TripPlanner: A Dynamic Tourism Recommendation System

A PROJECT REPORT

(Project Report Phase – II)

Submitted by

AKILLA VENKATA SESHA SAI	**-**	**212219220060**
MOHAMMED ZAID	**-**	**212219220031**
SURESHRAM E	**-**	**212219220054**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY

SAVEETHA ENGINEERING COLLEGE, THANDALAM.

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2023

BONAFIDE CERTIFICATE

Certified that this project report "**TripPlanner: A Dynamic Tourism Recommendation System**" is the bonafide work of **Mohammed Zaid (212219220031), Sureshram E (212219220054) and Akilla Venkata Sesha Sai (212219220060)**, who carried out the project work under my supervision.

SIGNATURE
Dr. K. Suresh Kumar,
HEAD OF THE DEPARTMENT
ASSOCIATE PROFESSOR
Department of Information Technology,
Saveetha Engineering College,
Saveetha Nagar, Thandalam,
Chennai-602105

SIGNATURE
Dr. G. Nalinipriya,
SUPERVISOR
PROFESSOR
Department of Information Technology,
Saveetha Engineering College,
Saveetha Nagar, Thandalam,
Chennai-602105

Submitted for VIVA-VOCE held on ………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We wish to express my gratitude to our Founder President **Dr.N.M.Veeraiyan**, Director **Dr. S. Rajesh,** Saveetha Engineering College, for their guidance and blessings.

We are very grateful to our Principal **Dr. N. Duraipandian M.E., Ph.D.,** for providing me with an environment to complete my project successfully.

We are indebted to our **Head of the Department, Dr. K. Suresh Kumar, M.Tech., Ph.D., for** his support during the entire course of this project work.

We are indebted to our **Supervisor Dr. G. Nalinipriya, M.E., Ph.D., Professor** for assisting me in the completion of my project with his/her exemplary Guidance and for his/her support during the entire course of this project.

Our heartfelt thanks to the **Project Coordinator, Dr. K. Suresh Kumar, M.Tech., Ph.D., Associate Professor,** Department of Information Technology, Saveetha Engineering College, for unstinted support throughout this project.

We also thank all the staff members of our department for their help in making this project successful.

ABSTRACT

With the advancement of the Internet, innovation, and communication methods, specifically, the advancement of online travel agencies (OTA), the development of information for vacationers has increased at all levels (lodgings, restaurants, transportation, legacy, vacationer events, exercises, and so on). This includes the development of information regarding legacy, vacationer events, and exercises. However, the potential results that are provided to visitors by web crawlers (or even specific vacationer destinations) can be both overwhelming and the essential outcomes are frequently buried in enlightened "commotion," which delays or at the very least slows down the process of decision-making. A number of recommender systems have been developed to provide assistance to travelers in the process of trip planning and in locating the information that they require. In this article, we will present an overview of the many different proposal processes that are employed in the travel industry. Based on the findings of this review, an engineering and theoretical framework for the travel industry recommender system is provided. This structure was developed in light of a half-and-half suggestion method. The proposed structure goes further than merely providing a list of vacation destinations on the basis of the preferences of tourists. It is possible to think of it as an excursion planner who assembles a particular plan for a specified amount of time throughout a visit by utilizing a wide range of travel industry resources. A clear goal is to construct a recommendation framework based on massive data advancements, artificial knowledge, and functional evaluation.

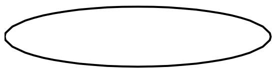
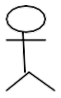
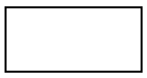
KEYWORDS*:*

Recommender Frameworks, The travel industry, Trip arranging, Client Profiling, Data analytics, Machine learning, Point of Interest

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS

S.NO.	SYMBOL NAME	SYMBOL
1.	USECASE	
2.	ACTOR	
3.	ENTITY SET	

LIST OF ABBREVIATIONS

S.NO	ABBREVIATION	EXPANSIION
1	UML	Unified Modeling Language
2	POI	Point of Interest
3	ML	Machine Learning
4	GPS	Global Positioning System

CHAPTER 1
INTRODUCTION

Recommendation frameworks can be highly useful in the tourism industry when it comes to organizing tours or searching for services among a variety of sights and activities. These frameworks utilize data filtering to recommend cost-effective offers to users, such as items that are comparable to other items they have previously purchased and enjoyed, or items that have been popular among other customers with similar tastes. One way to anticipate a client's level of interest in a certain item is to use the client's interests obtained during the route as inputs. There are various methods for calculating these levels of appreciation.

The project will classify subjects into categories based on data sources, including a compilation of feedback from previous customers on a wide range of issues. This approach is known as collaborative filtering, where the system recommends goods that have received positive reviews from other customers who share similar preferences. With the proliferation of informal groups, social data has become an increasingly important factor in developing social recommendation systems. These systems estimate the degree of similarity between a prospective client and their social circle based on a set of factors.

Recent research has recognized the significance of context-oriented data such as geography, weather patterns, and others in recommender systems. In the tourism industry, there are various types of entities, including people, places, and objects that hold importance. These entities can be categorized into different formats, such as landmarks, parks, galleries, and more. One approach that has been utilized to improve recommender systems in the travel industry is the "half-and-half" technique. This approach involves combining two different recommendation methods, namely, content-based and collaborative filtering, to provide better recommendations. However, implementing this framework poses several challenges due to the diversity of recommendation approaches, data sources, and information concepts in the travel industry.

1.1 OBJECTIVE

The objective of this project is to develop an intelligent tourism recommendation system using big data and AI techniques that can generate personalized recommendations for tourists based on their preferences and interests. The system will be implemented as an Android application and will utilize machine learning algorithms to analyze and process large volumes of data related to tourist attractions, accommodations, and other relevant factors. The goal is to provide users with accurate and timely recommendations that enhance their travel experiences, while also helping tourism businesses to better understand and meet the needs of their customers.

1.2 SCOPE OF THE PROJECT

The scope of this project is to design, develop and implement a tourism recommendation system based on big data and AI that provides personalized recommendations to tourists. The system will be developed using state-of-the-art machine learning techniques and will be able to analyze large volumes of data related to tourist attractions, accommodations, and other relevant factors. The project will involve data collection, processing, and storage from various sources to build a comprehensive database that will be used to generate recommendations.

The system will also employ various algorithms for collaborative filtering, content-based filtering, and hybrid filtering to provide customized recommendations based on the user's interests, preferences, and location. The project will also involve developing an Android application that provides a user-friendly interface for tourists to access the recommendation system. The interface will be designed to be intuitive, easy to navigate, and provide a seamless experience for users. The scope of the project also includes testing, evaluating and refining the recommendation system to ensure its effectiveness and accuracy. The project will aim to contribute to the tourism industry by providing personalized recommendations that enhance the travel experience for tourists while also helping tourism businesses better understand and serve the needs of their customers.

CHAPTER 2
LITERATURE REVIEW

Xianfeng Chen, et al in their paper "Approaching Another Tourism Recommender" proposed the concept, application and development status of travel recommender systems through the collection and arrangement of relevant literature published in recent years. Also, it pays special attention to the analysis of key technologies in the system, pointing out its complexity and uniqueness an application. Besides, the limitations of recommendation methods based on collaborative filtering and content-based filtering are considered as well, then the using knowledge-based filtering or hybrid recommendation method is proposed. It also discusses the role and application of tourism decision-making theory in the recommender system, and finally puts forward a general model of travel recommender system and future research hot spots. It's wished that this research can expand the vision and serve as a reference in this field.

Khalid AL Fararni et al, in their paper "The Optimal Tour Problem in Smart Tourism Recommender Systems Hybrid Recommender System for Tourism Based on Big Data and AI: A Conceptual Framework" proposed an overview of the various recommendation approaches used in the field of tourism. An architecture and a conceptual framework for tourism recommender system are proposed, based on a hybrid recommendation approach. The proposed system goes beyond the recommendation of a list of tourist attractions, tailored to tourist preferences. It can be seen as a trip planner that designs a detailed program, including heterogeneous tourism resources, for a specific visit duration. The ultimate goal is to develop a recommender system based on big data technologies, artificial intelligence, and operational research to promote tourism in Morocco, specifically in the Daraˆa-Tafilalet region.

Zineb Aarab et al, in their paper "Toward a Smart Tourism Recommender System: Applied to Tangier City" proposed and we investigate the various ways in which context information can be used to build recommender systems that are intelligent and adaptive. It gives an overview of the multifaceted concept of context, discusses many context-oriented approaches and systems, and explains how such approaches might be used in a variety of application domains. Through the use of a context metamodel, we will offer in this paper our understanding of the concept of context. We also provide an efficient tourist recommendation system that takes into account individual preferences and takes into account usage context along with personal context, social context, and environmental background. An applied case study was carried out in Tangier, which is located in Morocco and is going to be the site of the construction of a new intelligent international metropolis over the course of the next ten years.

R. Logesh and V. Subramaniyaswamy, in their paper "Exploring Hybrid Recommender Systems for Personalized Travel Applications" proposed a hybrid recommendation strategies in the e-Tourism area in order to bridge the gap between the problems that consumers face in the real world and the challenges that academics face in the digital world. In this article, we have discussed the challenges that have been encountered when doing research on e-tourism apps and have outlined a potential solution for improving the quality of tailored recommendations. By combining the user's contextual information, we were able to construct a Personalized Context-Aware Hybrid Travel Recommender System (PCAHTRS). The performance of the proposed PCAHTRS is assessed using the real-time, large-scale datasets provided by Yelp and TripAdvisor. The results of the experiments show that the performance of the proposed strategy is significantly better than that of standard ways. The report comes to a close with some principles for potential future work, which are intended to assist researchers in developing effective answers to recommendation issues.

Phatpicha Yochum et al, in their paper “Linked Open Data in Location-Based Recommendation System on Tourism Domain: A Survey” suggested to present a systematic review and mapping of the linked open data in location-based recommendation system on tourism domain, but also to provide an overview of the current research status in the area. First, we classify journal papers in this area from 2001 to 2018 by the year of publication. Second, we analyze and categorize journal papers by the different recommendation applications including problem formulations, data collections, proposed algorithms/systems, and experimental results. Third, we group the linked open data sources used in location-based recommendation system on tourism. Next, we summarize the research achievements and present the distribution of the different categories of location-based recommendation applications via linked open data. Last, we also guide the possible future research direction for the linked open data in location-based recommendations on tourism.

F´atima Leal et al, in their paper “Trust and Reputation Modelling for Tourism Recommendations Supported by Crowdsourcing” proposed a tourism recommendation system which integrates: (i) user profiling using the multi-criteria ratings; (ii) k-Nearest Neighbours (k-NN) prediction of the user ratings; (iii) Trust & Reputation modelling; and (iv) incremental model update, i.e., providing near real-time recommendations. In terms of contributions, this paper provides two different Trust & Reputation approaches: (i) general reputation employing the pairwise trust values using all users; and (ii) neighbour-based reputation employing the pairwise trust values of the common neighbours. The proposed method was experimented using crowdsourced datasets from Expedia and TripAdvisor platforms.

CHAPTER 3
SYSTEM ANALYSIS

3.1 RECOMMENDER SYSTEMS AND THEIR APPLICATION TO TOURISM

Nowadays users can find a lot of information on the Internet. That is why sometimes it becomes a hard and complex task to select the information a user is interested in. The user is often unable to look through all the available information. Therefore, highly interesting information can get lost in the middle of a sea of data. In the following paragraphs, we are going to mention the first three kinds of systems for searching information: search engines, helping systems and information filtering and retrieval systems: Search engines (such as Google or Yahoo). They are computer systems that index files stored in web servers. Searches are made with key words or hierarchical trees based on different topics. The result of the searches is a list of URLs in which the topics related to the introduced key words appear. Usually, the vast majority of information that search engines offer does not suit the user interests.

Helping systems: They teach the user how to use a particular program by describing it and explaining how it works. For instance, Microsoft includes a helping system that gathers information about the user characteristics and its actions, the state of the program, and the words that the user has searched, in order to calculate the probability of help that the user will need in related issues. These systems usually link web pages that provide additional information. The goal of these systems is to obtain the most relevant information for the user, minimizing the number of irrelevant items. Thus, filtering systems remove a huge amount of unwanted information. However, they could be more useful if they took user preferences into account with automatic learning techniques. If these systems were provided with artificial intelligence, they would supply users the content they are interested in, instead of just offering them a huge amount of information.

Before defining the concept of a recommender system itself, we should specify the difference between adaptable and adaptive systems. On the one hand, in the adaptable systems, the user decides the degree of adaptation, since he chooses the values of certain parameters. In the adaptive systems, on the other hand, the system adapts the suggestions without the intervention of the user. Both of these features can coexist in a single system without affecting its capacity. In fact, the combination of both aspects can increase the system effectiveness and efficiency.

3.2 EXISTING SYSTEM

In Existing system, the person who are visiting a particular city need to gather information from the person who is staying in the city or take the help of the guide in the city. The user needs to gather of all these information in order to visit the city. This requires a lot of time and pre-planning. In order to get each piece of information the user also needs to know what to search each time.

3.2.1 DISADVANTAGES OF EXISTING SYSTEM

- The existing system is quite difficult to use for beginners. Here the city information exists individually and to search any place requires the user to manually search it each time.
- There is also google maps which is very efficient but not easy to use when going to someplace new where the user doesn’t know what to look for.

- In the existing system, multiple locations can’t be compared at a time, making it difficult for the user to select the best place nearby.

3.3 . PROPOSED SYSTEM

The proposed system is an android application which integrates multiple features into a single easy to use app. This will prove to be extremely convenient for the users as it makes improvements over existing system. The app makes traveling and searching for nearby locations much easier and more flexible. It can be accessed over the Internet. The city information files can be stored in a centralized database which can be maintained by the system.

3.3.1. ADVANTAGES OF PROPOSED SYSTEM

- Trip planner works offline. No internet connection needed to Explore!
- Tourism apps allows travellers to explore city effectively.
- It can provide customized services to the users.
- Using a tourism app, travellers can get all kinds of bookings, such as hotels, cabs, events, etc., at one place.

CHAPTER 4
SYSTEM DESIGN

4.1. SYSTEM ARCHITECTURE

The architecture behind the approach is depicted in Fig. 4.1. It is configured into various modular tasks. From this, the process of recommendation can be understood. At first, the tourist tells the system which city he wants to visit, the time he wants to come and the place he wants to set as the beginning point. The start point is determined by the user's manual input or delivered by a GPS module embedded in his smart phone. After receiving these initial requirements, the system collects the POI information. Since the number of cities is limited and the scenic spots information would not change frequently, so the scenic spots metadata of cities have been collected offline. However, activities change every day, which means there are different activities on a separate day. So the system extracts activities information from events online. Then the system calculates the popularity of each POI, which makes up a POI database with other two attributes (name and geo-coordinates). With these preparations, the system could provide travel routes   for the user.

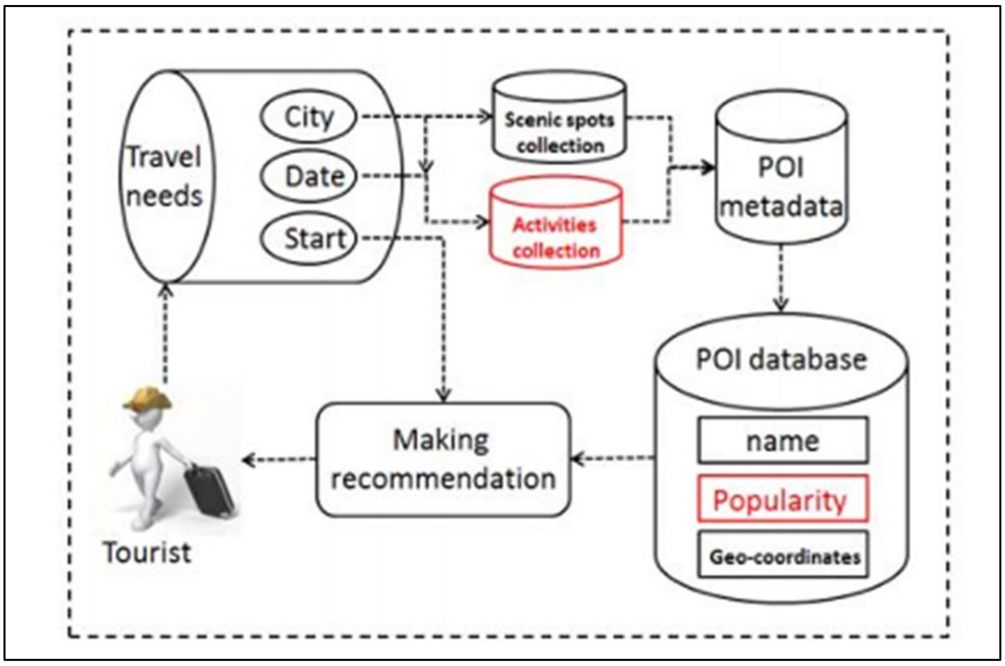


FIG 4.1. SYSTEM ARCHITECTURE

4.2. FLOW DIAGRAM

The flow of information starts at City Operator or a Data Provider which can either be common people or professional tourist guides, all the information generated are collected as info integrator. This information is made available in the system by using the means of API. When a user searches for content in the app, the content received will be from the integrator which in turn received it from City Operator or Data Providers.

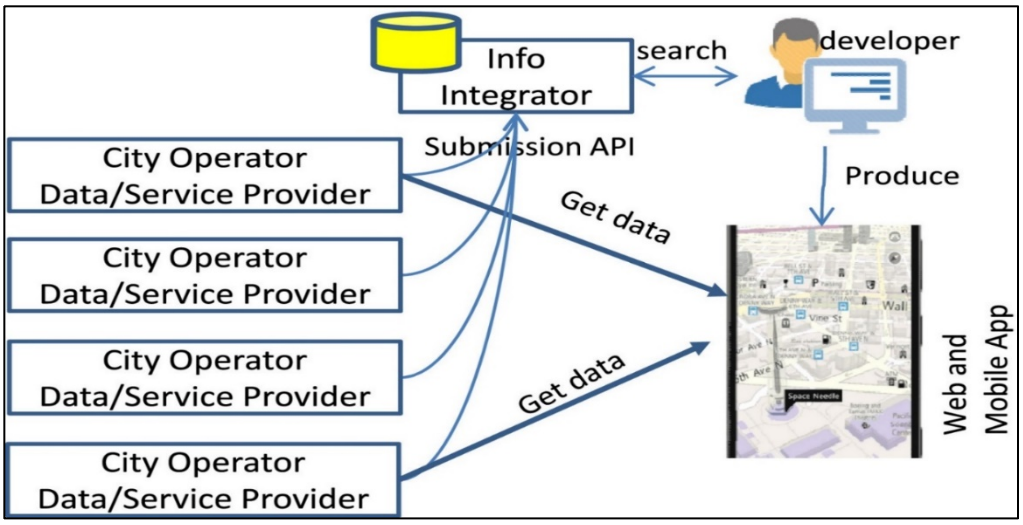

Info Integrator
search
developer
Submission API
Produce
City Operator Data/Service Provider
City Operator Data/Service Provider
City Operator Data/Service Provider
City Operator Data/Service Provider
Get data
Get data
Web and Mobile App

FIG 4.2. FLOW DIAGRAM

4.3. CLASS DIAGRAM

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

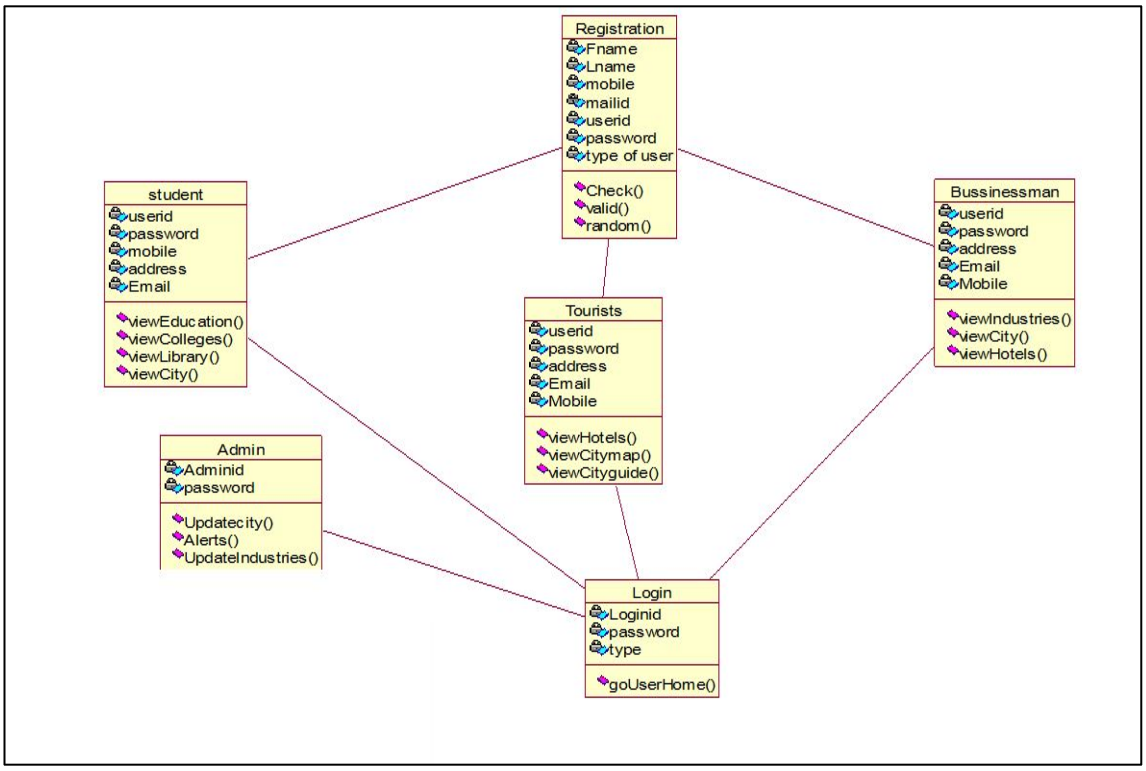


FIG 4.3. CLASS DIAGRAM DIAGRAM

4.4. UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

4.5. USE CASE DIAGRAMS

A use-case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally. A use case diagram is a way to summarize details of a system and the users within that system. It is generally shown as a graphic depiction of interactions among different elements in a system. Use case diagrams will specify the events in a system and how those events flow, however, use case diagram does not describe how those events are implemented.

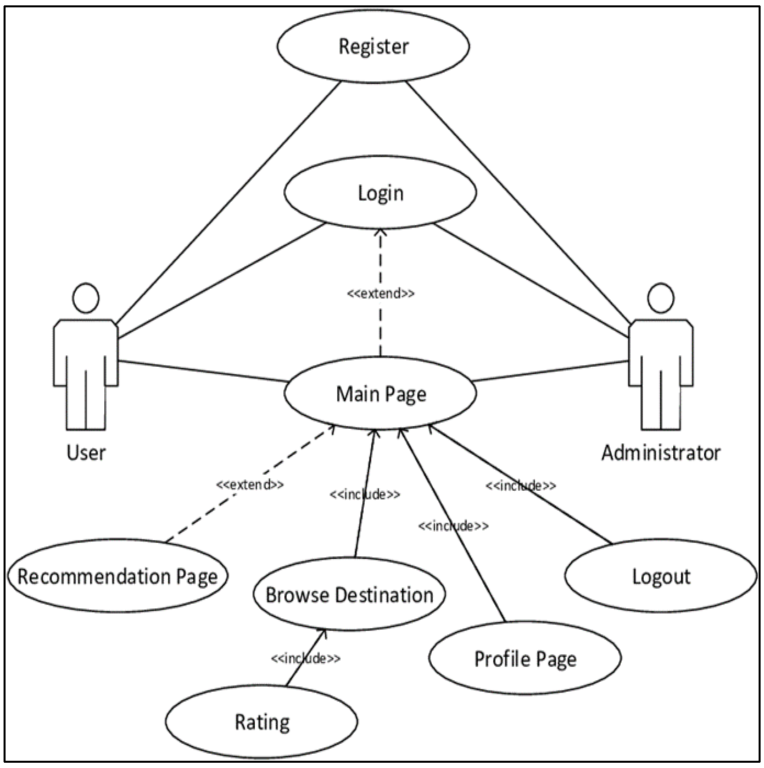


FIG 4.5. USE-CASE DIAGRAM

4.6. SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams. A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction.

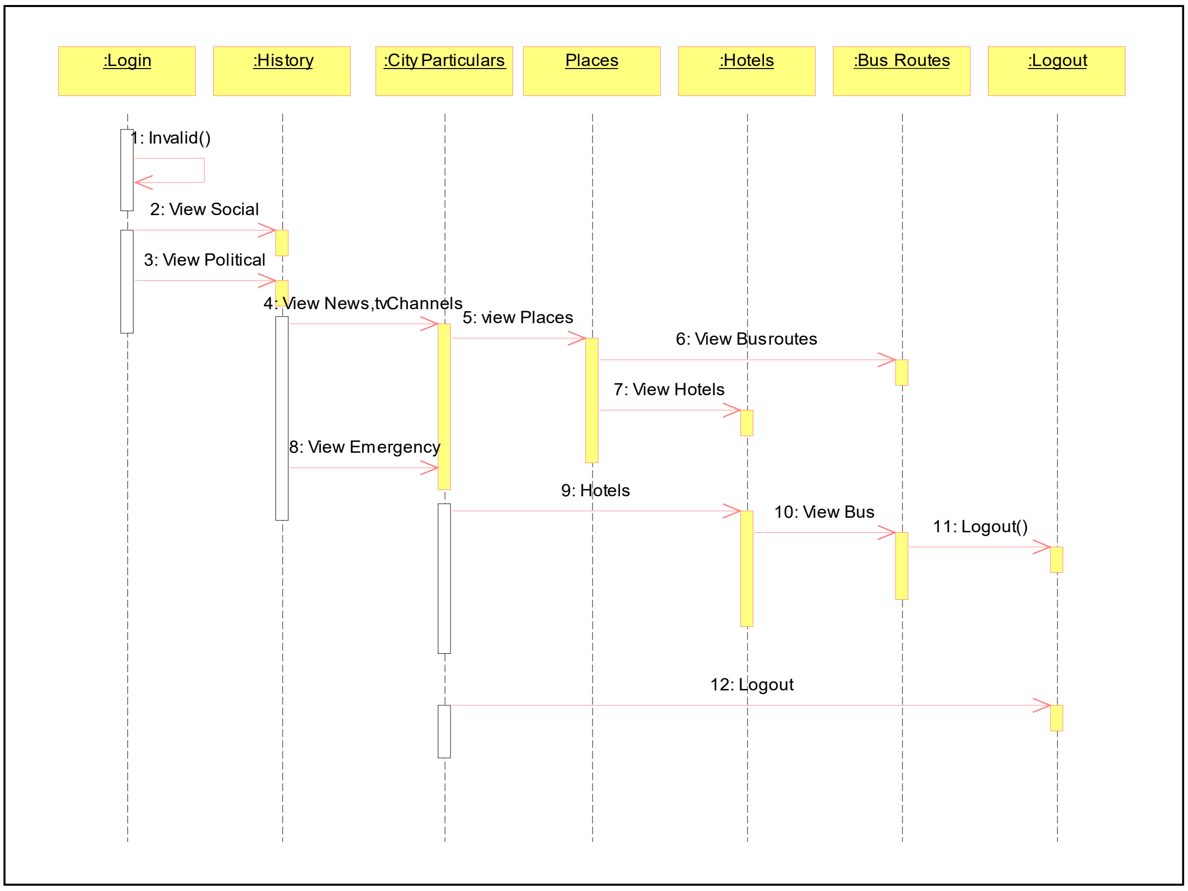


FIG 4.6. SEQUENCE DIAGRAM

CHAPTER 5
SYSTEM IMPLEMENTATION

5.1. HARDWARE SPECIFICATIONS

- System - Pentium-IV
- Speed - 2.4 GHZ
- Hard disk – 40 GB
- RAM – 512 MB

5.2. SOFTWARE SPECIFICATIONS

- Operating System - Windows 7
- Coding language - Java
- IDE – NetBeans
- Database – MYSQL

5.3. LIST OF MODULES

- Administrator Module
- User Module
- Tourism And City Guide
- POI Categories

5.3.1 ADMINISTRATOR MODULE

The admin module is the major module as it is responsible for carrying out the major operations regarding site updates, business updates, job alerts etc., It maintains information regarding other four modules. The various software components in administrator module are update alerts, update industries, update hotels, view resumes, update site information. The details entered by admin are complete city history such as political, political leaders and social, Business details such as top companies in the city and its information, Job details such as vacancies and companies profile, Emergency such as phone numbers of with respect to an emergency, Conventional places such as description, location, address and image of the place, regarding news, papers, city channels such which papers are available in the city,

5.3.2 USER MODULE

User module is maintaining the information regarding the city tourist spot, hotels in the city entertainment in this city etc.., the user after registration as a tourist is considered as authorized user. The various software components in the tourist module are view theaters, view hotels, view city map, view ATM locations, view hospitals, view city history, view travel agency, view bus routes.

5.3.3 TOURISM AND CITY GUIDE

It is a section of the application which provides the detailed information about the area, the famous places of those areas, restaurants, hotels, shopping malls etc. and all the related details of these places. This provides the user an easy way to visit any place. It also helps the user to compare any two places before being able to visit it. This makes planning easy.

5.3.4 POI CATEGORIES

The application contains a lot of information about the various categories of nearby facilities and other amenities that a user might need. This is updated based on the geographic location through the use of GPS on the smartphone. So, when a user is in any particular location, they will be shown all the best possible landmarks, gyms, points of interest, famous hotspots etc. This is done live and gets shown in real-time to the user.

5.4 IMPLEMENTATION OF THE PROPOSED SYSTEM BASED ON COLLABORATIVE FILTERING ALGORITHM

Step 1: Collect data: Gather relevant data from multiple sources such as tourist attractions, accommodations, and other relevant factors to construct a comprehensive database for generating recommendations.

Step 2: Preprocess data: Clean and preprocess the collected data to ensure that it is consistent, error-free, and contains no missing values.

Step 3: Define similarity metrics: Establish similarity metrics, such as cosine similarity or Pearson correlation, to calculate the similarity between items and users.

Step 4: Split the data: Divide the data into training and testing sets to evaluate the accuracy of the recommendation system.

Step 5: Build a model: Construct a recommendation model using the collaborative filtering algorithm, which identifies similar users or items based on their preferences and recommends items that they have enjoyed to other users with similar preferences.

Step 6: Evaluate the model: Use various evaluation metrics such as precision, recall, and F1-score to assess the performance of the recommendation system and identify areas for improvement.

Step 7: Tune the parameters: Optimize the parameters of the algorithm to enhance the performance of the recommendation system.

Step 8: Deploy the system: Launch the recommendation system as an Android application with an intuitive interface, allowing tourists to access personalized recommendations based on their interests, preferences, and location.

Step 9: Monitor and update: Monitor the performance of the system and update it regularly to maintain its accuracy and relevance.

Step 10: Test and refine: Test the system with actual users and collect feedback to refine and improve the system to provide a seamless and satisfactory experience for users.

5.5 PSEUDOCODE FOR THE PROPOSED SYSTEM IMPLEMENTATION

```
// Step 1: Collect data
data = gather_data()

// Step 2: Preprocess data
data = preprocess_data(data)

// Step 3: Define similarity metrics
similarity_metric = define_similarity_metric()

// Step 4: Split the data
training_data, testing_data = split_data(data)

// Step 5: Build a model
model = build_model(training_data, similarity_metric)

// Step 6: Evaluate the model
performance_metrics = evaluate_model(model, testing_data)

// Step 7: Tune the parameters
tuned_model = tune_parameters(model)

// Step 8: Deploy the system
deploy_android_app(tuned_model)

// Step 9: Monitor and update
```

```
while (true) {
    performance_metrics = monitor_system()
    if (performance_metrics.degrade) {
        update_system()
    }
}

// Step 10: Test and refine
while (true) {
    user_feedback = get_user_feedback()
    if (user_feedback) {
        refine_system(user_feedback)
    }
}
```

CHAPTER 6

SYSTEM TESTING AND RESULTS

6.1 CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

6.1.1 NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user**.**

6.1.2 VALUE CONVENTION

Value conventions ensure values for variable at any point of time. This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

6.2 TEST CASES

Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements, and design parameters. Although the tests used are primarily functional in nature, non- functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output without any knowledge of the test object's internal structure.

6.3 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware systems. This model was tested with all its components of software and hardware and faced few errors in the initial testing. After altering the code and circuit connections the model passed all the test cases such as connectivity, efficiency, time complexity and much more.

6.4 UNIT TESTING

In computer programming, unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. This model was split into software part and hardware part. Both parts were tested separately and the test cases successfully passed.

6.5 FUNCTIONAL TESTING

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional Testing usually describes what the system does. This model was tested against the specifications created. There were some issues in the deploying of the application. After careful examination of the code and debugging the model was executed successfully passing all the required test cases.

6.6 PERFORMANCE TESTING

In software engineering, performance testing is in general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. This model was tested under various conditions and different kinds of inputs were given. The model threw errors when inappropriate inputs were given. The application was tested under various older verisions of the operating system and it ran successfully

6.7 TESTING TECHNIQUES

6.7.1 TESTING

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding any undiscovered error. A successful test is one that uncovers a yet undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. and represents the ultimate the review of specification design and coding. Any engineering product can be tested in one of the two ways:

• WHITE BOX TESTING

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. White-box testing is a method of testing the application at the level of the source code. The test cases are derived using the design techniques such as control flow testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition/decision coverage. This application successfully passed all the test cases designed for the white box testing.

• BLACK BOX TESTING

Black-box testing is a method of software testing that examines the functionality of an application (e.g. what the software does) without peering into its internal structures or workings. This method of test can be applied to virtually every level of software testing such as unit, integration, system and acceptance. It performs according to specification and all internal components have been adequately exercised. This model passed all the test cases in the black box testing along with unit, integration, system and various other testing.

6.7.2 INTEGRATION TESTING

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. This model deployed as a whole system was tested against all the test cases and issues that evolved were sorted and the system was tested several times until the test cases were all passed.

6.8 RESULT

The travel and tourism industry is growing at a tremendous rate. It is also considered one of the leading industries that serve millions of people across the globe. All this has been possible due to digitization. In this era of smartphones and other gadgets, people are always looking for the best-in-class travel services that enhance their travel experience. On the other hand, businesses belonging to the tourism industry are going all-in to provide top-notch services to the people.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

Early in the twenty first century, recommender systems were created to aid vacationers in the decision-making process and to battle the overload of information that was available at the time. Within the scope of this project, we provided an in-depth analysis of the currently active recommender frameworks in the travel industry. Following that, we presented an additional calculated structure to put the recommender frameworks in place within the travel industry. The goal of our half-breed engineering is to improve the user's experience by pointing out the things that are most important to the user and assisting the user in customizing his or her journey. After the configurations of components that are thought to be significant to the traveler have been selected, our framework will generate a suitable journey by integrating these components using operational project approaches. This will occur once a suitable journey has been constructed. This technical work will be done with cutting-edge technology such as huge data instruments, artificial intelligence techniques, and the internet of things.

7.2 FUTURE ENHANCEMENTS

As interesting directions for future work, we identify the following two lines. First covering access range can be increased, rating system can also be embedded according to user satisfaction. Apart from android, it can also be made for Windows and IOS users. Also, the application can be translated into different languages. Famous landmarks or (POIs) can be shown based on the geographic location of the user. Making a to-do list that connects multiple locations, making it easier to plan out any trip.

8. REFERENCES

[1] O. Artemenko, V. Pasichnyk, N. Kunanec and D. Tabachyshyn, "Using context analysys for providing real time recommendations in e-tourism mobile location-based recommender systems," 2019 IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT), 2019, pp. 166-169, doi: 10.1109/STC-CSIT.2019.8929822

[2] X. Shao, G. Tang and B. -K. Bao, "Personalized Travel Recommendation Based on Sentiment-Aware Multimodal Topic Model," in IEEE Access, vol. 7, pp. 113043-113052, 2019, doi: 10.1109/ACCESS.2019.2935155.

[3] Chen, Liu, Q., & Qiao, X. (2020). Approaching Another Tourism Recommender. 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C). https://doi.org/10.1109/qrs-c51114.2020.00097

[4] Y. Tulashvili, Y. Turbal, D. A. Alkaleg, V. Pasichnyk, A. S. Sumayya Ali and N. Kunanets, "The Optimal Tour Problem in Smart Tourism Recommender Systems," 2020 IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT), 2020, pp. 246-250, doi: 10.1109/CSIT49958.2020.9322043.

[5] Aarab, Z., Elghazi, A., Saidi, R., Rahmani, M.D. (2018). Toward a Smart Tourism Recommender System: Applied to Tangier City. In: Ben Ahmed, M., Boudhir, A. (eds) Innovations in Smart Cities and

Applications. SCAMS 2017. Lecture Notes in Networks and Systems, vol 37. Springer, Cham. https://doi.org/10.1007/978-3-319-74500-8_59

[6] Bornträger, C., Cheverst, K., Davies, N., Dix, A., Friday, A., & Seitz, J. (2003). Experiments with Multi-modal Interfaces in a Context-Aware City Guide. Human-Computer Interaction with Mobile Devices and Services, 116–130. doi:10.1007/978-3-540-45233-1_10

[7] Sowmya, M.R., Prakash, S., Singh, S.K., Maloo, S., Yadav, S. (2019). Smart Tourist Guide (Touristo). In: Shetty, N., Patnaik, L., Nagaraj, H., Hamsavath, P., Nalini, N. (eds) Emerging Research in Computing, Information, Communication and Applications. Advances in Intelligent Systems and Computing, vol 906. Springer, Singapore. https://doi.org/10.1007/978-981-13-6001-5_23

[8]Parameswaran, A.N., Shivaprakasha, K.S., Bhandarkar, R. (2021). Smart Tourism Development in a Smart City: Mangaluru. In: Reddy, A., Marla, D., Favorskaya, M.N., Satapathy, S.C. (eds) Intelligent Manufacturing and Energy Sustainability. Smart Innovation, Systems and Technologies, vol 213. Springer, Singapore. https://doi.org/10.1007/978-981-33-4443-3_31

[9] Yan, X., Juan, Z. (2021). Research and Implementation of Intelligent Tourism Guide System Based on Cloud Computing Platform. In: MacIntyre, J., Zhao, J., Ma, X. (eds) The 2020 International Conference on Machine Learning and Big Data Analytics for IoT Security and Privacy. SPIOT 2020. Advances in Intelligent Systems and Computing, vol 1283. Springer, Cham. https://doi.org/10.1007/978-3-030-62746-1

[10] Santos, F., Almeida, A., Martins, C., Oliveira, P., Gonçalves, R. (2017). Tourism Recommendation System based in User Functionality and Points-of-Interest Accessibility levels. In: Mejia, J., Muñoz, M., Rocha, Á., San Feliu, T., Peña, A. (eds) Trends and Applications in Software Engineering. CIMPS 2016. Advances in Intelligent Systems and Computing, vol 537. Springer, Cham. https://doi.org/10.1007.

[11] P. Yochum, L. Chang, T. Gu and M. Zhu, "Linked Open Data in Location-Based Recommendation System on Tourism Domain: A Survey," in IEEE Access, vol. 8, pp. 16409-16439, 2020, doi: 10.1109/ACCESS.2020.2967120.

[12] K. A. Fararni, F. Nafis, B. Aghoutane, A. Yahyaouy, J. Riffi and A. Sabri, "Hybrid recommender system for tourism based on big data and AI: A conceptual framework," in Big Data Mining and Analytics, vol. 4, no. 1, pp. 47-55, March 2021, doi: 10.26599/BDMA.2020.9020015.

[13] N. Gong, H. Fu and C. Gong, "A Research on the Perception of Authenticity of Self-service Tourists Based on the Background of Smart Tourism," 2021 7th Annual International Conference on Network and Information Systems for Computers (ICNISC), 2021, pp. 826-830, doi: 10.1109/ICNISC54316.2021.00153.

[14] Logesh, R., Subramaniyaswamy, V. (2019). Exploring Hybrid Recommender Systems for Personalized Travel Applications. In: Mallick, P., Balas, V., Bhoi, A., Zobaa, A. (eds) Cognitive Informatics and Soft Computing. Advances in Intelligent Systems and Computing, vol 768.

Springer, Singapore. https://doi.org/10.1007/978-981-13-0617-4_52

[15] Leal, F., Malheiro, B., Burguillo, J.C. (2018). Trust and Reputation Modelling for Tourism Recommendations Supported by Crowdsourcing. In: Rocha, Á., Adeli, H., Reis, L.P., Costanzo, S. (eds) Trends and Advances in Information Systems and Technologies. WorldCIST'18 2018. Advances in Intelligent Systems and Computing, vol 745. Springer, Cham. https://doi.org/10.1007/978-3-319-77703-0_81

[16] G. Hirakawa, G. Satoh, K. Hisazumi and Y. Shibata, "Data Gathering System for Recommender System in Tourism," 2015 18th International Conference on Network-Based Information Systems, 2015, pp. 521-525, doi: 10.1109/NBiS.2015.78.

[17] R. Chen, Q. Hua, Y. -S. Chang, B. Wang, L. Zhang and X. Kong, "A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks," in IEEE Access, vol. 6, pp. 64301-64320, 2018, doi: 10.1109/ACCESS.2018.2877208. L. Amarpuri, N. Yadav, G. Kumar and S. Agrawal, "Prediction of CO2 emissions using deep learning hybrid approach: A Case Study in Indian Context", Twelfth International Conference on Contemporary Computing (IC3), 2019

[18] S. Migliorini, D. Carra and A. Belussi, "Distributing Tourists among POIs with an Adaptive Trip Recommendation System," in IEEE Transactions on Emerging Topics in Computing, vol. 9, no. 4, pp. 1765-1779, 1 Oct.-Dec. 2021, doi: 10.1109/TETC.2019.2920484.

CHAPTER 9
APPENDICES

9.1 SOURCE CODE :

MainActivity.java

```
package com.example.cityguide.activity;
import android.Manifest;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import android.util.Log;
import android.view.MenuItem;
import android.view.View;
import android.view.WindowManager;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.view.GravityCompat;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentTransaction;
import com.bumptech.glide.Glide;
import com.example.cityguide.R;
import com.example.cityguide.circle;
import com.example.cityguide.fragments.fragmentCategory;
import com.example.cityguide.fragments.fragmentAbout;
import com.example.cityguide.fragments.fragmentMap;
import com.example.cityguide.fragments.fragmentHome;
import com.example.cityguide.fragments.fragmentLocation;
import com.example.cityguide.fragments.fragmentProfile;
import com.example.cityguide.fragments.fragmentSaved;
import com.example.cityguide.modals.Users;
import com.example.cityguide.sign.SignInActivity;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.navigation.NavigationView;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class MainActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener {
    ImageView imagevew;
    TextView name;
    DatabaseReference rootref;
    FirebaseAuth mauth;
    FirebaseUser firebaseUser;
    public static Location currentLocation;
    FusedLocationProviderClient fusedLocationProviderClient;
    private static final int REQUEST_CODE = 101;
    NavigationView navigationView;
    androidx.appcompat.widget.Toolbar toolbar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREE
N, WindowManager.LayoutParams.FLAG_FULLSCREEN);
        setContentView(R.layout.activity_main);
        mauth = FirebaseAuth.getInstance();
        toolbar        =        (androidx.appcompat.widget.Toolbar)
findViewById(R.id.toolbar);
        toolbar.setTitle("MY PLACE");
        navigationView = (NavigationView) findViewById(R.id.nav_view);
        View hView = navigationView.getHeaderView(0);
        imagevew = (ImageView) hView.findViewById(R.id.imageView);
        name = (TextView) hView.findViewById(R.id.menu_header_name);
        fusedLocationProviderClient                                  =
LocationServices.getFusedLocationProviderClient(this);

        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
            this,    drawer,    toolbar,    R.string.navigation_drawer_open,
R.string.navigation_drawer_close);
        drawer.setDrawerListener(toggle);
        toggle.syncState();
        navigationView.setNavigationItemSelectedListener(this);
        //showing default fragment
        displaySelectedFragment(R.id.nav_home);
        getRef();
        currentlocation();
    }
    @Override
    public void onBackPressed() {
        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer);
        if (drawer.isDrawerOpen(GravityCompat.START)) {
            drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
        }
    }
    void getRef() {
        firebaseUser = FirebaseAuth.getInstance().getCurrentUser();
        rootref = FirebaseDatabase.getInstance().getReference("Users");
        rootref.addValueEventListener(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                for (DataSnapshot dataSnapshot : snapshot.getChildren()) {
                    Users user = dataSnapshot.getValue(Users.class);

Glide.with(getApplicationContext()).load(user.getImageUrl().toString()).trans
form(new circle(getApplicationContext())).into(imagevew);
                    name.setText("Zaid");
                    Log.d("name", firebaseUser.getEmail());
                }
            }
            @Override
            public void onCancelled(@NonNull DatabaseError error) {
            }
        });
    }
    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem item) {
        // item id is being passed into the method here
        displaySelectedFragment(item.getItemId());

        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer);
        drawer.closeDrawer(GravityCompat.START);
        return true;
    }
    public void setScreenTitle(int item_id) {
        String title = "";
        switch (item_id) {
            case R.id.nav_home:
                title = "Home";
                break;
            case R.id.nav_category:
                title = "Category";
                break;
            case R.id.nav_Location:
                title = "Location";
                break;
            case R.id.nav_save:
                title = "Saved Location";
                break;
            case R.id.nav_profile:
                title = "Profile";
                break;
            case R.id.nav_logout:
                title = "Three";
                break;
            case R.id.nav_map:
                title = "MAP";
                break;
            case R.id.nav_about:
                title = "About";
                break;
        }
        toolbar.setTitle(title);
    }
    public void displaySelectedFragment(int item_id) {
        Fragment fragment = null;
        switch (item_id) {
            case R.id.nav_home:
                fragment = new fragmentHome();
                navigationView.getMenu().getItem(0).setChecked(true);
                break;

            case R.id.nav_category:
                fragment = new fragmentCategory();
                navigationView.getMenu().getItem(1).setChecked(true);
                break;

            case R.id.nav_Location:
                fragment = new fragmentLocation();
                navigationView.getMenu().getItem(2).setChecked(true);
                break;
            case R.id.nav_save:
                fragment = new fragmentSaved();
                navigationView.getMenu().getItem(3).setChecked(true);
                break;
            case R.id.nav_profile:
                fragment = new fragmentProfile();
                navigationView.getMenu().getItem(4).setChecked(true);
                break;
            case R.id.nav_logout:
                logOut();
                navigationView.getMenu().getItem(5).setChecked(true);
                break;
            case R.id.nav_map:
                fragment = new fragmentMap();
                break;
            case R.id.nav_about:
                fragment = new fragmentAbout();
```

```
            break;
        }
        if (fragment != null) {
            FragmentTransaction ft =
getSupportFragmentManager().beginTransaction();
            //this is where the id of the FrameLayout is being mentioned. Hence
the fragment would be loaded into the framelayout
            ft.replace(R.id.container, fragment);
            ft.commit();
        }
        /** setting title to the screen **/
        setScreenTitle(item_id);
    }
    private void logOut() {
        AlertDialog.Builder builder = new
AlertDialog.Builder(MainActivity.this);
        builder.setMessage("Do you want to Log Out ?");
        builder.setTitle("LOG OUT");
        builder.setCancelable(false);
        builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
            @Override
            public void onClick(DialogInterface dialog, int which) {
                outofScreen();
            }
        });
        builder.setNegativeButton("No", new DialogInterface.OnClickListener()
{
```

```
            @Override
            public void onClick(DialogInterface dialog,
                                int which) {
                dialog.cancel();
            }
        });
        AlertDialog alertDialog = builder.create();
        alertDialog.show();
    }
    private void outofScreen() {
        FirebaseAuth.getInstance().signOut();
        Intent intent = new Intent(MainActivity.this, SignInActivity.class);
        startActivity(intent);
    }
    void currentlocation(){
        if(ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(
                this, Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION},
REQUEST_CODE);
            return;
        }
        Task<Location> task = fusedLocationProviderClient.getLastLocation();
        task.addOnSuccessListener(new OnSuccessListener<Location>() {
            @Override
            public void onSuccess(Location location) {
                if (location != null) {
                    currentLocation = location;
                    Log.d("lan", String.valueOf(currentLocation.getLatitude()));
                    Log.d("lot", String.valueOf(currentLocation.getLongitude()));
                }
            }
        });
    }
    @Override
    public void onRequestPermissionsResult ( int requestCode, @NonNull
String[] permissions,@NonNull int[] grantResults){
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
        switch (requestCode) {
            case REQUEST_CODE:
                if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                    currentlocation();
                }
                break;
        }
    }
   public  String getLatitude(){
        return String.valueOf(currentLocation.getLatitude());
    }
    public  String getLongitude(){
        return String.valueOf(currentLocation.getLongitude());
    }
    public  Location getLoction(){
        return  currentLocation;
    }
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:versionCode="1"
    android:versionName="1.0"
    android:compileSdkVersion="31"
    android:compileSdkVersionCodename="12"
    package="com.example.cityguide"
    platformBuildVersionCode="31"
    platformBuildVersionName="12">
    <uses-sdk
        android:minSdkVersion="26"
        android:targetSdkVersion="31" />
    <uses-permission
        android:name="android.permission.INTERNET" />
    <uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission
        android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
    <uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
    <uses-feature
        android:glEsVersion="0x20000"
        android:required="true" />
    <application
        android:theme="@ref/0x7f1101c0"
        android:label="@ref/0x7f10001d"
        android:icon="@ref/0x7f0700a5"
        android:debuggable="true"
        android:allowBackup="true"
        android:supportsRtl="true"
        android:extractNativeLibs="false"
        android:roundIcon="@ref/0x7f0d0001"
android:appComponentFactory="androidx.core.app.CoreComponentFactory">
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@ref/0x7f09000a" />
        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyBfbv99FCA24Mw6i9iE9vfMGCbJNULfmrM"
/>
        <activity
            android:theme="@ref/0x7f1101bf"
            android:name="com.example.cityguide.activity.SplashActivity">
            <intent-filter>
              <action
                android:name="android.intent.action.MAIN" />
              <category
                android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:theme="@ref/0x7f1101be"
            android:name="com.example.cityguide.sign.SignInActivity" />
        <activity
            android:theme="@ref/0x7f1101be"
            android:name="com.example.cityguide.sign.signupActivity" />
        <activity
            android:theme="@ref/0x7f1101be"
            android:name="com.example.cityguide.sign.UserActivity" />
        <activity
            android:theme="@ref/0x7f11000a"
            android:name="com.example.cityguide.activity.MainActivity" />
        <activity
            android:theme="@ref/0x7f11000a"
            android:name="com.example.cityguide.activity.categoryActivity" />
```

```
<activity
   android:theme="@ref/0x7f11000a"
   android:name="com.example.cityguide.activity.detailActivity" />
<activity
   android:theme="@ref/0x7f11000a"
   android:name="com.example.cityguide.activity.viewActivity" />
<activity
   android:theme="@ref/0x01030010"
   android:name="com.google.firebase.auth.internal.GenericIdpActivity"
   android:exported="true"
   android:excludeFromRecents="true"
   android:launchMode="2">
   <intent-filter>
      <action
         android:name="android.intent.action.VIEW" />
      <category
         android:name="android.intent.category.DEFAULT" />
      <category
         android:name="android.intent.category.BROWSABLE" />
      <data
         android:scheme="genericidp"
         android:host="firebase.auth"
         android:path="/" />
   </intent-filter>
</activity>
<activity
   android:theme="@ref/0x01030010"
   android:name="com.google.firebase.auth.internal.RecaptchaActivity"
```

```
            android:exported="true"
            android:excludeFromRecents="true"
            android:launchMode="2">

            <intent-filter>

                <action
                    android:name="android.intent.action.VIEW" />

                <category
                    android:name="android.intent.category.DEFAULT" />

                <category
                    android:name="android.intent.category.BROWSABLE" />

                <data
                    android:scheme="recaptcha"
                    android:host="firebase.auth"
                    android:path="/" />
            </intent-filter>
        </activity>
        <service
android:name="com.google.firebase.auth.api.fallback.service.FirebaseAuthFal
lbackService"
            android:enabled="true"
            android:exported="false">
            <intent-filter>
            <action
          android:name="com.google.firebase.auth.api.gms.service.START" />
            <category
                android:name="android.intent.category.DEFAULT" />
          </intent-filter>
        </service>
        <service
android:name="com.google.firebase.components.ComponentDiscoveryServic
e"
          android:exported="false"
          android:directBootAware="true">
          <meta-data
android:name="com.google.firebase.components:com.google.firebase.auth.Fir
ebaseAuthRegistrar"
     android:value="com.google.firebase.components.ComponentRegistrar" />
          <meta-data
android:name="com.google.firebase.components:com.google.firebase.databas
e.DatabaseRegistrar"
    android:value="com.google.firebase.components.ComponentRegistrar" />
          <meta-data
android:name="com.google.firebase.components:com.google.firebase.firestor
e.FirestoreRegistrar"
android:value="com.google.firebase.components.ComponentRegistrar" />
          <meta-data
android:name="com.google.firebase.components:com.google.firebase.storage.
StorageRegistrar"
android:value="com.google.firebase.components.ComponentRegistrar" />
        </service>
        <uses-library
           android:name="org.apache.http.legacy"
           android:required="false" />
        <activity
           android:theme="@ref/0x01030010"
android:name="com.google.android.gms.auth.api.signin.internal.SignInHubA
ctivity"
           android:exported="false"
           android:excludeFromRecents="true" />
        <service
android:name="com.google.android.gms.auth.api.signin.RevocationBoundSer
vice"
android:permission="com.google.android.gms.auth.api.signin.permission.RE
VOCATION_NOTIFICATION"
           android:exported="true"
           android:visibleToInstantApps="true" />
        <provider
           android:name="com.google.firebase.provider.FirebaseInitProvider"
           android:exported="false"
           android:authorities="com.example.cityguide.firebaseinitprovider"
           android:initOrder="100"
           android:directBootAware="true" />
        <activity
           android:theme="@ref/0x01030010"
android:name="com.google.android.gms.common.api.GoogleApiActivity"
           android:exported="false" />
    </application></manifest>
```

9.2 SCREENSHOTS

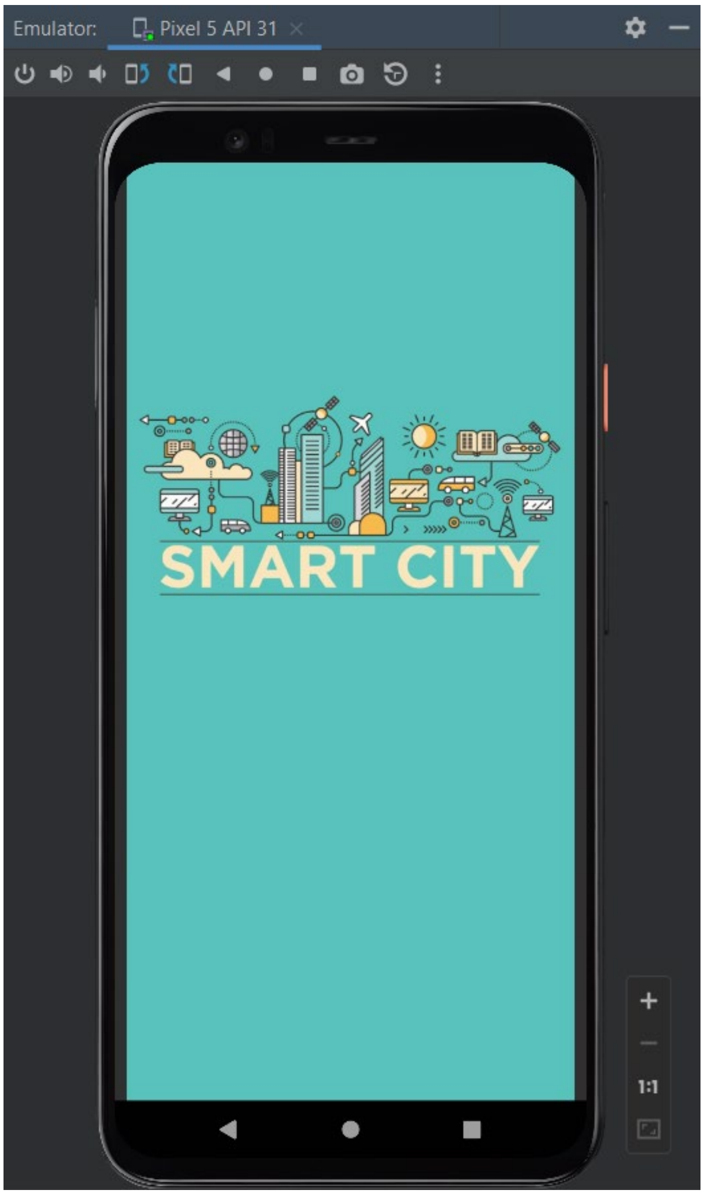


FIG 9.1.1 WELCOME SCREEN

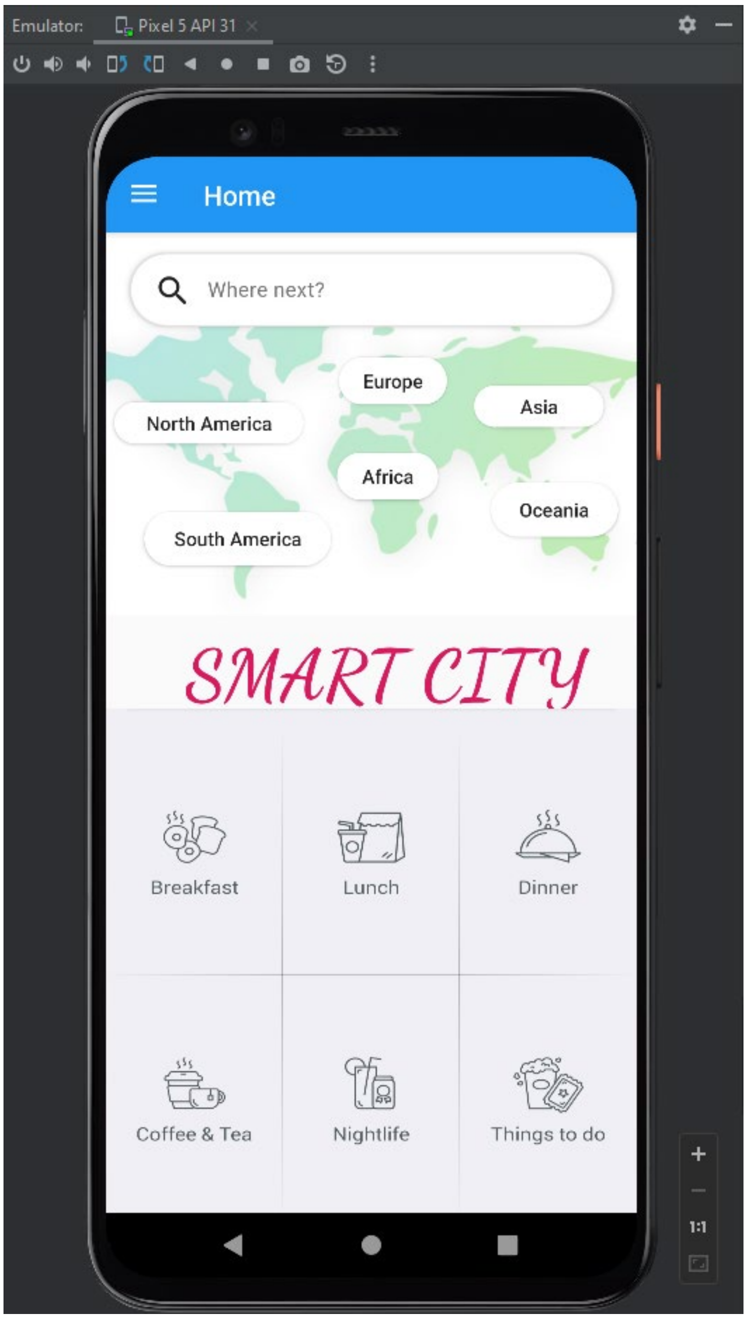


FIG 9.1.2. HOMESCREEN

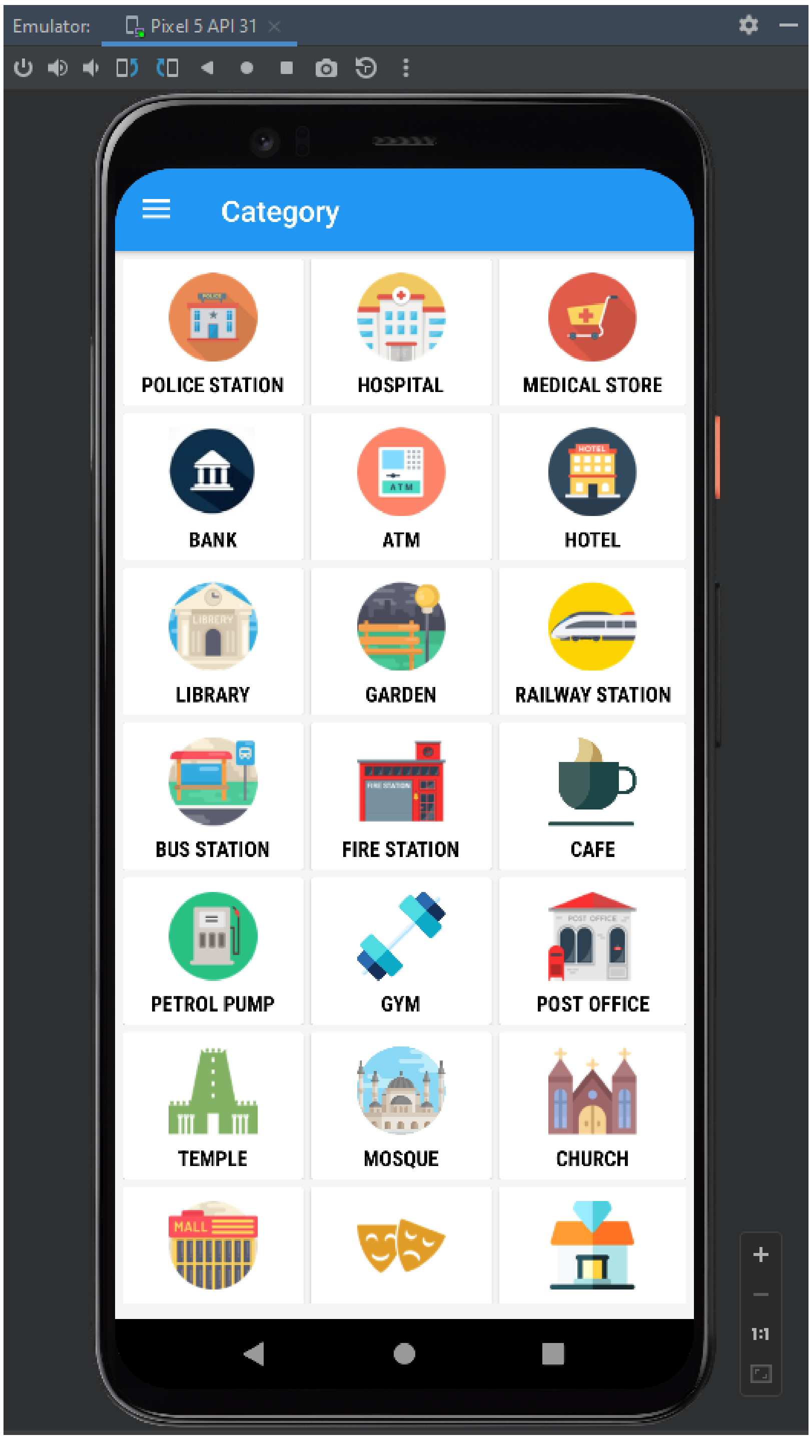


FIG 9.1.3. CATEGORIES

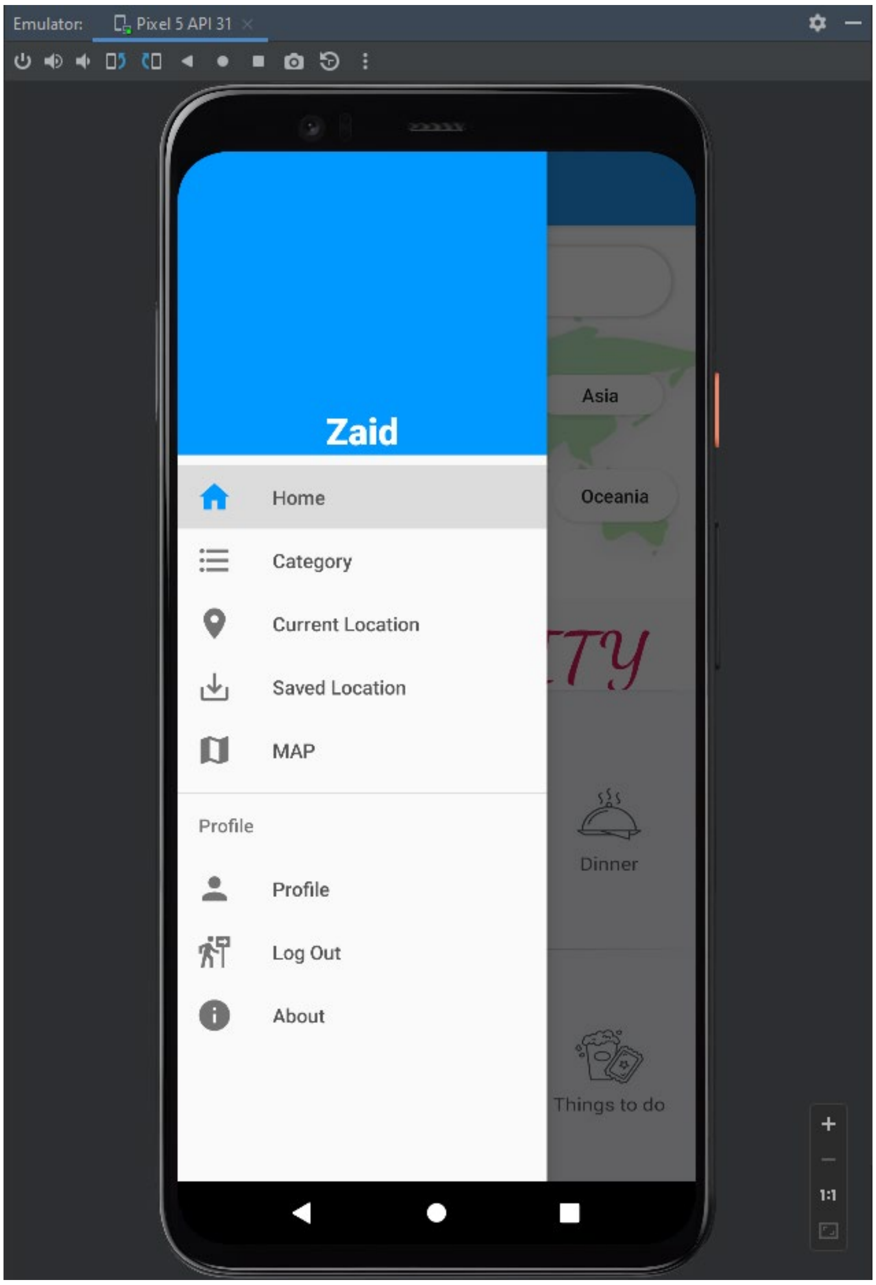


FIG 9.1.4. SIDEBAR

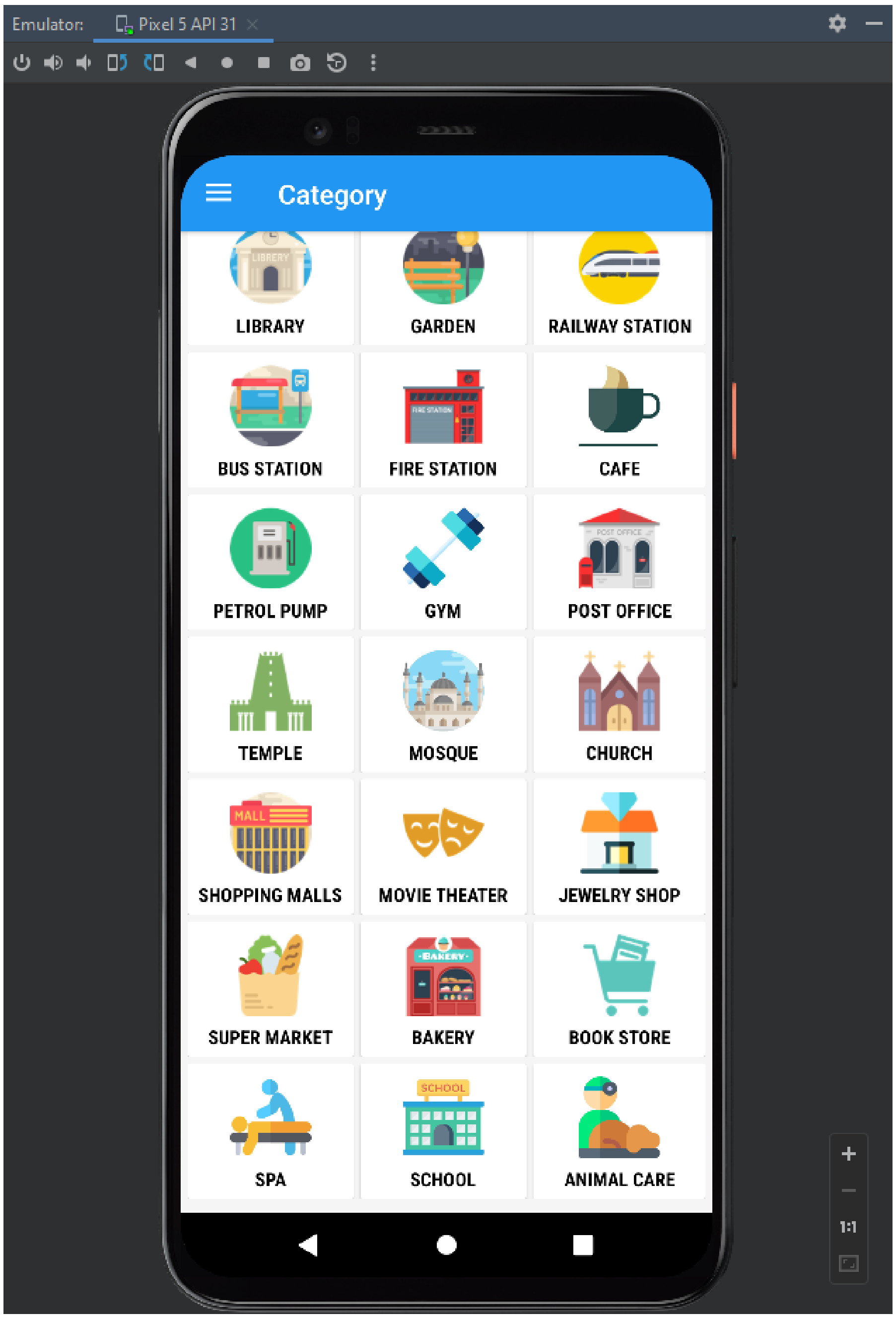
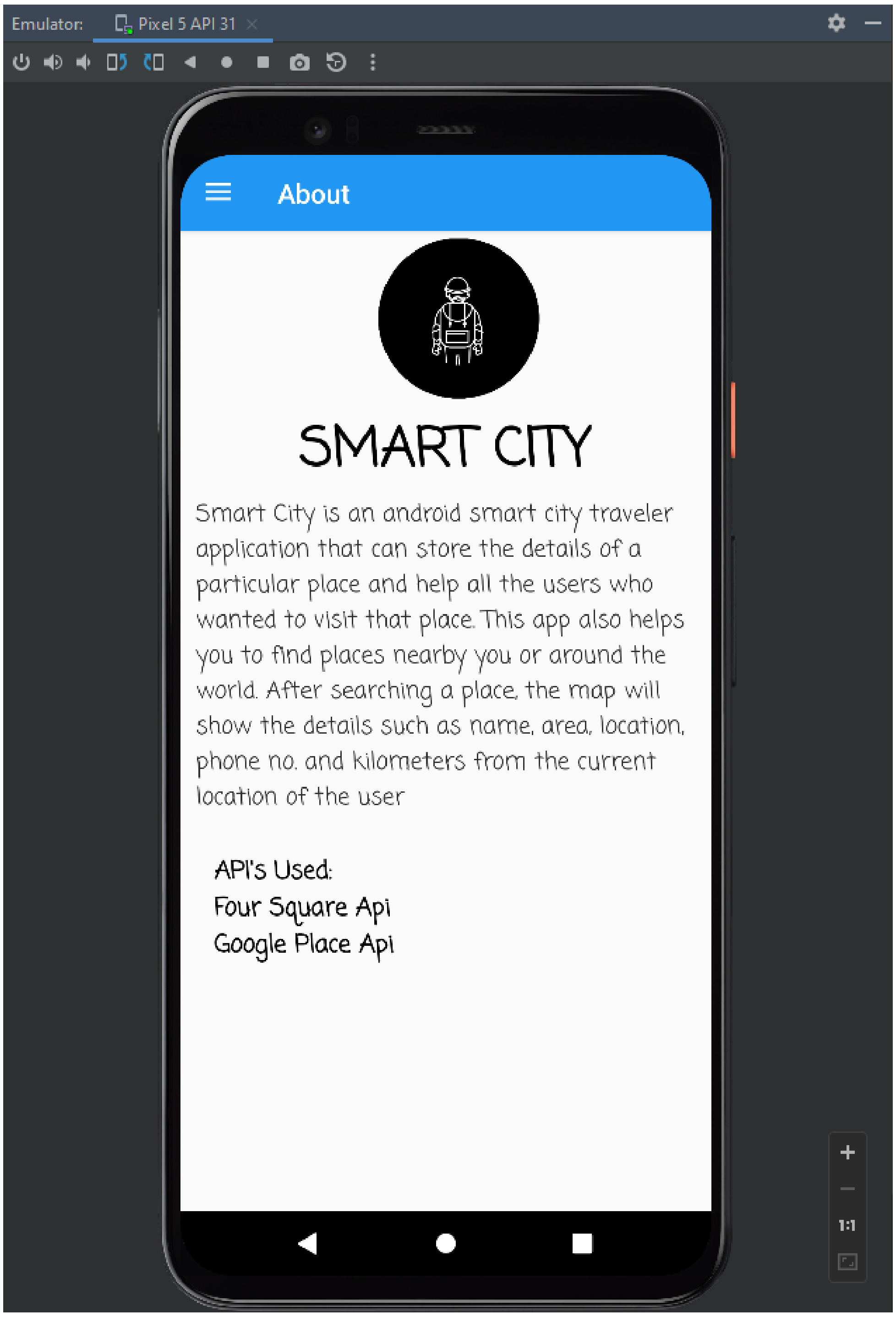


FIG 9.1.5 CATEGORIES 2

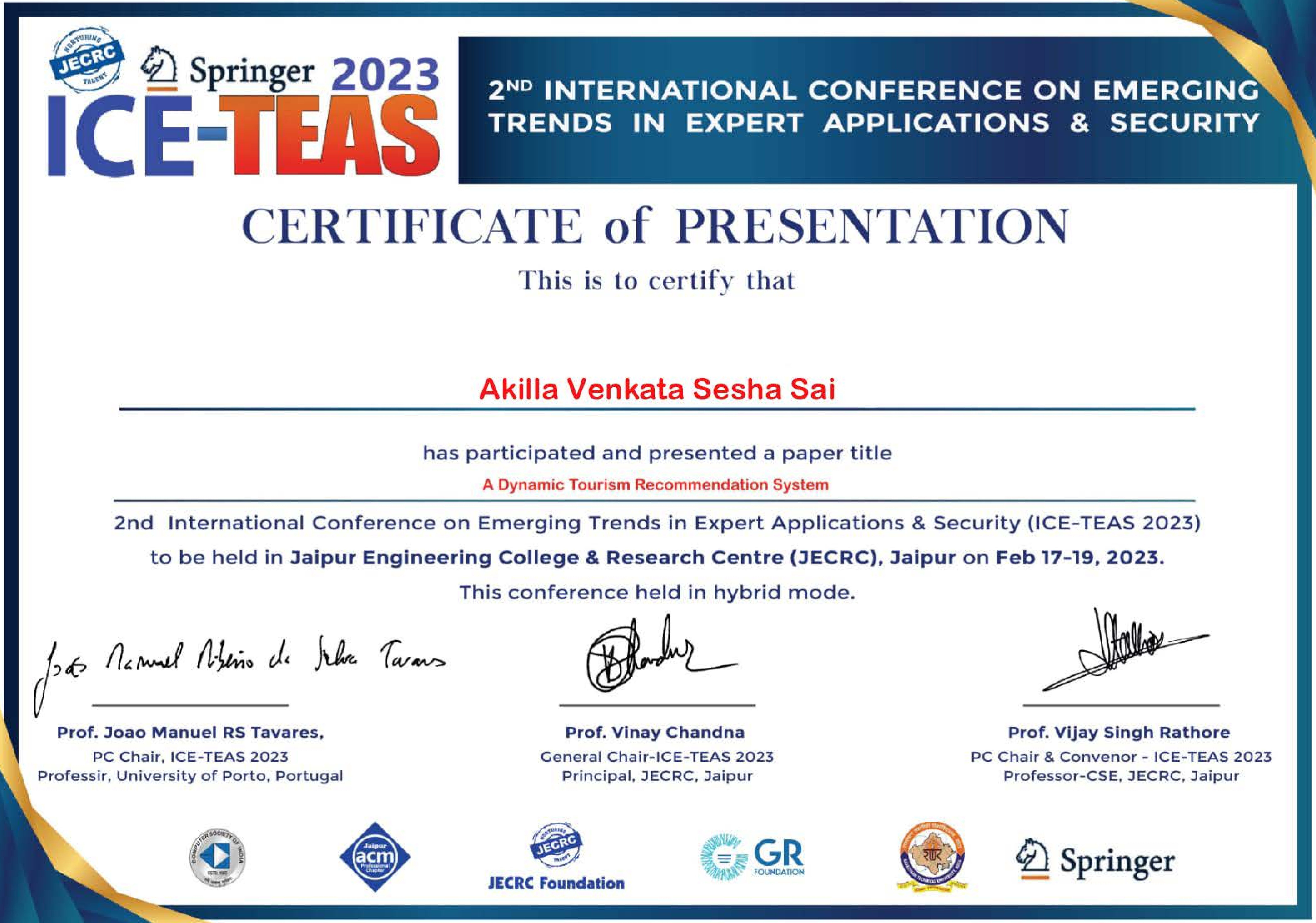


SMART CITY

Smart City is an android smart city traveler application that can store the details of a particular place and help all the users who wanted to visit that place. This app also helps you to find places nearby you or around the world. After searching a place, the map will show the details such as name, area, location, phone no. and kilometers from the current location of the user

API's Used:
Four Square Api
Google Place Api

FIG 9.1.6 ABOUT PAGE



Springer 2023 ICE-TEAS

2ND INTERNATIONAL CONFERENCE ON EMERGING TRENDS IN EXPERT APPLICATIONS & SECURITY

CERTIFICATE of PRESENTATION

This is to certify that

Akilla Venkata Sesha Sai

has participated and presented a paper title

A Dynamic Tourism Recommendation System

2nd International Conference on Emerging Trends in Expert Applications & Security (ICE-TEAS 2023) to be held in **Jaipur Engineering College & Research Centre (JECRC), Jaipur** on **Feb 17-19, 2023.**

This conference held in hybrid mode.

[illegible] | [illegible] | [illegible]

Prof. Joao Manuel RS Tavares,
PC Chair, ICE-TEAS 2023
Professir, University of Porto, Portugal

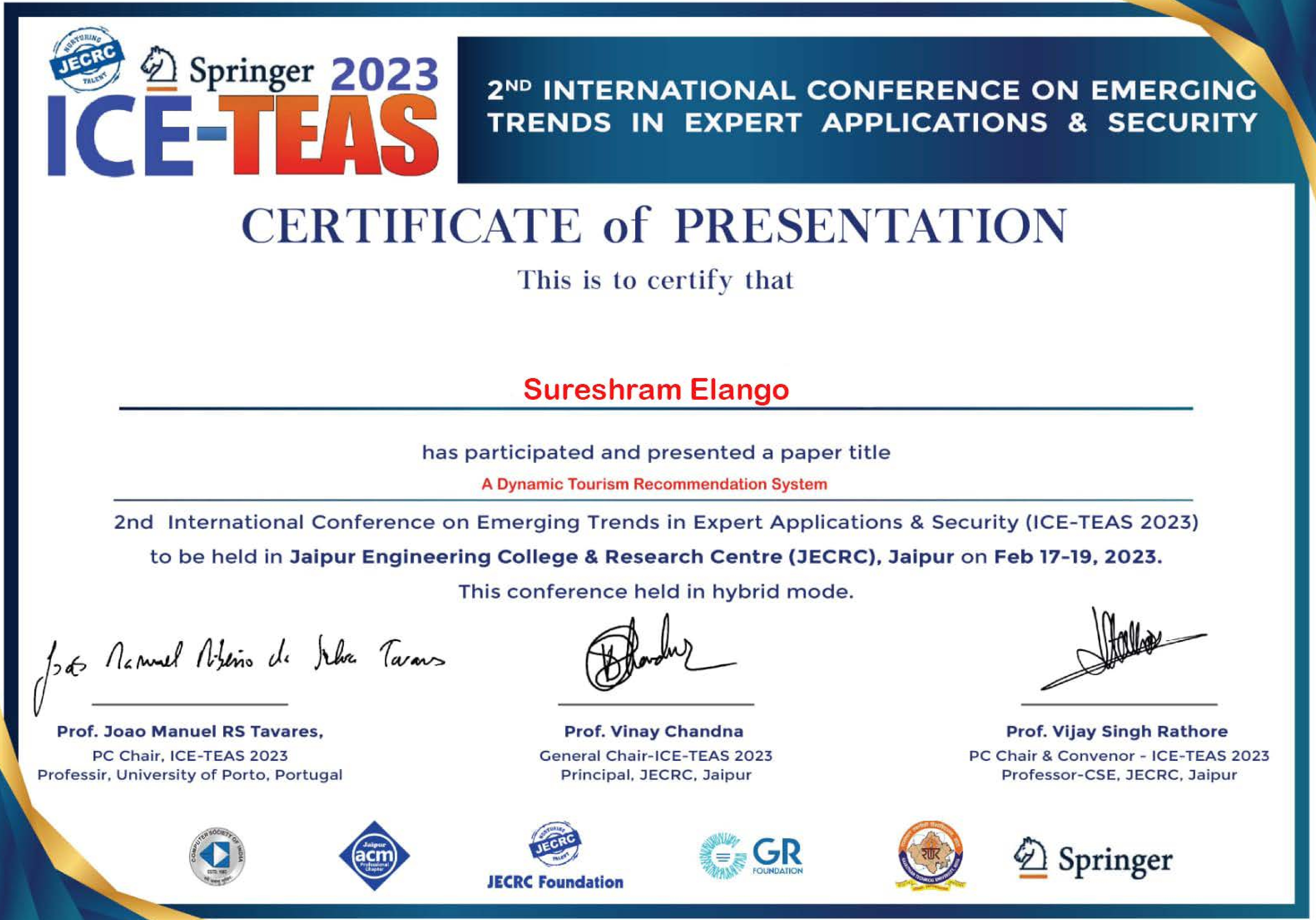
Prof. Vinay Chandna
General Chair-ICE-TEAS 2023
Principal, JECRC, Jaipur

Prof. Vijay Singh Rathore
PC Chair & Convenor - ICE-TEAS 2023
Professor-CSE, JECRC, Jaipur

JECRC Foundation

GR FOUNDATION

Springer



Springer 2023 ICE-TEAS

2ND INTERNATIONAL CONFERENCE ON EMERGING TRENDS IN EXPERT APPLICATIONS & SECURITY

CERTIFICATE of PRESENTATION

This is to certify that

Sureshram Elango

has participated and presented a paper title

A Dynamic Tourism Recommendation System

2nd International Conference on Emerging Trends in Expert Applications & Security (ICE-TEAS 2023) to be held in **Jaipur Engineering College & Research Centre (JECRC), Jaipur** on **Feb 17-19, 2023.**

This conference held in hybrid mode.

[illegible]

Prof. Joao Manuel RS Tavares,
PC Chair, ICE-TEAS 2023
Professir, University of Porto, Portugal

[illegible]

Prof. Vinay Chandna
General Chair-ICE-TEAS 2023
Principal, JECRC, Jaipur

[illegible]

Prof. Vijay Singh Rathore
PC Chair & Convenor - ICE-TEAS 2023
Professor-CSE, JECRC, Jaipur

JECRC Foundation

GR FOUNDATION

Springer

JECRC Springer 2023 ICE-TEAS

2ND INTERNATIONAL CONFERENCE ON EMERGING TRENDS IN EXPERT APPLICATIONS & SECURITY

CERTIFICATE of PRESENTATION

This is to certify that

Mohammed Zaid

has participated and presented a paper title

A Dynamic Tourism Recommendation System

2nd International Conference on Emerging Trends in Expert Applications & Security (ICE-TEAS 2023) to be held in **Jaipur Engineering College & Research Centre (JECRC), Jaipur** on **Feb 17-19, 2023.**

This conference held in hybrid mode.

[illegible]	[illegible]	[illegible]
Prof. Joao Manuel RS Tavares,	**Prof. Vinay Chandna**	**Prof. Vijay Singh Rathore**
PC Chair, ICE-TEAS 2023	General Chair-ICE-TEAS 2023	PC Chair & Convenor - ICE-TEAS 2023
Professir, University of Porto, Portugal	Principal, JECRC, Jaipur	Professor-CSE, JECRC, Jaipur

JECRC Foundation

GR FOUNDATION

Springer