void

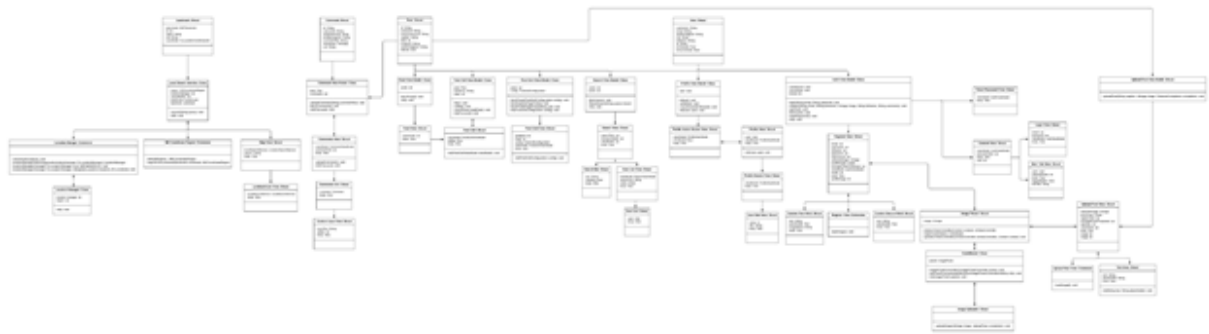
Fields:

- uid: The ID of the user who has logged in to the application.

Functions:

- ProfilePage(): Constructor for Profile Page class.
- getData(): This function is used for getting data.

4.2.2.2 iOS



4.2.2.2.1 Landmark

Landmark: Struct
<ul style="list-style-type: none">- placemark: MKPlacemark- id: int- name: String- title String- coordinate: CLLocationCoordinate2D

Fields:

- placemark: Includes information about the place.
- id: ID of the place.
- name: Name of the place.
- title: Title of the place.
- coordinate: Coordination of the place.

4.2.2.2.2 Local Search Service

Local Search Service: Class
<ul style="list-style-type: none">- region: MKCoordinateRegion- locationManager: let- cancellables: var- landmarks: [Landmark]- landmark: Landmark
<ul style="list-style-type: none">- search(String query): void- init(): void

Fields:

- region: Region.
- locationManager: Location manager.
- cancellables: Temporary variable for location.
- landmarks: Landmarks lists.
- landmark: Landmark.

Functions:

- search(): This function is for searching.
- init(): Initialization function.

4.2.2.2.3 Location Manager-Extension

Location Manger: Extension
<ul style="list-style-type: none">- checkAuthorization(): void- locationManagerDidChangeAuthorization(manager CLLocationManager): locationManager- locationManager(manager CLLocationManager, error didFailWithError): void- locationManager(manager CLLocationManager, didUpdateLocations locations, [CLLocation]): void

Functions:

- checkAuthorization(): This function is for checking authorization.
- locationManagerDidChangeAuthorization(): This function is for checking whether the manager made a change the authorization or not.
- locationManager(): Constructor for location manager.

4.2.2.2.4 Location Manager-Class

Location Manager: Class
- location manager: let - region: var
- init(): void

Fields:

- locationManager: Location manager.
- region: Region of the location.

Functions:

- init(): Initialization function.

4.2.2.2.5 MK Coordinate Region

MK Coordinate Region: Extension
- defaultRegion() : MKCoordinateRegion - regionFromLandmark(landmark Landmark): MKCoordinateRegion

Functions:

- defaultRegion(): This function is for setting default region.
- regionFromLandmark(): This function is for finding region from the landmark.

4.2.2.2.6 Map View

Map View: Struct
- localSearchService: LocationSearchService - search: String - body: View

Fields:

- localSearchService: Location search service object.
- search: Searched content.

- body: Body of the view.

4.2.2.2.7 Landmark List View

Landmark List View: Struct
<ul style="list-style-type: none"> - localSearchService: LocalSearchService - body: View

Fields:

- localSearchService: Local search service object.
- body: Body of the view.

4.2.2.2.8 Comment

Comment: Struct
<ul style="list-style-type: none"> - id: String - username: String - postOwnerId: String - profileImageUrl: String - commentText: String - timestamp: Timestap - uid: String

Fields:

- id: ID of the comment.
- username: Username of the user who made the comment.
- postOwnerId: ID of the user who owns the post.
- profileImageUrl: Profile image of the user who made the comment.
- commentText: The content of the comment.
- timestamp: The date the comment was made.
- uid: ID of the user who made the comment.

4.2.2.2.9 Comment View Model

Comment View Model: Class
- post: Post - comments: var
- uploadComment(String commentText): void - fetchComments(): void - init(Post post): void

Fields:

- post: The post that was commented about.
- comments: All comments of the post.

Functions:

- uploadComment(): This function is for uploading the comment.
- fetch(): This function is for fetching all the comments of the post.
- init(): Initialization function.

4.2.2.2.10 Comments View

Comments View: Struct
- viewModel: CommentViewModel - commentText: var - body: View
- uploadComment(): void - init(Post post): void

Fields:

- viewModel: Comment view model.
- commentText: The content of the comment.
- body: Body of the view.

Functions:

- uploadComment(): This function is for uploading the comment.
- init(): Initialization function.

4.2.2.2.11 Comments Cell

Comments Cell: Struct
- comment: Comment - body: View

Fields:

- comment: Comment object.
- body: Body of the view.

4.2.2.2.12 Custom Input View

Custom Input View: Struct
- inputText: String - action: var - body: View

Fields:

- inputText: Placeholder text.
- action: Action button.
- body: Body of the view.

4.2.2.2.13 Post

Post: Struct
- id: String - ownerId: String - ownerUsername: String - caption: String - likes: int - imageUrl: String - ownerImageUrl: String - didLike: bool

Fields:

- id: ID of the post.
- ownerId: ID of owner of the post.
- ownerUsername: Username of owner of the post.
- caption: Caption of the post.
- likes: Amount of likes of the post.
- imageUrl: URL of the image of the post.
- timestamp: The date of the post was shared.
- ownerImageUrl: The URL of the profile picture of owner of the post.
- didlike: Whether the post is liked or not.

4.2.2.2.14 Feed View Model-Class

Feed View Model: Class
- posts: var
- fetchPosts(): void - init(): void

Fields:

- posts: Posts.

Functions:

- fetchPosts(): This function is for fetching the posts.
- init(): Initialization function.

4.2.2.2.15 Feed View

Feed View: Struct
- viewModel: var - body: View

Fields:

- viewModel: View model object.
- body: Body of the view.

4.2.2.2.16 Feed Cell View Model

Feed Cell View Model: Class
- post: Post - likeString: String - label: var
- like(): void - unlike(): void - checkIfUserLikedPost(): void - init(Post post): void

Fields:

- post: Post object.
- likeString: Controls the string "like" or "likes" based on the amount of likes.
- label: Label of the string.

Functions:

- like(): This function is for liking the post.
- unlike(): This function is for unliking the post.
- checkIfUserLikedPost(): This function checks whether the user has liked the post before.
- init(): Initialization function.

4.2.2.2.17 Feed Cell

Feed Cell: Struct
- viewModel: FeedCellViewModel - didlike: Bool - body: View
- init(FeedCellViewModel viewModel): void

Fields:

- viewModel: Feed cell view model object.
- didLike: Whether the post is liked or not.
- body: Body of the view.

Functions:

- init(): Initialization function.

4.2.2.2.18 Post Grid View Model

Post Grid View Model: Class
- posts: var - config: PostGridConfiguration
- fetchPosts(PostGridConfiguration config): void - fetchExplorePagePosts(): void - fetchProfilePagePostst(String uid): void - init(PostGridConfiguration config): void

Fields:

- posts: Posts.
- config: Configuration for post grid.

Functions:

- fetchPosts(): This function is for fetching posts.
- fetchExplorePagePosts(): This function is for fetching explore page posts.
- fetchProfilePagePosts(): This function is for fetching profile page posts.
- init(): Initialization function.

4.2.2.2.19 Post Grid View

Post Grid View: Struct
- gridItems: let - width: let - config: PostGridConfiguration - viewModel: PostGridViewModel - body: view
- init(PostGridConfiguration config): void

Fields:

- gridItems: Grid items.
- width: Width of the grid.
- PostGridConfiguration: Post grid view model object.
- body: Body of the view.

Functions:

- init(): Initialization function.

4.2.2.2.20 Search View Model

Search View Model: Class
- users: var - posts: var
- fetchUsers(): void - filteredUsers(String query): [User] - init(): void

Fields:

- users: Users.
- posts: Posts.

Functions:

- fetchUsers(): This function is for fetching users.
- filteredUsers(): Lists of filtered users.
- init(): Initialization function.

4.2.2.2.21 Search View

Search View: Struct
- searchText: var - inSearchMode: var - viewModel: var - body: View

Fields:

- searchText: Searched text.
- inSearchMode: Whether view is in search mode or not.
- viewModel: View model object.
- body: Body of the view.

4.2.2.2.22 Search Bar

Search Bar: Struct
- text: String - isEditing: Bool - body: View

Fields:

- text: The content written into the search bar to search.
- isEditing: Whether the user is editing the text or not.
- body: Body of the view.

4.2.2.2.23 User List View

User List View: Struct
- viewModel: SearchViewModel - searchText: String - users: [User] - body: View

Fields:

- viewModel: Search view model object.
- searchText: Searched text.
- users: Users list.
- body: Body of the view.

4.2.2.2.24 User Cell

User Cell: Struct
- user: User - body: View

Fields:

- user: User object.
- body: Body of the view.

4.2.2.2.25 User

User: Struct
- username: String - email: String - imageUrl: String - uid: String - fullname: String - id: String - isFollowed: Bool - isCurrentUser: Bool

Fields:

- username: Username of the user.
- email: Email of the user.
- imageUrl: URL of the profile picture of the user.
- uid: User ID.
- fullname: Fullname of the user.
- id: Document ID of the posts in profile view.
- isFollowed: Bool variable for whether the user has already followed the user or not.
- isCurrentUser: Bool variable for whether the user is the current user or not.

4.2.2.2.26 Profile View Model

Profile View Model: Class
- user: User
- follow(): void - unfollow(): void - checkIfUserIsFollowed(): void - init(User user): void

Fields:

- user: User object.

Functions:

- follow(): This function is for following the user.
- unfollow(): This function is for unfollowing the user.
- checkIfUserIsFollowing(): This function checks whether the user has followed the user before or not.
- init(): Initialization function.

4.2.2.2.27 Profile Action Button View

Profile Action Button View: Struct
- viewModel: ProfileViewModel - isFollowed: Bool - body: View

Fields:

- viewModel: Profile view model object.
- isFollowed: This function checks whether the user has already followed the user or not.
- body: Body of the view.

4.2.2.2.28 Profile View

Profile View: Struct
<ul style="list-style-type: none"> - user: User - viewModel: ProfileViewModel - body: View
<ul style="list-style-type: none"> - init(User user): void

Fields:

- user: User object.
- viewModel: Profile view model object.
- body: Body of the view.

Functions:

- init(): Initialization function.

4.2.2.2.29 Profile Header View

Profile Header View: Struct
<ul style="list-style-type: none"> - viewModel: ProfileViewModel - body: View

Fields:

- viewModel: Profile view model object.
- body: Body of the view.

4.2.2.2.30 User Stat View

User Stat View: Struct
<ul style="list-style-type: none"> - value: int - title: String - body: View

Fields:

- value: Amount of posts.
- title: Text below the amount of posts.
- body: Body of the view.

4.2.2.2.31 Auth View Model

Auth View Model: Class
<ul style="list-style-type: none"> - userSession: User - currentUser: User - shared: let
<ul style="list-style-type: none"> - login(String email, String password): void - register(String email, String password, UIImage image, String fullname, String username): void - signOut(): void - fetchUser(): void - resetPassword(): void - init(): void

Fields:

- userSession: Users session.
- currentUser: User object belongs to current user.
- shared: AuthViewModel object.

Functions:

- login(): This function is for logging in.
- register(): This function is for registering.
- signOut(): This function is for signing out.
- fetchUser(): This function is for fetching user.
- resetPassword(): This function is for resetting the users password.
- init(): Initialization function.

4.2.2.2.32 Register View-Struct

Register View: Struct
<ul style="list-style-type: none">- email: var- username: var- fullname: var- password: var- repassword: var- selectedImage: UIImage- profileImage: Image- isImagePickerPresented: var- viewModel: AuthViewModel- mode: var- body: View- profileImage: let

Fields:

- email: Email of the user.
- username: Username of the user.
- fullname: Fullname of the user.
- password: Password of the user.
- selectedImage: Selected image by the user.
- profileImage: Profile image of the user.
- isImagePickedPresented:
- viewModel: Auth view model object.
- mode: Screen movement animation oriantation between register and login screen.
- body: Body of the view.
- profileImage: Image.

4.2.2.2.33 Custom Text Field

Custom Text Field: Struct
<ul style="list-style-type: none">- text: String- placeholder: Text- imageName: String- body: View

Fields:

- text: Input text.
- placeholder: Placeholder.
- imageName: Name of the image.
- body: Body of the view.

4.2.2.2.34 Register View-Extension

Register View: Extension
- loadImage(): void

Functions:

- loadImage(): This function is for loading an image.

4.2.2.2.35 Custom Secure Field

Custom Secure Field: Struct
- text: String - placeholder: Text - body: View

Fields:

- text: Input text.
- placeholder: Placeholder.
- body: Body of the view.

4.2.2.2.36 Image Picker

Image Picker: Struct
- image: UIImage
- makeUIViewController(Context context): UIViewController - makeCoordinator(): Coordinator - updateUIViewController(UIViewController viewController, Context context): void

Fields:

- image: Image.

Functions:

- makeUIViewController(): Function that controls image picker.
- makeCoordinator(): Function that controls coordinator.
- updateUIViewController(): Function that controls choosing image.

4.2.2.2.37 Coordinator

Coordinator: Class
- parent: ImagePicker
- imagePickerController(UImagePickerController picker): void - didFinishPickingMediaWithInfo(UImagePickerControllerInfoKey info): void - init(ImagePicker parent): void

Fields:

- parent: Image picker object.

Functions:

- imagePickerController(): Function that controls image picker.
- init(): Initialization function.

4.2.2.2.38 Image Uploader

Image Uploader: Struct
- uploadImage(UIImage image, UploadType completion): void

Functions:

- `uploadImage()`: This function is for uploading an image.

4.2.2.2.39 Reset Password View

Reset Password View: Struct
<ul style="list-style-type: none">- <code>viewModel</code>: <code>AuthViewModel</code>- <code>body</code>: <code>View</code>

Fields:

- `viewModel`: Auth view model object.
- `body`: Body of the view.

4.2.2.2.40 Content View

Content View: Struct
<ul style="list-style-type: none">- <code>viewModel</code>: <code>AuthViewModel</code>- <code>selectedIndex</code>: <code>var</code>- <code>body</code>: <code>View</code>- <code>user</code>: <code>let</code>

Fields:

- `viewModel`: Auth view model object.
- `selectedIndex`: Selected index.
- `body`: Body of the view.
- `user`: User.

4.2.2.2.41 Login View

Login View: Struct
<ul style="list-style-type: none">- email: var- password: var- viewModel: AuthViewModel- body: View

Fields:

- email: Email of the user.
- password: Password of the user.
- viewModel: Auth view model object.
- body: Body of the view.

4.2.2.2.42 Main Tab View

Main Tab View: Struct
<ul style="list-style-type: none">- user: User- selectedIndex: int- body: View- logoutButton: View- tabTitle: String

Fields:

- user: User object.
- selectedIndex: Selected index.
- body: Body of the view.
- logoutButton: Logout button.
- tabTitle: Title of the tab.

4.2.2.2.43 Upload Post View Model

Upload Post View Model: Struct
- uploadPost(String caption, UIImage image, FirestoreCompletion completion): void

Functions:

- uploadPost(): This function is for uploading a post.

4.2.2.2.44 Upload Post View-Struct

Upload Post View: Struct
- selectedImage: UIImage - postImage: Image - captionText: var - isImagePickerPresented: var - tabIndex: int - viewModel: var - body: view - image: let - image: let

Fields:

- selectedImage: Selected image.
- postImage: Image of the post.
- captionText: Caption of the post.
- isImagePickerPresented: A bool that controls image picker.
- tabIndex: The order of the posts.
- viewModel: View model object.
- body: Body of the view.
- image: Image.

4.2.2.2.45 Upload Post View-Extension

Upload Post View: Extension
- loadImage(): void

Functions:

- loadImage(): This function is for loading an image.

4.2.2.2.46 Text Area

Text Area: Struct
- text: String - placeholder: String - body: View
- init(String text, String placeholder): void

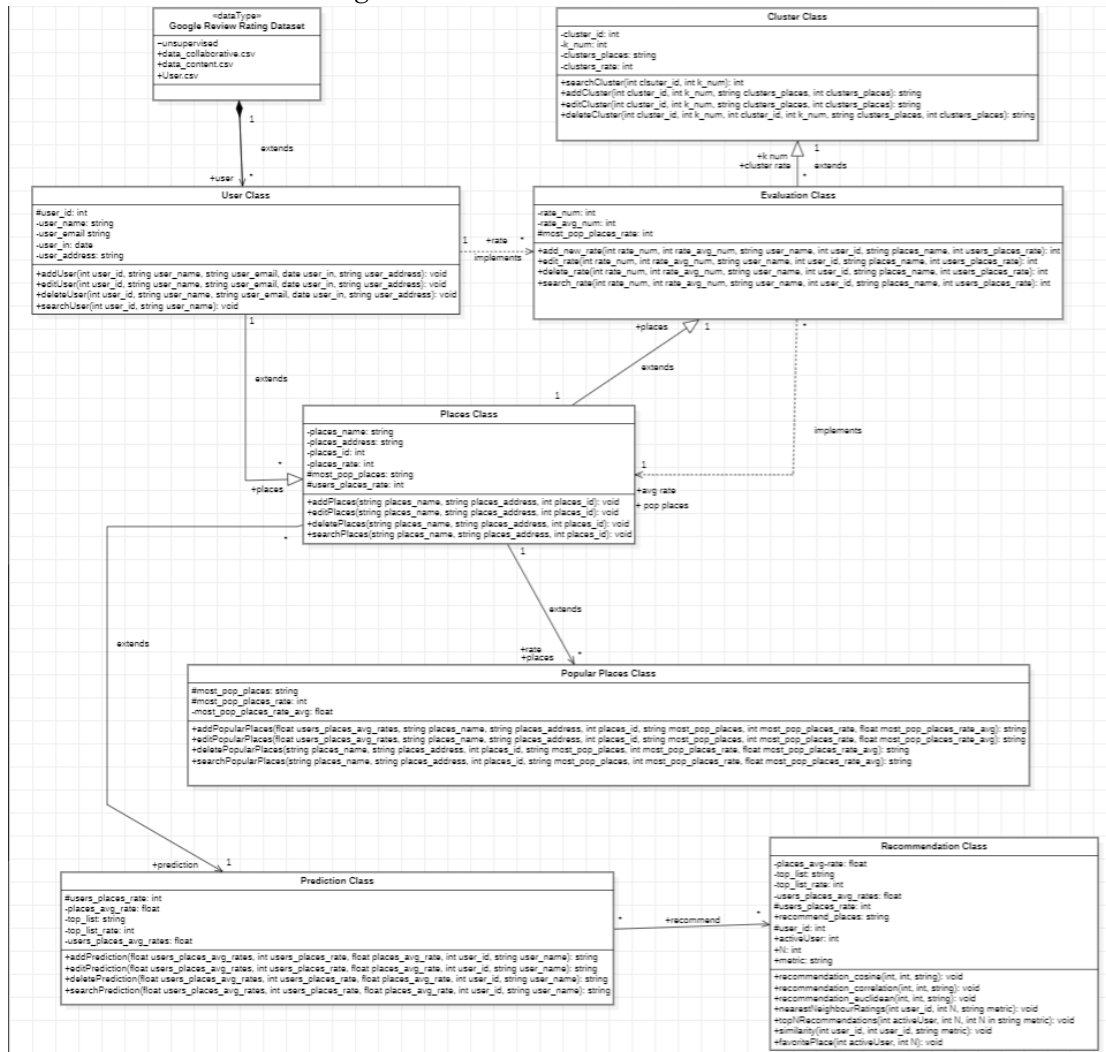
Fields:

- text: Input text.
- placeholder: Placeholder.
- body: Body of the view.

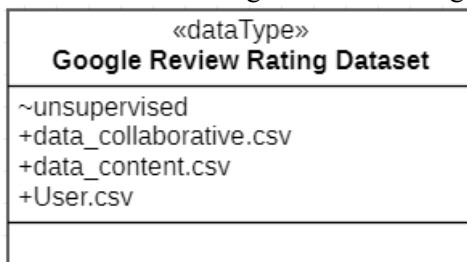
Functions:

- init(): Initialization function.

4.2.2.3 Machine Learning



4.2.2.3.1 Google Review Rating Dataset



4.2.2.3.2 User Class

User Class
<ul style="list-style-type: none">- user_id: int- user_name: string- user_email: string- user_in: date- user_address: string
<ul style="list-style-type: none">+ addUser(int, string, string, date, string): void+ editUser(int, string, string, date, string): void+ deleteUser(int, string, string, date, string): void+ searchUser(int, string): void

Fields:

- user_id: ID of the user.
- user_name: Name of the user.
- user_in: User's registration date.
- user_address: User's address.

4.2.2.3.3 Evaluation Class

Evaluation Class
<ul style="list-style-type: none">- rate_num: int- rate_avg: int# most_pop_places_rate: int
<ul style="list-style-type: none">+ add_new_rate(int, int, string, int, string, int): int+ edit_rate(int, int, string, int, string, int): int+ delete_rate(int, int, string, int, string, int): int+ search_rate(int, int, string, int, string, int): int

Fields:

- rate_num: Rating given by the user to a place.
- rate_avg: Rating of the place.
- most_pop_place_rate: The rank of the place in the popular places list.

Functions:

- add_new_rate(): This function is used for adding a new rate to a place.
- edit_rate(): This function is used for the editing rate of a place.
- delete_rate(): This function is used for deleting the rate of the place.
- search_rate(): This function is used for searching a rate of place by their ID.

4.2.2.3.4 Cluster Class

Cluster Class
<ul style="list-style-type: none">- cluster_id: int- k_num: int- cluster_places: string- clusters_rate: int
<ul style="list-style-type: none">+ searchCluster(int, int): int+ addCluster(int, int, string, int): string+ editCluster(int, int, string, int): string+ deleteCluster(int, int, int, int, int, string, int): string

Fields:

- cluster_id: ID value of the cluster.
- k_num: How many clusters the data will be divided into is determined by the k value.
- clusters_places: Places of the elements of the cluster.
- clusters_rate: Rate of the cluster value.

Functions:

- searchCluster(): This function is used for searching a cluster.
- addCluster(): This function is used for adding clusters.
- editCluster(): This function is used for editing a cluster.
- deleteCluster(): This function is used for deleting a cluster.

4.2.2.3.5 Places Class

Places Class
<ul style="list-style-type: none">- places_name: string- places_address: string- places_id: int- places_rate: int# most_pop_places: string# users_places_rate: int
<ul style="list-style-type: none">+ addPlaces(string, string, int): void+ editPlaces(string, string, int): void+ deletePlaces(string, string, int): void+ searchPlaces(string, string, int): void

Fields:

- places_name: Name of the place.
- places_address: Address of the place.

- places_id: ID value of the place.
- places_rate: Rate value of the place.
- most_pop_places: Most popular places.
- users_place_rate: Rating given by the user.

Functions:

- addPlaces(): This function is used for adding a place.
- editPlaces(): This function is used for editing a place.
- deletePlaces(): This function is used for deleting a place.
- searchPlaces(): This function is used for searching a place.

4.2.2.3.6 Prediction Class

Prediction Class
<pre># users_places_rate: int - places_avg_rate: float - top_list: string - top_list_rate: int - users_places_avg_rates: float</pre>
<pre>+ addPrediction(float, int, float, int, string): string + editPrediction(float, int, float, int, string): string + deletePrediction(float, int, float, int, string): string + searchPrediction(float, int, float, int, string): string</pre>

Fields:

- users_rate_places: User's rating of the place.
- places_avg_rate: Places average rate.
- top_list: Most liked places.
- top_list_rate: Most liked places rates.
- users_places_avg_rates: Average of ratings given by users.

Functions:

- addPrediction(): This function is used for adding prediction.
- editPrediction(): This function is used for editing prediction.
- searchPrediction(): This function is used for searching a prediction.
- deletePrediction(): This function is used for deleting a prediction.

4.2.2.3.7 Recommendation Class

Recommendation Class
<div><div>-places_avg-rate: float</div><div>-top_list: string</div><div>-top_list_rate: int</div><div>-users_places_avg_rates: float</div><div>#users_places_rate: int</div><div>+recommend_places: string</div><div>#user_id: int</div><div>+activeUser: int</div><div>+N: int</div><div>+metric: string</div></div>
<div>+recommendation_cosine(int, int, string): void</div> <div>+recommendation_correlation(int, int, string): void</div> <div>+recommendation_euclidean(int, int, string): void</div> <div>+nearestNeighbourRatings(int, int, string): void</div> <div>+topNRecommendations(int, int, int, string): void</div> <div>+similarity(int, int, string): void</div> <div>+favoritePlace(int, int): void</div>

Fields:

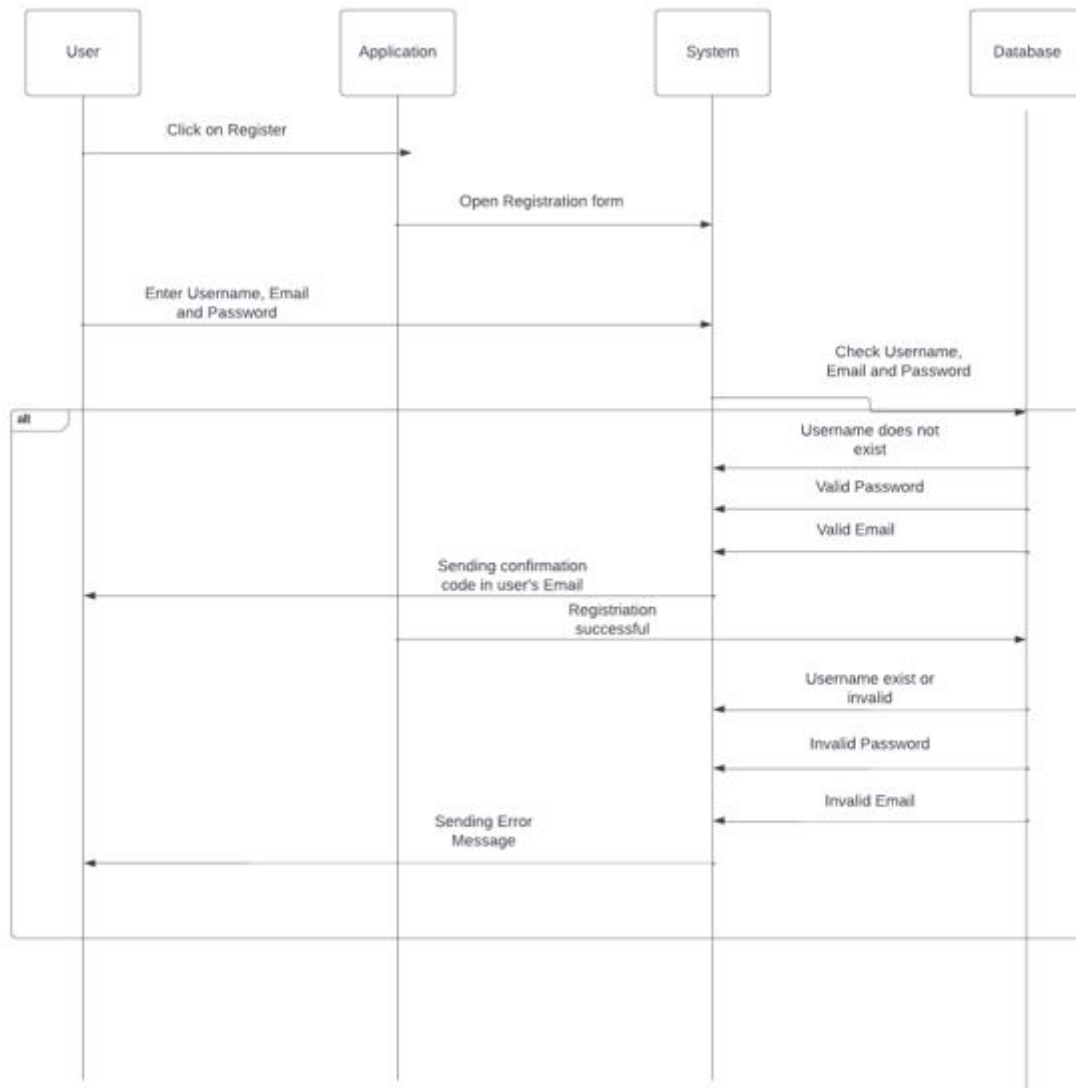
- users_places_rate: User's rating of the place.
- places_avg_rate: Places average rate.
- top_list: Most liked places.
- top_list_rate: Most liked places rates.
- users_places_avg_rates: Average of ratings given by users.
- recommend_places: Recommendation of users which rating and visiting places.
- user_id: Users have an id.
- activeUser: The recommend the places for specific user.
- N: User number to recommend places.
- metric: Which one select methods or theorems.

Functions:

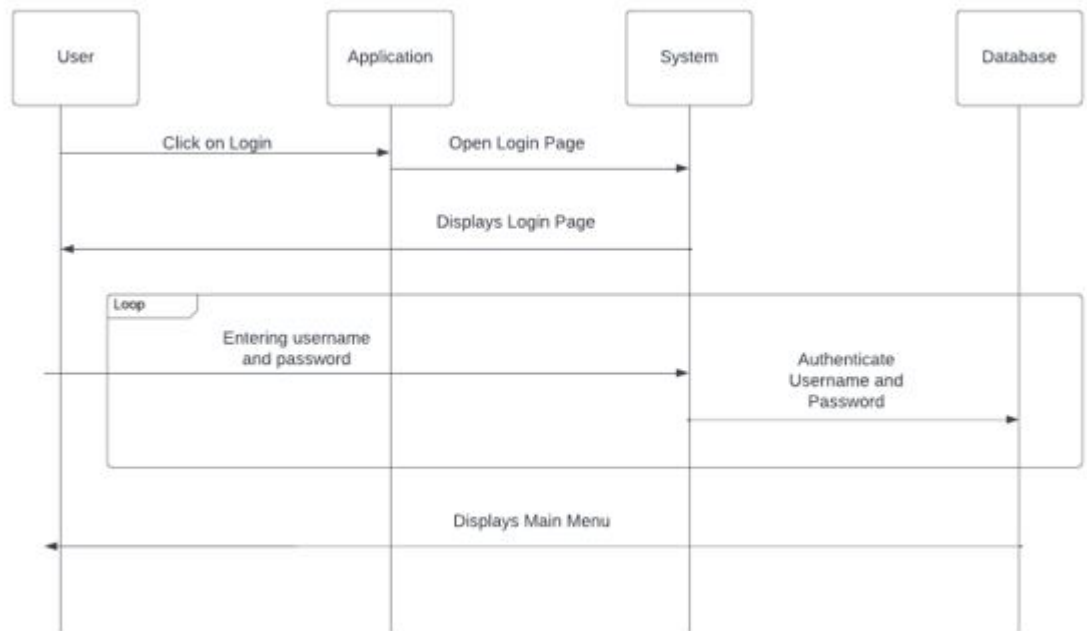
- recommendation_cosine(): This function is recommend for user to cosine theorem.
- recommendation_correlation(): This function is recommend user of correlation theorem.
- recommendation_euclidean(): This function is recommend user for euclidean theorem.
- nearestNeighbourRating(): This function is used the nearest neighbor method, it calculates users' pre-evaluations for more optimized and realistically.
- topNRecommendations(): This function is used for the most highest recommendation part of users.
- similarity(): This function is used for similarity calculator.
- favoritePlace(): This function is used for places where users are most frequent and have the highest ratings.

4.2.3 Sequence Diagrams

4.2.3.1 Register Page

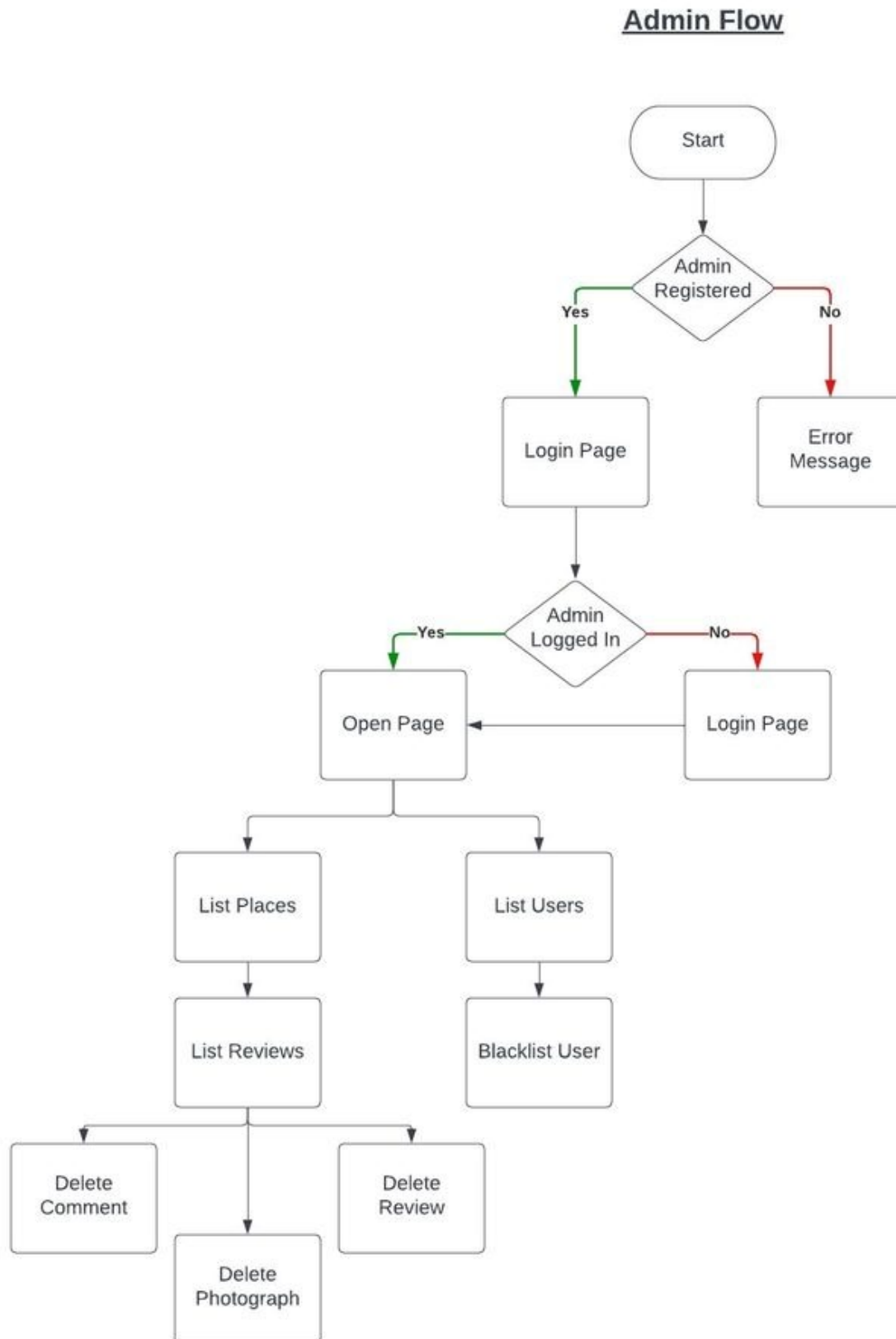


4.2.3.2 Login Page



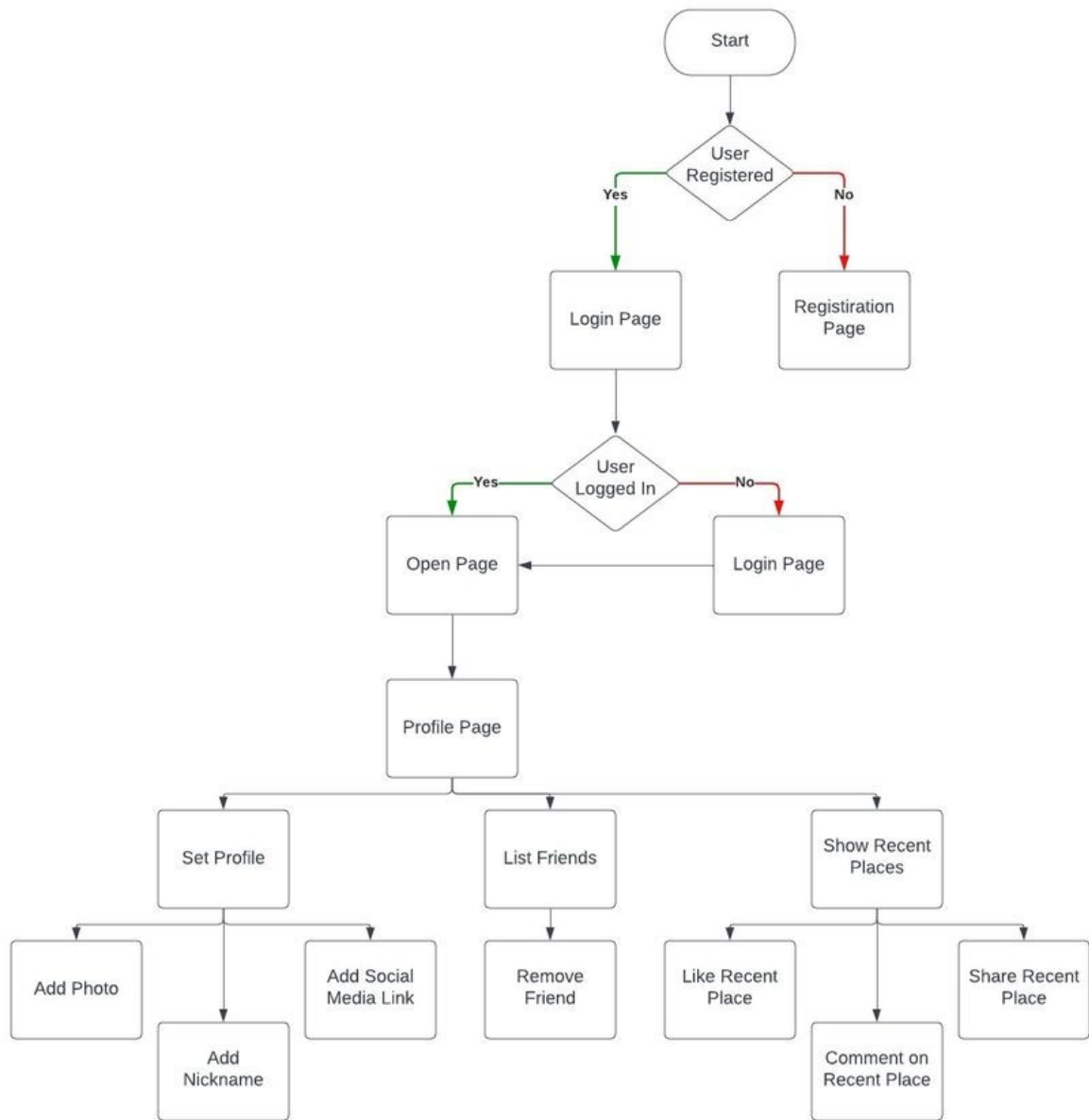
4.2.4 Activity Diagram

4.2.4.1 Admin

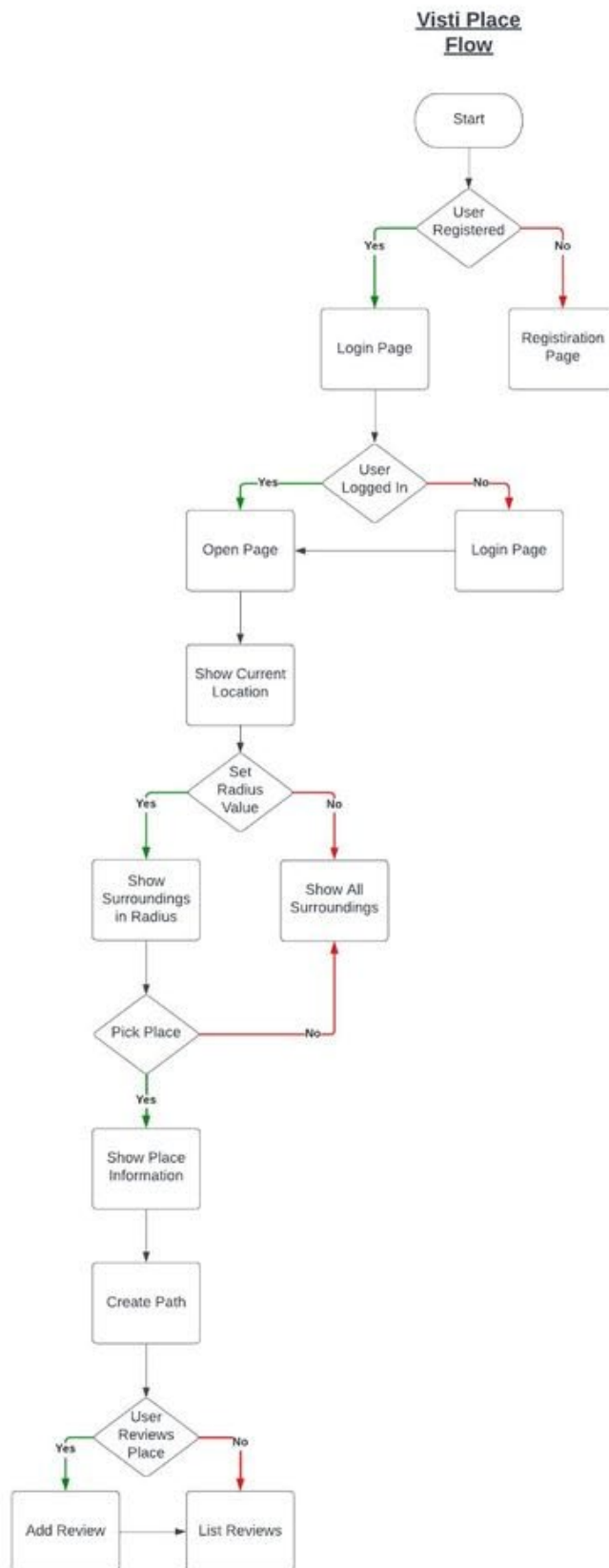


4.2.4.2 Profile

Profile Flow



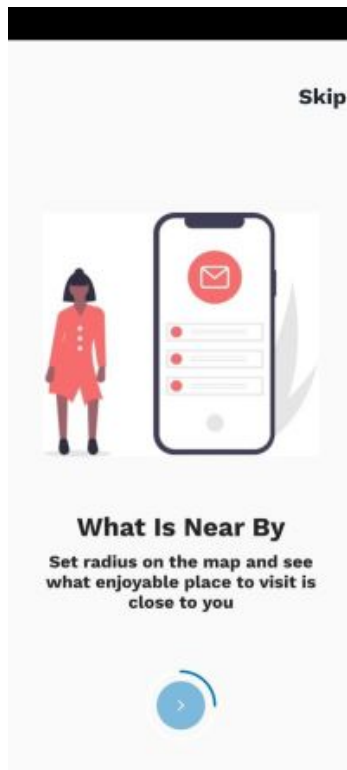
4.2.4.3 Visit Place



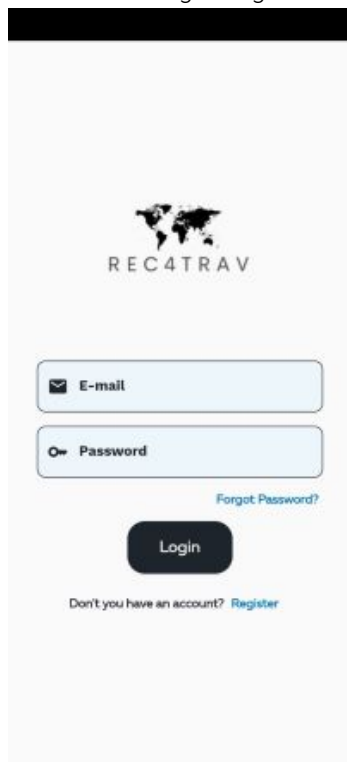
4.3 User Interface Design

4.3.1 Android

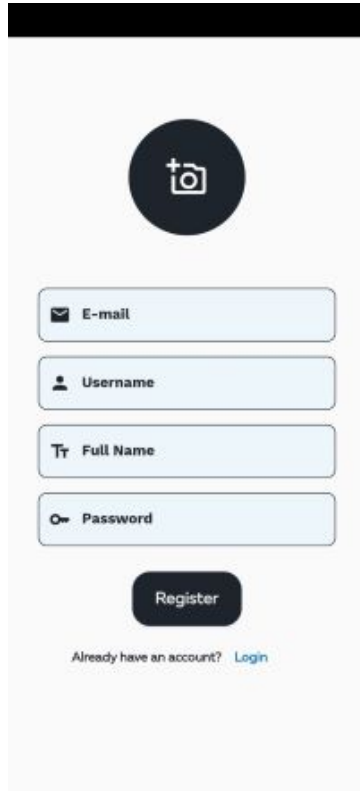
4.3.1.1 Open Page



4.3.1.2 Login Page



4.3.1.3 Sign Up Page

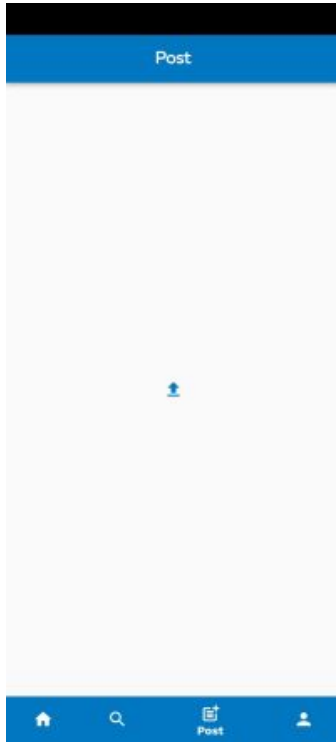


The Sign Up page features a dark blue header. Below it is a circular profile picture placeholder with a white camera icon. The registration form consists of four light blue input fields with rounded corners, each with a small icon on the left: an envelope for 'E-mail', a person for 'Username', a 'Tt' for 'Full Name', and a key for 'Password'. Below the fields is a dark blue 'Register' button. At the bottom, there is a link that says 'Already have an account? Login'.

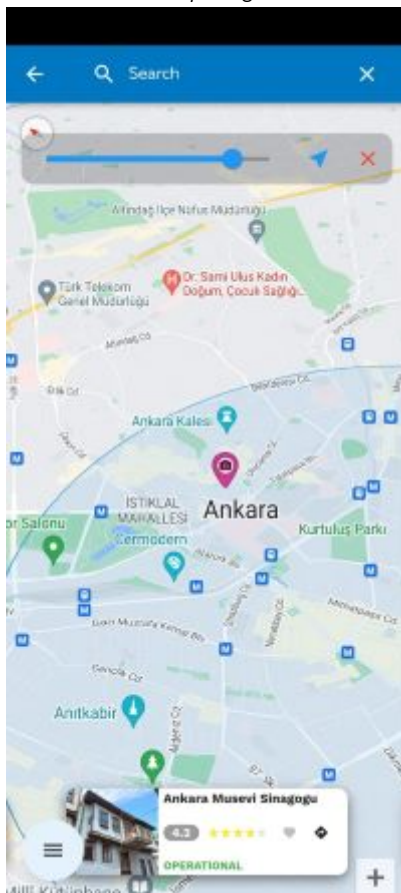
4.3.1.4 Main Page



4.3.1.5 New Post Page



4.3.1.6 Map Page

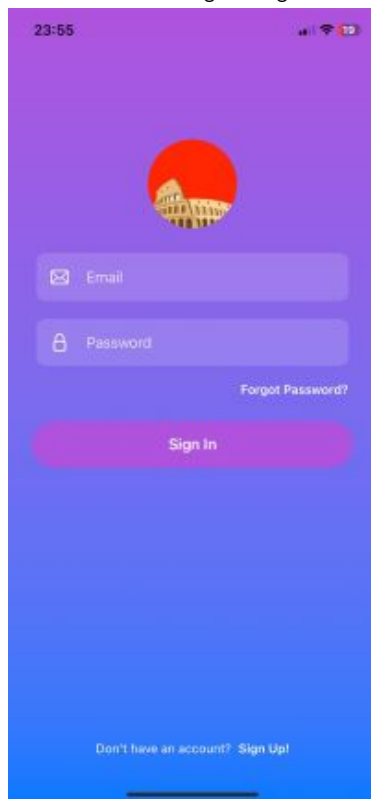


4.3.1.7 Profile Page

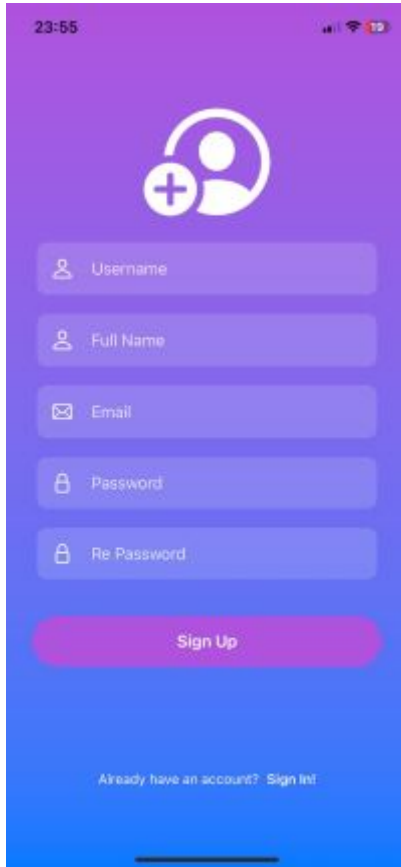


4.3.2 Apple

4.3.2.1 Login Page

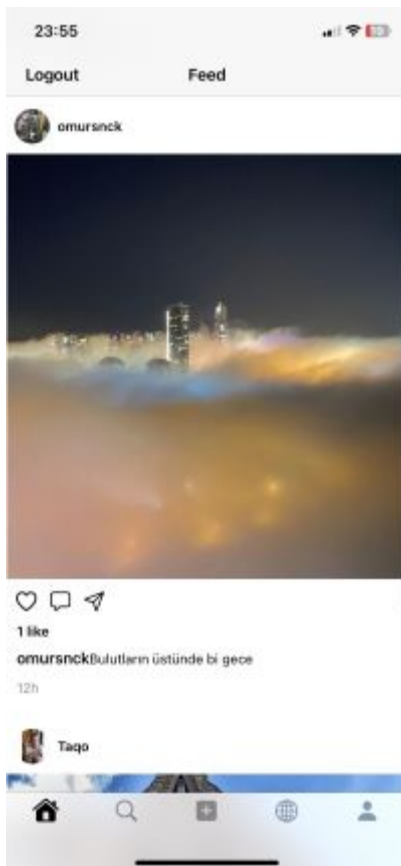


4.3.2.2 Sign Up Page

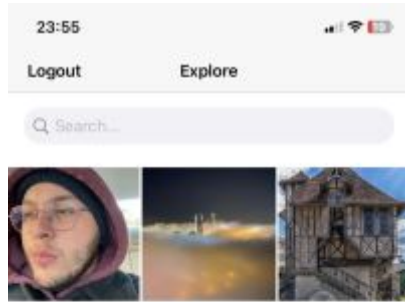


A screenshot of a mobile application's sign-up page. The background is a gradient from purple at the top to blue at the bottom. At the top left, the time is 23:55. At the top right, there are icons for signal strength, Wi-Fi, and battery. In the center, there is a white icon of a person with a plus sign inside a circle. Below this icon are five input fields, each with a small icon on the left: a person icon for 'Username', a person icon for 'Full Name', an envelope icon for 'Email', a lock icon for 'Password', and a lock icon for 'Re Password'. Below these fields is a large, rounded, pink button with the text 'Sign Up'. At the bottom, there is a link that says 'Already have an account? Sign In!'.

4.3.2.3 Feed



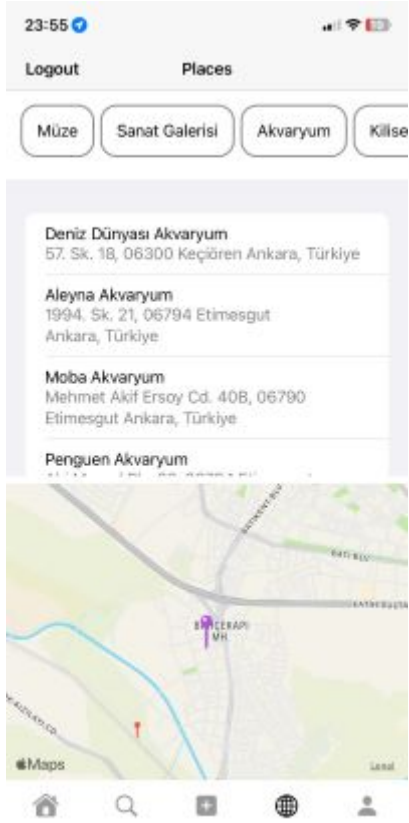
4.3.2.4 Explore Page



4.3.2.5 New Post Page



4.3.2.6 Map Page



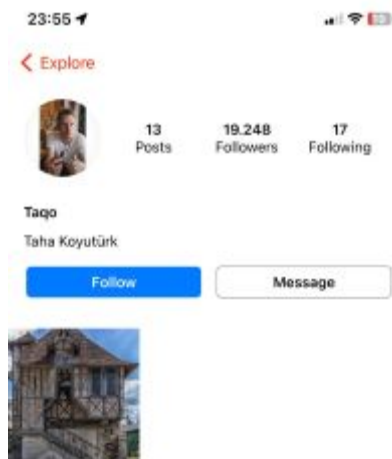
4.3.2.7 Profile Page



4.3.2.8 Search Page



4.3.2.9 Another Users Profile Page



4.4 Constraints

4.4.1 Time

We got limited Time to Create this project so we will be using 3 steps finish method.

-1st step is creating the general parts of the project (Creating UI, adding new users activating the mail system, etc..) which should take 1 week,

-2nd step is Adding details step which is Creating under layers of the project (Giving votes, calculating the average, banning user,s etc.) which should again take 1 week,

-3rd step is Completing the hardest part (which is Creating a map, showing restaurants and places around users, etc...) and completing the project which should take around 2 weeks.

4.4.2 Performance

As our project is a mobile project that is going to run on Users' phones our project's performance will be dependent on User's internet, Android, iOS phone Memory, and Android, iOS version.

4.4.3 Application Constraints

-User can only use approximately 5000 words when giving a review. The real value of the word is to be confirmed later.

-The maximum distance that a map can be drawn is approximately 20 kilometers. The real value of the distance is to be confirmed later.

5 Test Plan Document

5.1 Introduction

5.1.1 Overview

The purpose of this test plan document is to test for “Recommendation for Travelers” Project according to the Software Requirement Specification (SRS) and Software Design Description (SDD) documents we have prepared.

5.1.2 Scope

In our test document user login, showing recommended places, filtering, commenting and rating will be tested. These functions will be explained in detail in this document.

5.1.3 Glossary

Acronym	Definition
SI	SIGN IN
SU	SIGN UP
SO	SIGN OUT
U	USER
GM	GOOGLE MAPS PAGE
GMR	GOOGLE MAPS RECOMMENDATION
UI	USER INTERFACE
MP	MAIN PAGE
PP	PROFILE PAGE

5.2 Features to be Tested

5.2.1 Sign Up (SU)

Users must fill the fields. The fields are profile page, email, nickname and password.

5.2.2 Sign In (SI)

Users enter their email and password if they are already signed up.

5.2.3 Sign Out (SO)

Users can sign out from the logged application from the profile page.

5.2.4 Google Maps (GM)

Google Maps opens when the maps button from the page clicked. Google Maps checks your current location and if you wish to search any place or destination, users can experience.

5.2.5 Google Maps Recommendation

Recommendation System works when user is inside of the maps and sets the radius of the circle.

5.2.6 Main Page

When user sign up or sign in to the application, main page opens. Inside of the main page, we can see navigation bar and 4 buttons.

5.2.7 Profile Page

Users click on the far-right button on the navigator bar, they will successfully be redirected to the main page.

5.3 Features not to be Tested

The hardware of the mobile application will not be tested.

5.4 Pass/Fail Criteria

According to the SRS and SDD documents we have prepared, which functions of all functions are explained in detail in the documents. If the functions in these documents cannot be worked, the test will fail. Otherwise, the test will pass.

5.5 Exit Criteria

In this document, “H” refers to Highest priority and “M” refers to medium priority. If all the high and medium priority cases are passed the product is considered to be successful.

5.6 Test Design Specification

5.6.1 Sign In Part

TCID	Requirement	Priority	Scenario Description
SIUS01		H	Testing page if page opening
SIUS02		H	Enter valid email, invalid password
SIUS03		H	Enter valid Password, invalid email
SIUS04		H	Checking to see if there is a user in the firebase

5.6.2 Sign Up Part

TCID	Requirement	Priority	Scenario Description
SUUS01		H	Load picture, enter valid email, nickname and password
SUUS02		H	Load picture, enter valid email, nickname and blank password
SUUS03		H	Load picture, enter valid email, blank nickname and valid password
SUUS04		H	Load picture, enter invalid email, valid nickname and password
SUUS05		H	Not loading picture, enter valid email, nickname and password

5.6.3 Maps Part

TCID	Requirement	Priority	Scenario Description
MAP01		H	Testing Map loads
MAP02		H	Empty search bar field
MAP03		H	Empty origin and destination field
MAP04		H	Empty origin part, valid destination field
MAP05		H	Empty destination part, valid origin field
MAP06		H	Filled destination and origin part
MAP07		L	Setting radius is valid for slider
MAP08		H	Testing the location data loads valid
MAP09		H	Testing the API data for the location
MAP10		H	Recommendation system ML is accurate
MAP11		H	Recommendation system ML is inaccurate

5.6.4 Social Media Part

TCID	Requirement	Priority	Scenario Description
SM01		H	Testing Posting Picture
SM02		H	Testing Commenting post
SM03		H	Testing Add friend
SM04		H	Testing Remove Friend

5.7 Detailed Test Cases

5.7.1 SIUS01

TCID	SIUS01
Purpose	Testing page if page opening
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	No Dependency
Setup	Developer can create.
Procedure	[A01] Go to the introduction page. [A02] Click the skip button or the navigator button. [A03] Observe that login page appears.
Cleanup	Refresh the page.

5.7.2 SIUS02

TCID	SIUS02
Purpose	Enter valid email, invalid password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	No dependency
Setup	Sign in page should be open
Procedure	[A01] Go to the sign in page. [A02] Fill the email field valid. [A03] Fill the password field invalid. [A04] Click the Login button. [A05] Observe that the login is successful and main page appears.
Cleanup	Logout

5.7.3 SIUS03

TCID	SIUS03
Purpose	Enter valid Password, invalid email
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SIUS02 must be worked
Setup	Sign in page should be open
Procedure	[A01] Go to the sign in page. [A02] Fill the email field invalid. [A03] Fill the password field valid. [A04] Click the Login button. [A05] Observe that the login is not successful and error occurs.
Cleanup	Logout

5.7.4 SIUS04

TCID	SIUS04
Purpose	Checking to see if there is a user in the firebase
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SIUS02 must be worked
Setup	Sign in page should be open
Procedure	[A01] Go to the sign in page. [A02] Fill the email field valid. [A03] Fill the password field valid. [A04] Click the Login button. [A05] Observe that the login is not successful and error occurs.
Cleanup	Logout

5.7.5 SUUS01

TCID	SUUS01
Purpose	Load picture, enter valid email, nickname and password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	No Dependency
Setup	Sign up Page should be open
Procedure	[A01] Go to sign up page. [A02] Load picture. [A03] Fill the email field valid. [A04] Fill the nickname field valid. [A05] Fill the password field valid. [A06] Click the Register button. [A07] Observe that the register is successful and main page appears.
Cleanup	Logout

5.7.6 SUUS02

TCID	SUUS02
Purpose	Load picture, enter valid email, nickname and invalid password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SUUS01 must be worked
Setup	Sign up page should be open
Procedure	[A01] Go to sign up page. [A02] Load picture. [A03] Fill the email field valid. [A04] Fill the nickname field valid. [A05] Fill the password field invalid. [A06] Click the Register button. [A07] Observe that the register is unsuccessful and error occurs.
Cleanup	Logout

5.7.7 SUUS03

TCID	SUUS03
Purpose	Load picture, enter valid email, invalid nickname and valid password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SUUS01 and SUUS02 must be worked
Setup	Sign up page should be open
Procedure	[A01] Go to sign up page. [A02] Load picture. [A03] Fill the email field valid. [A04] Fill the nickname field invalid. [A05] Fill the password field valid. [A06] Click the Register button. [A07] Observe that the register is unsuccessful and error occurs.
Cleanup	Logout

5.7.8 SUUS04

TCID	SUUS04
Purpose	Load picture, enter invalid email, valid nickname and password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SUUS01 and SUUS02 must be worked
Setup	Sign up page should be open
Procedure	[A01] Go to sign up page. [A02] Load picture. [A03] Fill the email field invalid. [A04] Fill the nickname field valid. [A05] Fill the password field valid. [A06] Click the Register button. [A07] Observe that the register is unsuccessful and error occurs.
Cleanup	Logout

5.7.9 SUUS05

TCID	SUUS05
Purpose	Not loading picture, enter valid email, nickname and password
Requirements	-
Priority	High
Estimated Time Needed	0-1 second
Dependency	SUUS01 and SUUS04 must be worked
Setup	Sign up page should be open
Procedure	[A01] Go to sign up page. [A02] Don't load picture. [A03] Fill the email field valid. [A04] Fill the nickname field valid. [A05] Fill the password field invalid. [A06] Click the Register button. [A07] Observe that the register is unsuccessful and error occurs.
Cleanup	Logout

5.7.10 MAP01

TCID	MAP01
Purpose	Testing Map loads
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	SIUS01 and SUUS01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Observe map page opening.
Cleanup	Refresh page.

5.7.11 MAP02

TCID	MAP02
Purpose	Empty search bar field
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click the search bar. [A03] Click text field and don't text. [A04] Click search. [A05] Observe the map searching is unsuccessful, but no error occurs.
Cleanup	Refresh Page

5.7.12 MAP03

TCID	MAP03
Purpose	Empty origin and destination field
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click the destination bar. [A03] Click origin text field and don't fill. [A04] Click destination text field and don't fill. [A05] Click 'go'. [A06] Observe the maps directions is unsuccessful, but no error occurs.
Cleanup	Refresh Page

5.7.13 MAP04

TCID	MAP04
Purpose	Empty origin part, valid destination field
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click the destination bar. [A03] Click origin text field and don't fill. [A04] Click destination text field and fill. [A05] Click 'go'. [A06] Observe the maps directions is unsuccessful, but no error occurs.
Cleanup	Refresh Page

5.7.14 MAP05

TCID	MAP05
Purpose	Empty destination part, valid origin field
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click the destination bar. [A03] Click origin text field and fill. [A04] Click destination text field and don't fill. [A05] Click 'go'. [A06] Observe the maps directions is unsuccessful, but no error occurs.
Cleanup	Refresh Page

5.7.15 MAP06

TCID	MAP06
Purpose	Filled destination and origin part
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click the destination bar. [A03] Click origin text field and fill. [A04] Click destination text field and fill. [A05] Click 'go'. [A06] Observe the maps directions is successful.
Cleanup	Refresh Page

5.7.16 MAP07

TCID	MAP07
Purpose	Setting radius is valid for slider
Requirements	-
Priority	Low
Estimated Time Needed	0-1 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click anywhere in the map. [A03] The previously entered default value can be changed. [A04] The value can change the circle size and the circle search's locations. [A05] Observe that the locations inside of the circle is successful and location data appear.
Cleanup	Refresh Page

5.7.17 MAP08

TCID	MAP08
Purpose	Testing the location data loads valid
Requirements	-
Priority	High
Estimated Time Needed	1-5 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click anywhere in the map. [A03] Set radius. [A04] Observe that locations marked or not. If markers can't load, then test is unsuccessful.
Cleanup	Refresh Page

5.7.18 MAP09

TCID	MAP09
Purpose	Testing the API data for the location
Requirements	-
Priority	High
Estimated Time Needed	1-5 minute.
Dependency	MAP01 must work.
Setup	The user should be shown a map
Procedure	[A01] Click the map button. [A02] Click anywhere in the map. [A03] Set radius. [A04] Observe that location's data loaded into the bottom card successful. If couldn't load, it will be unsuccessful.
Cleanup	Refresh Page

5.7.19 MAP10

TCID	MAP10
Purpose	Recommendation system ML is accurate
Requirements	-
Priority	High
Estimated Time Needed	1 minute.
Dependency	MAP01, MAP08 and MAP09 must work.
Setup	The user should be shown a map
Procedure	<p>[A01] Click the map button.</p> <p>[A02] Click anywhere in the map.</p> <p>[A03] Location marker appears.</p> <p>[A04] On the card below, ML system works.</p> <p>[A05] Observe that the places on the card related with the user. If related, it is successful.</p>
Cleanup	Refresh Page

5.7.20 MAP11

TCID	MAP11
Purpose	Recommendation system ML is inaccurate
Requirements	-
Priority	High
Estimated Time Needed	1 minute.
Dependency	MAP01, MAP08 and MAP09 must work.
Setup	The user should be shown a map
Procedure	<p>[A01] Click the map button.</p> <p>[A02] Click anywhere in the map.</p> <p>[A03] Location marker appears.</p> <p>[A04] On the card below, ML system works.</p> <p>[A05] Observe that the places on the card related with the user. If it is unrelated, it is unsuccessful.</p>
Cleanup	Refresh Page

5.7.21 SM01

TCID	SM01
Purpose	Testing Posting Picture
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	SIUS01 and SUUS01 must be worked
Setup	User profile page should be shown
Procedure	<p>[A01] Click post page from the navigation bar.</p> <p>[A02] Click load picture button.</p> <p>[A03] Select a picture what you want to load.</p> <p>[A04] Load image.</p> <p>[A05] Write comment to post.</p> <p>[A06] Observe that uploaded image of the post is successful.</p>
Cleanup	Refresh Page

5.7.22 SM02

TCID	SM02
Purpose	Testing Commenting post
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	SIUS01, SUUS01 and SM01 must be worked
Setup	User profile page should be shown
Procedure	<p>[A01] Click post page from the navigation bar.</p> <p>[A02] Click load picture button.</p> <p>[A03] Select a picture what you want to load.</p> <p>[A04] Load image.</p> <p>[A05] Write comment to post.</p> <p>[A06] Observe that comment of the post is successful.</p>
Cleanup	Refresh Page

5.7.23 SM03

TCID	SM03
Purpose	Testing Add friend
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	SIUS01 and SUUS01 must be worked
Setup	User profile page should be shown
Procedure	<p>[A01] Click discover page from the navigation bar.</p> <p>[A02] Click search bar.</p> <p>[A03] Type the nickname of the user.</p> <p>[A04] Click on the user you want to add as a friend.</p> <p>[A05] Profile page appears.</p> <p>[A06] Click the add friend button.</p> <p>[A07] Observe that friend request is successful or not.</p>
Cleanup	Refresh Page

5.7.24 SM04

TCID	SM04
Purpose	Testing Remove Friend
Requirements	-
Priority	High
Estimated Time Needed	0-1 minute.
Dependency	SIUS01, SUUS01 and SM03 must be worked
Setup	User profile page should be shown
Procedure	<p>[A01] Click discover page from the navigation bar.</p> <p>[A02] Click search bar.</p> <p>[A03] Type the nickname of the user.</p> <p>[A04] Click on the user you want to add as a friend.</p> <p>[A05] Profile page appears.</p> <p>[A06] Click the remove friend button.</p> <p>[A07] Observe that removing friend is successful or not.</p>
Cleanup	Refresh Page

6 Test Results

6.1 Individual Test Results

SIUS01	-	High	Testing page if page opening	30.05.2023	Abdullah Ömür Şenocak	Pass
SIUS02	-	High	Enter valid email, invalid password	30.05.2023	Abdullah Ömür Şenocak	Fail
SIUS03	-	High	Enter valid password, invalid email	30.05.2023	Abdullah Ömür Şenocak	Fail
SIUS04	-	High	Checking to see if there is a user in the firebase	30.05.2023	Abdullah Ömür Şenocak	Pass
SUUS01	-	High	Load picture, enter valid email, nickname and password	30.05.2023	Abdullah Ömür Şenocak	Pass
SUUS02	-	High	Load picture, enter valid email, nickname and blank password	30.05.2023	Abdullah Ömür Şenocak	Fail
SUUS03	-	High	Load picture, enter valid email, blank nickname and valid password	30.05.2023	Abdullah Ömür Şenocak	Fail
SUUS04	-	High	Load picture, enter invalid email, valid nickname and password	30.05.2023	Abdullah Ömür Şenocak	Fail
SUUS05	-	High	Not loading picture, enter valid email, nickname and password	30.05.2023	Abdullah Ömür Şenocak	Fail
MAP01	-	High	Testing map loads	29.05.2023	Taha Koyutürk	Pass
MAP02	-	High	Empty search bar field	29.05.2023	Taha Koyutürk	Pass
MAP03	-	High	Empty origin and destination field	29.05.2023	Taha Koyutürk	Fail
MAP04	-	High	Empty origin part, valid destination field	29.05.2023	Taha Koyutürk	Fail
MAP05	-	High	Empty destination part, valid origin field	29.05.2023	Taha Koyutürk	Fail
MAP06	-	High	Filled destination and origin part	29.05.2023	Taha Koyutürk	Pass
MAP07	-	Low	Setting radius is valid for slider	29.05.2023	Taha Koyutürk	Pass
MAP08	-	High	Testing the location data loads valid	29.05.2023	Taha Koyutürk	Pass
MAP09	-	High	Testing the API data for the location	30.05.2023	Pelin Su Gök	Pass
MAP10	-	High	Recommendation system ML is accurate	30.05.2023	Pelin Su Gök	Pass
MAP11	-	High	Recommendation system ML is inaccurate	30.05.2023	Pelin Su Gök	Fail
SM01	-	High	Testing Post Picture	30.05.2023	Pelin Su Gök	Pass
SM02	-	High	Testing Comment on Post	30.05.2023	Pelin Su Gök	Pass
SM03	-	High	Testing Add Friend	30.05.2023	Pelin Su Gök	Pass
SM04	-	High	Testing Remove Friend	30.05.2023	Pelin Su Gök	Pass

6.2 Summary of Test Results

Priority	Number of Test Cases	Executed	Passed
H	23	23	13
L	1	1	1

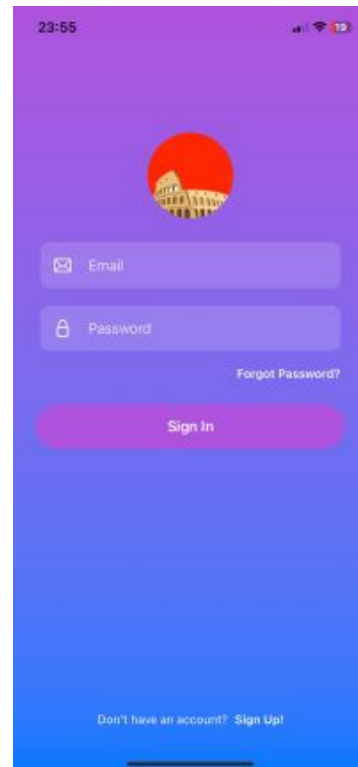
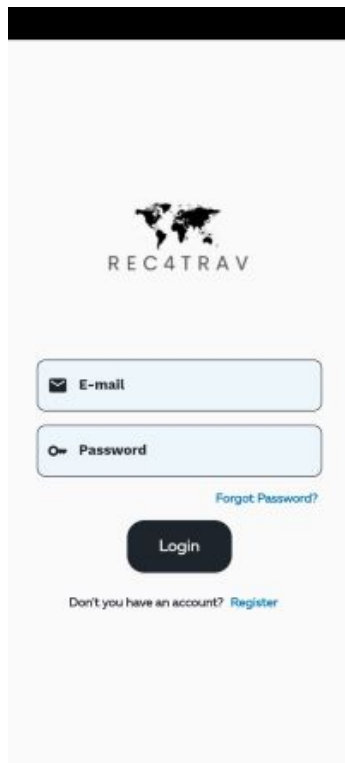
6.3 Exit Criteria

Criteria	Yes or No
100% of the test executed	Yes
100% of the test cases passed	No
All high, medium and low priority test cases passed	Yes

7 User Manual

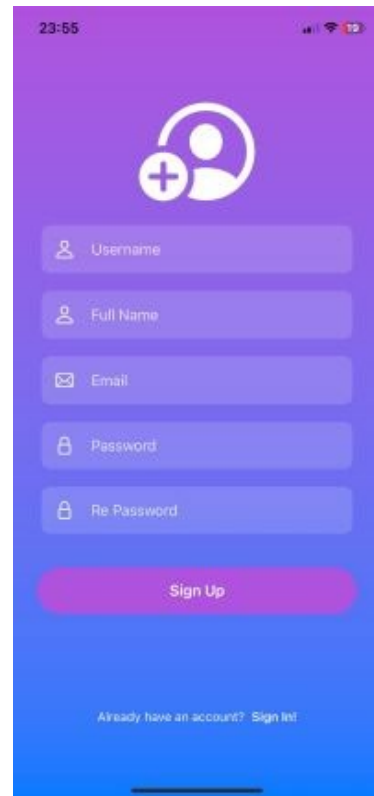
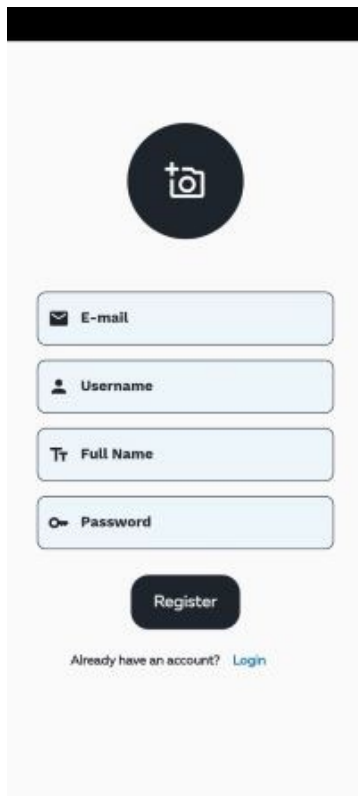
7.1 Login Page

The user can log in to the system by entering their e-mail and password information.



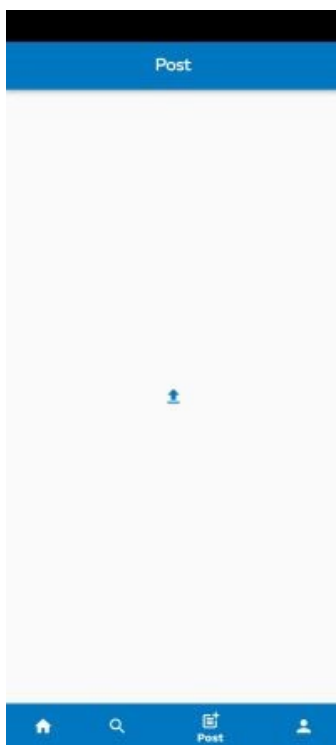
7.2 Sign Up Page

The user can register into the system by entering their name, surname, e-mail and password information.



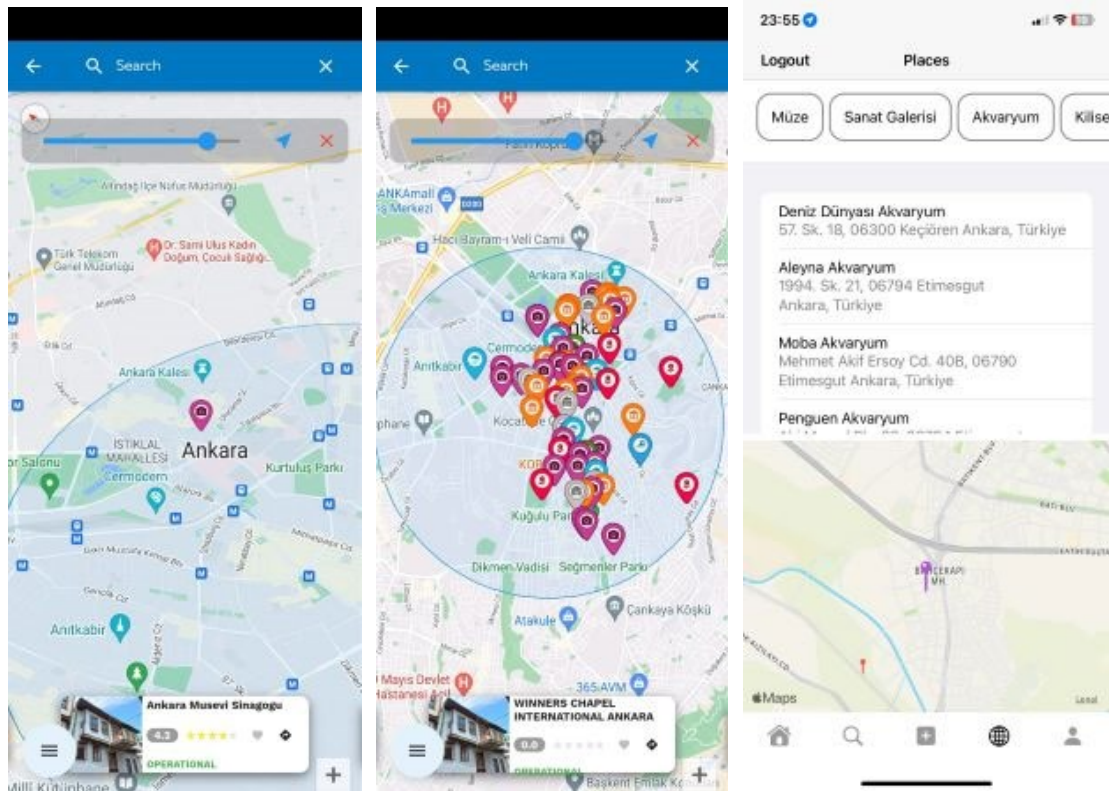
7.3 New Post Page

The user can add a new post to the application by choosing a picture from gallery or by taking a picture. For user to choose a picture from gallery user has to allow the application to access to their gallery on their phone.



7.4 Map Page

User can choose where they want to go by clicking on the name or the icon of the place.



7.5 Profile Page

User can see their profile information on their profile page, and also users can edit their profile information by “Edit Profile”.



8 Project Work Plan

8.1 Project Work Plan

Project Phases	Week1	Week2	Week3	Week4	Week5	Week6	Week7	Week8	Week9	Week10	Week11	Week12	Week13	Week14	Presentation
Updated Project Work Plan / LR / SRS / SDD															
Test Plan Document															
Midterm-Interactive-Demo															
Final Product 1st Release															
User Manual															
Project Report / Test Results / Project Tracking Form															
Project Poster															
Updated Project Webpage															
Demo Video															
Presentation															

8.2 Project Work Plan-Detailed

Project Phases	Week1	Week2	Week3	Week4	Week5	Week6	Week7	Week8	Week9	Week10	Week11	Week12	Week13	Week14	Presentation
Updated Project Work Plan / LR / SRS / SDD															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Final Product Demo															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Marketing Presentation Demo															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Final Product S.M Release															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
User Manual															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Project Report / Test Results / Project Tracking Form															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Project Report															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Updated Project Workpage															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Desktop Video															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Project Launch															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															
Presentation															
Talha KOYUTURK															
Abdullah Omur SENOCAK															
Perlin Su GOK															
Huseyin ERBASAN															
Ceren EROGLU															

9 Conclusion

In our CENG 408 study, we investigated and detailed the traveler system research recommendations. This report describes the design stage of our suggested application for traveler systems. The mobile application makes recommendations for locations within a user-specified radius based on their preferences. In order for users to interact and engage with other users, it is also allowed to leave comments and assessments of the areas visited. We found that customers may benefit from such frequently used recommendation applications as we were developing this project. Additionally, we attempted to enhance the drawbacks of competing programs and incorporate their strengths into our own. As a result, we have made it simpler for users to choose a lot faster and more convenient route, as well as to offer recommendations based on specific users. We have noticed that these programs have made it simpler for consumers to travel. We thoroughly investigated the Kotlin and similar programming languages during the project while designing our mobile application. Additionally, we decided to use Flutter and Swift because we felt it would be better for our project moving ahead. The fact that Flutter is a Google-created open-source platform on which UI software may be developed is another factor in our decision to use it. Nevertheless, one of the primary factors that led us to choose Flutter was the fact that it can be utilized on a variety of platforms (IOS, Android, etc.).

10References

- [1] V.K, Muneer & K P, Dr. (2020). The evolution of travel recommender systems: A comprehensive review. *Malaya Journal of Matematik*. 8. 10.26637/MJM0804/0075.
- [2] C. A. Gomez-Urbe and N. Hunt, *The Netflix recommender system: Algorithms, business value, and innovation*, TMIS, 2016.
- [3] Kenteris, Michael & Gavalas, Damianos & Mpitzopoulos, Aristides. (2010). A Mobile tourism recommender system. *Proceedings - IEEE Symposium on Computers and Communications*. 840-845. 10.1109/ISCC.2010.5546758.
- [4] L R, Roopesh & Bomatpalli, Tulasi. (2019). A Survey of Travel Recommender System. 10.13140/RG.2.2.34775.32168.
- [5] Isinkaye, Folasade Olubusola, Yetunde O. Folajimi, and Bolande Adefowoke Ojokoh. "Recommendation systems: Principles, methods and evaluation." *Egyptian informatics journal* 16.3 (2015): 261-273.
- [6] Das, Debashis, Laxman Sahoo, and Sujoy Datta. "A survey on recommendation system." *International Journal of Computer Applications* 160.7 (2017).
- [7] Shani, Guy, and Asela Gunawardana. "Evaluating recommendation systems." *Recommender systems handbook* (2011): 257-297.
- [8] Ko, Hyeyoung, et al. "A survey of recommendation systems: recommendation models, techniques, and application fields." *Electronics* 11.1 (2022): 141.
- [9] Lee, Tong Queue, Young Park, and Yong-Tae Park. "A time-based approach to effective recommender systems using implicit feedback." *Expert systems with applications* 34.4 (2008): 3055-3062.
- [10] Mitchell, Tom Michael. *Machine learning*. Vol. 1. New York: McGraw-hill, 2007.
- [11] Zhou, Zhi-Hua. *Machine learning*. Springer Nature, 2021.
- [12] Cunningham, Pádraig, Matthieu Cord, and Sarah Jane Delany. "Supervised learning." *Machine learning techniques for multimedia: case studies on organization and retrieval* (2008): 21-49.
- [13] Singh, Amanpreet, Narina Thakur, and Aakanksha Sharma. "A review of supervised machine learning algorithms." *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. Ieee, 2016.
- [14] Montgomery, Douglas C., Elizabeth A. Peck, and G. Geoffrey Vining. *Introduction to linear regression analysis*. John Wiley & Sons, 2021.
- [15] LaValley, Michael P. "Logistic regression." *Circulation* 117.18 (2008): 2395-2399.
- [16] Webb, Geoffrey I., Eamonn Keogh, and Risto Miikkilainen. "Naïve Bayes." *Encyclopedia of machine learning* 15 (2010): 713-714.
- [17] Sun, Shiliang, and Rongqing Huang. "An adaptive k-nearest neighbor algorithm." *2010 seventh international conference on fuzzy systems and knowledge discovery*. Vol. 1. IEEE, 2010.

-
- [18] Barlow, Horace B. "Unsupervised learning." *Neural computation* 1.3 (1989): 295-311.
- [19] Xu, Rui, and Donald Wunsch. "Survey of clustering algorithms." *IEEE Transactions on neural networks* 16.3 (2005): 645-678.
- [20] Kurita, Takio. "Principal component analysis (PCA)." *Computer Vision: A Reference Guide* (2019): 1-4.
- [21] Tukey, John W. *Exploratory data analysis*. Vol. 2. 1977.
- [22] Charland, Andre, and Brian Leroux. "Mobile application development: web vs. native." *Communications of the ACM* 54.5 (2011): 49-53.
- [23] McWherter, Jeff, and Scott Gowell. *Professional mobile application development*. John Wiley & Sons, 2012.
- [24] Gavalas, Damianos, and Daphne Economou. "Development platforms for mobile applications: Status and trends." *IEEE software* 28.1 (2010): 77-86.
- [25] Moskala, Marcin, and Igor Wojda. *Android Development with Kotlin*. Packt Publishing Ltd, 2017.
- [26] Bracha, Gilad. *The Dart programming language*. Addison-Wesley Professional, 2015.
- [27] García, Cristian González, et al. "Swift vs. objective-c: A new programming language." *IJIMAI* 3.3 (2015): 74-81.
- [28] Li, Yishan, and Sathiamoorthy Manoharan. "A performance comparison of SQL and NoSQL databases." 2013 IEEE Pacific Rim conference on communications, computers and signal processing (PACRIM). IEEE, 2013.
- [29] Mehta, Heeket, Pratik Kanani, and Priya Lande. "Google maps." *International Journal of Computer Applications* 178.8 (2019): 41-46.
- [30] Travel Shop Turkey, <https://travelshopturkey.com/>. Date Accessed: March 26, 2023.
- [31] Google Maps Platform | <https://developers.google.com/maps/documentation>, Date Accessed: March 26, 2023.
- [32] Flutter Documentation | <https://docs.flutter.dev/>, Date Accessed: March 26, 2023.
- [33] Adenowo, Adetokunbo AA, and Basirat A. Adenowo. "Software engineering methodologies: a review of the waterfall model and object-oriented approach." *International Journal of Scientific & Engineering Research* 4.7 (2013): 427-434.
- [34] Aughey, Robert J. "Applications of GPS technologies to field sports." *International journal of sports physiology and performance* 6.3 (2011): 295-310.
- [35] Indriana, Marcelli, and Muhammad Leyri Adzani. "UI/UX analysis & design for mobile e-commerce application prototype on Gramedia.com." 2017 4th International Conference on New Media Studies (CONMEDIA). IEEE, 2017.
- [36] Gavalas, Damianos & Konstantopoulos, Charalampos & Mastakas, Konstantinos & Pantziou, Grammati. (2014). Mobile Recommender Systems in Tourism. *Journal of Network and Computer Applications*. 39. 319–333. 10.1016/j.jnca.2013.04.006.
- [37] Park, Deuk-Hee & Kim, Hyea-Kyeong & Choi, Il-Young & Kim, Jae Kyeong. (2011). A Literature Review and Classification of Recommender Systems on Academic Journals. *Journal of Intelligence and Information Systems*. 17.
- [38] IEEE Standard for Information Technology - Systems Design - Software Design Descriptions **110**
<https://cengproject.cankaya.edu.tr/wpcontent/uploads/sites/10/2017/12/SDD-ieee-1016-2009.pdf>, Date Accessed: March 26, 2023.

[39] SOFTWARE DESIGN DOCUMENT | <https://senior.ceng.metu.edu.tr/2014/such/documents/SDD.pdf>, Date Accessed: March 26, 2023.

[40] Lucid | https://lucid.app/documents#/documents?folder_id=home, Date Accessed: March 26, 2023.

[41] Dataset_1 used in the machine learning (recommendation) part of the project: https://www.kaggle.com/datasets/gaurav146/tourist?select=data_collaborative.csv

[42] Dataset_2 used in the machine learning (recommendation) part of the project: https://www.kaggle.com/datasets/gaurav146/tourist?select=data_content.csv

[43] Dataset_3 used in the machine learning (recommendation) part of the project: <https://www.kaggle.com/datasets/gaurav146/tourist?select=User.csv>