**ANDROID APPLICATION FOR BUS TRACKING AND TICKETING**

**A Project Report**

Submitted in partial fulfillment of the

Requirements for the award of the Degree of

### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

**By**

Joehan Misquitta

19021

**Under the esteemed guidance of**

**Mr. Akshay Nimkar**

Logo

Description automatically generated

**DEPARTMENT OF INFORMATION TECHNOLOGY**

### RISHI DAYARAM AND SETH HASSARAM NATIONAL COLLEGE

### AND SETH WASSIAMULL ASSOMUL SCIENCE COLLEGE

***(Affiliated to University of Mumbai)***

**MUMBAI, 400050**

**MAHARASHTRA 2021**

### RISHI DAYARAM AND SETH HASSARAM NATIONAL COLLEGE

### AND SETH WASSIAMULL ASSOMUL SCIENCE COLLEGE,

***(Affiliated to University of Mumbai)***

**MUMBAI-MAHARASHTRA-400050**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

Logo

Description automatically generated

### CERTIFICATE

This is to certify that the project entitled, **"** **ANDROID APPLICATION FOR BUS TRACKING AND TICKETING"**, is bonafied work of **JOEHAN MISQUITTA** bearing Seat. No: **19021** submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

#### **Internal Guide** Coordinator

**External Examiner**

**Date: College Seal**

## Abstract

In major Cities like Mumbai the bus plays a significant role in helping citizens commute to their work or college. The problem arises when they are not on time and delays every one’s schedule. This project aims to develop an android application to solve this problem by helping citizens track the current location of the bus using their phones. Everyone including the driver who installs this app, signup using their Google account and permits location access can be used like a node to track the bus they currently are on. This will help the commuters decide between waiting for the bus or go with an alternative mode of transport. Additionally, commuters can buy tickets or the bus pass online and verify them with the Conductor using a QR code this also helps in keeping a count of how many travelers are on the bus.

Working of the application: Users installs the app on their phone, registers using Google account and allows the app access to their location. When a user wants to either track or travel on a particular bus, they search for it on the home map screen. When they board the bus, and their current location appears to be on the same route the bus is supposed to travel the are used as a node to track the bus. If they have purchased a ticket on the app or use the app pass the can access the QR code to verify the ticket/pass with the Conductor.

### ACKNOWLEDGEMENT

I take this opportunity to express my deepest gratitude to the Department of B.Sc.I.T. for nurturing and teaching us the nuances required for developing his project. Our Head of Department Mr. Dinesh Himatsinghani has been very helpful for making all facilities available for us. They always have watchful eye on the progress throughout our project. I am immensely thankful to them for their valuable and timely inputs.

Once again, I am grateful to my guide Mr. Akshay Nimkar without whom this project would be impossible. I owe her my greatest appreciation. I would like to thank her for the guidance, support, patience, and faith in our capabilities and giving me the flexibility in terms of working and reporting schedules.

This Project was a great learning and humbling experience and reflects my academic work at National College. I am sure that this experience will hold me a good stead when I begin my professional careers.

### DECLARATION

I hereby declare that the project entitled, “**ANDROID APPLICATION FOR BUS TRACKING AND TICKETING**” done at **Mumbai**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of

#### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be submitted as final semester project as part of our curriculum.

**Name and Signature of the Student**

### 

### TABLE OF CONTENTS

## Chapter 1: INTRODUCTION

## Chapter 2: SURVEY OF TECHNOLOGIES

## Chapter 3: REQUIREMENTS AND ANALYSIS

## Chapter 4: SYSTEM DESIGN

## Chapter 5: IMPLEMENTATION AND TESTING

## Chapter 6: RESULTS AND DISCUSSION

## Chapter 7: CONCLUSIONS

## Chapter 8: REFERENCES

## Tables of Figures

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.No** |  | **Contents** | **Pg.no** |
| 1. |  | INTRODUCTION |  |
|  | 1.1 | Background |  |
|  | 1.2 | Objective |  |
|  | 1.3 | Purpose, Scope, and Applicability |  |
|  | 1.3.1 | Purpose |  |
|  | 1.3.2 | Scope |  |
|  | 1.3.3 | Applicability |  |
|  | 1.4 | Achievements |  |
|  | 1.5 | Organisation of Report |  |
|  |  |  |  |
| 2. |  | SURVEY OF TECHNOLOGIES |  |
|  |  |  |  |
| 3. |  | REQUIREMENTS AND ANALYSIS |  |
|  | 3.1 | Problem Definition |  |
|  | 3.2 | Requirements Specifications |  |
|  | 3.3 | Planning and Scheduling |  |
|  | 3.4 | Software and Hardware Requirements |  |
|  | 3.5 | Preliminary Product Description |  |
|  | 3.6 | Conceptual Models |  |
|  |  |  |  |
| 4. |  | SYSTEM DESIGN |  |
|  | 4.1 | Basic Modules |  |
|  | 4.2 | Data Design |  |
|  | 4.2.1 | Schema Design |  |
|  | 4.2.2 | Data Integrity and Constraints |  |
|  | 4.3 | Procedural Design |  |
|  | 4.3.1 | Logic Diagrams |  |
|  | 4.4 | User Interface Design |  |
|  | 4.5 | Security Issues |  |
|  | 4.6 | Test Cases Design |  |
|  |  |  |  |
| 5. |  | IMPLEMENTATION AND TESTING |  |
|  | 5.1 | Implementation Approaches |  |
|  | 5.2 | Coding Details and Code Efficiency |  |
|  | 5.2.1 | Code Efficiency |  |
|  | 5.3 | Testing Approach |  |
|  | 5.3.1 | Unit Testing |  |
|  | 5.3.2 | Integrated Testing |  |
|  | 5.3.3 | Beta Testing |  |
|  | 5.4 | Test Cases |  |
|  |  |  |  |
| 6. |  | RESULTS AND DISCUSSIONS |  |
|  | 6.1 | Test Reports |  |
|  | 6.2 | User Documentation |  |
|  |  |  |  |
| 7. |  | CONCLUSIONS |  |
|  | 7.1 | Conclusion |  |
|  | 7.1.1 | Significance of the System |  |
|  | 7.2 | Limitations of the System |  |
|  | 7.3 | Future Scope of the Project |  |
|  |  |  |  |
| 8. |  | REFERENCES |  |

**Chapter 1**

## Introduction

* 1. **Background:**

For Metro Cities like Mumbai, Public Transport is backbone of their economies because people use them to go to their workplaces, colleges, schools, shopping centers, etc. One part of the public transport system is the Bus, probably one cheapest mode of transport it helps people travel to places in a city where trains are not available or required, but it is also used as a last mile travel solution by people who use the train, But some problems with the bus is they are delayed most the time and it hinders with everyone’s schedule, even if they arrive at the stop they are at full capacity, this is worse when it’s the peak hours of travelling. This project aims to develop a solution to these problems by developing an android application to help commuters track the location of the buses help them make decisions if they want to wait for the bus or take an alternative mode of transport.

* 1. **Objective:**

The Objective of this Project is to Create an application to help users track the current location of a bus to help them make decisions regarding the mode they use to travel around the city, but it will also help users buy the bus tickets and passes in the app so that they don’t have to worry about carrying around cash or coins for the tickets which are inconvenient and heavy to carry around, and they also will not need to go to the depot to buy passes which is inconvenient in general.

* 1. **Purpose, Scope and Applicability:**
     1. **Purpose:**

The purpose of this application is to track buses to help commuters decide between travelling with the bus or an alternative mode of transport if the buses are delayed, commuters can also buy tickets and passes from anywhere and even before they get on the bus.

* + 1. **Scope:**
* While the application is designed to help track buses in the public transport system, it can also be used by private organization who use buses to transport their workers.
* The application is user friendly to help make the application accessible to everyone who own a smartphone and has a google account.
  + 1. **Applicability:**

Here applicability refers to the how applicable to the real world the application is,

* Users can use this to track buses.
* It will tell users the estimated time it will take a bus to reach the stop they are on.
* Users can also search for bus stops or depots nearby using the explore function.
* Users can use the app to buy tickets to the bus and they can authenticate the ticket using a QR code.
* Users can also buy a pass for the bus for any given duration that can be authenticate using a QR Code.
  1. **Achievements:**

During the Making of this Project, I learned Android App Development and Learnt to implement Apps Using Google Maps SDK API.

* 1. **Organization of the Report:**

In this report we discuss about system analysis, system design, implementation and testing, results and discussion, conclusion and future work and references

1. **System analysis**:

In this Section, We Discuss about technologies used in this project.

1. **Requirements and Analysis:**

This section Discusses about Problem definition, Requirements Specification, Planning and Scheduling, Software and Hardware Requirements and Conceptual Models.

1. **System Design:**

This Section includes Describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudocode, and other documentation.

1. **Implementation and Testing:**

This Section includes core segments of code, Testing Methodologies and Testing Approach (i.e., Unit Testing and integration testing).

1. **Results and Discussion:**

This Section Discusses the results of implementation and testing of the system in the project.

1. **Conclusion and Future Work:**

This Section discusses the conclusion of the report and possible future work including further development of the system.

1. **References:**

This Section includes references to sources of information used in this report.

**Chapter 2**

### System Analysis

1. **Android**

Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google. It was unveiled in November 2007, with the first commercial Android device, the HTC Dream, being launched in September 2008.

It is free and open-source software; its source code is known as Android Open-Source Project (AOSP), which is primarily licensed under the Apache License. However most Android devices ship with additional proprietary software pre-installed, most notably Google Mobile Services (GMS) which includes core apps such as Google Chrome, the digital distribution platform Google Play and associated Google Play Services development platform.

About 70 percent of Android smartphones run Google's ecosystem; some with vendor-customized user interface and software suite, such as TouchWiz and later One UI by Samsung, and HTC Sense. Competing Android ecosystems and forks include Fire OS (developed by Amazon) or LineageOS. However, the "Android" name and logo are trademarks of Google which impose standards to restrict "uncertified" devices outside their ecosystem to use Android branding.

The source code has been used to develop variants of Android on a range of other electronics, such as game consoles, digital cameras, portable media players, PCs and others, each with a specialized user interface. Some well-known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google. Software packages on Android, which use the APK format, are generally distributed through proprietary application stores like Google Play Store, Samsung Galaxy Store, Huawei AppGallery, Cafe Bazaar, and GetJar, or open-source platforms like Aptoide or F-Droid.

Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013. As of May 2021, it has over three billion monthly active users, the largest installed base of any operating system, and as of January 2021, the Google Play Store features over 3 million apps. The current stable version is Android 11, released on September 8, 2020.

1. **Kotlin**

Kotlin is a cross-platform, statically typed, general-purpose programming language with type inference. Kotlin is designed to interoperate fully with Java, and the JVM version of Kotlin's standard library depends on the Java Class Library, but type inference allows its syntax to be more concise. Kotlin mainly targets the JVM, but also compiles to JavaScript (e.g., for frontend web applications using React) or native code (via LLVM); e.g., for native iOS apps sharing business logic with Android apps. Language development costs are borne by JetBrains, while the Kotlin Foundation protects the Kotlin trademark.

On 7 May 2019, Google announced that the Kotlin programming language is now its preferred language for Android app developers. Since the release of Android Studio 3.0 in October 2017, Kotlin has been included as an alternative to the standard Java compiler. The Android Kotlin compiler produces Java 8 bytecode by default (which runs in any later JVM), but lets the programmer choose to target Java 9 up to 16, for optimization, or allows for more features; has bidirectional record class interoperability support for JVM, introduced in Java 16, considered stable as of Kotlin 1.5.

Kotlin support for compilation directly to JavaScript (i.e. the classic back-end) is considered stable since Kotlin 1.3 by its developers, while the new Kotlin/JS(IR) is in beta as of version 1.5.30. The new optimized implementations of Kotlin/JVM(IR) and Kotlin/JS (IR-based) were introduced in version 1.4. Kotlin/JVM(IR) is considered stable as of the version 1.5 release. Kotlin/Native (for e.g., Apple silicon support) has been considered beta since version 1.3.

1. **Google Maps SDK API:**

With the Maps SDK for Android, you can add maps based on Google Maps data to your application. The API automatically handles access to Google Maps servers, data downloading, map display, and response to map gestures. You can also use API calls to add markers, polygons, and overlays to a basic map, and to change the user’s view of a particular map area. These objects provide additional information for map locations and allow user interaction with the map. The API allows you to add these graphics to a map:

* Icons anchored to specific positions on the map (Markers).
* Sets of line segments (Polylines).
* Enclosed segments (Polygons).
* Bitmap graphics anchored to specific positions on the map (Ground Overlays).
* Sets of images which are displayed on top of the base map tiles (Tile Overlays).

1. **MySQL:**

A database is a separate application that stores a collection of data. Each database has one or more distinct APIs for creating, accessing, managing, searching, and replicating the data it holds.

Other kinds of data stores can also be used, such as files on the file system or large hash tables in memory but data fetching, and writing would not be so fast and easy with those type of systems.

Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as **Foreign Keys**.

A **Relational Database Management System (RDBMS)** is a software that −

* Enables you to implement a database with tables, columns, and indexes.
* Guarantees the Referential Integrity between rows of various tables.
* Updates the indexes automatically.
* Interprets an SQL query and combines information from various tables.

RDBMS Terminology

Before we proceed to explain the MySQL database system, let us revise a few definitions related to the database.

* **Database** − A database is a collection of tables, with related data.
* **Table** − A table is a matrix with data. A table in a database looks like a simple spreadsheet.
* **Column** − One column (data element) contains data of one and the same kind, for example the column postcode.
* **Row** − A row (= tuple, entry or record) is a group of related data, for example the data of one subscription.
* **Redundancy** − Storing data twice, redundantly to make the system faster.
* **Primary Key** − A primary key is unique. A key value can not occur twice in one table. With a key, you can only find one row.
* **Foreign Key** − A foreign key is the linking pin between two tables.
* **Compound Key** − A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
* **Index** − An index in a database resembles an index at the back of a book.
* **Referential Integrity** − Referential Integrity makes sure that a foreign key value always points to an existing row.

MySQL Database

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons −

* MySQL is released under an open-source license. So you have nothing to pay to use it.
* MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
* MySQL uses a standard form of the well-known SQL data language.
* MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
* MySQL works very quickly and works well even with large data sets.
* MySQL is very friendly to PHP, the most appreciated language for web development.
* MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
* MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

**Chapter 3**

### Requirements and Analysis

**3.1 Problem Definition:**

One part of the public transport system is the Bus, probably one cheapest mode of transport it helps people travel to places in a city where trains are not available or required, but it is also used as a last mile travel solution by people who use the train, But some problems with the bus is they are delayed most the time and it hinders with everyone’s schedule, even if they arrive at the stop they are at full capacity, this is worse when it’s the peak hours of travelling.

**3.2 Requirements Specification:**

The system has the following system requirements:

1. System needs to get current location of the user
2. System should be able to display the current location for a given bus.
3. System should be able to compute the current location of the bus by averaging the current location of user who declare they are on the given bus.
4. Users should be able to buy ticket and passes for a bus using the application.
5. System should be able to verify the tickets and passes.

**3.3 Planning and Scheduling**

**Gantt chart:**

A Gantt chart is a type of bar chart that illustrates a project schedule.

This chart lists the tasks to be performed on the vertical axis,

and time intervals on the horizontal axis. The width of the horizontal bars in

the graph shows the duration of each activity. Gantt charts illustrate the start

and finish dates of the terminal elements and summary elements of a project.

Terminal elements and summary elements constitute the work breakdown

structure of the project. Modern Gantt charts also show the dependency (i.e.,

precedence network) relationships between activities. Gantt charts can be

used to show current schedule status using percent-complete shadings and a

vertical "TODAY" line as shown here. Gantt charts are sometimes equated with

bar charts. Gantt charts are usually created initially using an early start time

approach, where each task is scheduled to start immediately when its

prerequisites are complete. This method maximizes the float time available for

all tasks.

On a Gantt chart you can easily see:

* The start date of the project
* What the project tasks are
* Who is working on each task
* When tasks start and finish
* How long each task will take
* How the tasks group together, overlap and link with each other
* The finish date of the project.

A screenshot of a computer

Description automatically generated

**3.4 Software and Hardware Requirements:**

* **Software requirements:**
  + **Languages:** XML, Kotlin
  + **Database:** MySQL
  + **APIs:** Google Maps SDK API
* **Hardware Requirements:**
* Android Phone Compatible with Android 9+
* GPS Sensor
* 4GB RAM
* Internet Connection

**3.5 Preliminary Product Description:**

Preliminary Product Description is to identify he requirements and objectives of the new system and to define the functions and operation of the application.

The Main Objective of this app is to help user track current location of a bus and decide

between using the bus or an alternative mode of transport and users can use the app to purchase tickets and passes.

**3.6 Conceptual Models**

1. **ER Diagram:**

An entity–relationship model (or ER model) describes interrelated things of

interest in a specific domain of knowledge. A basic ER model is composed of

entity types (which classify the things of interest) and specifies relationships

that can exist between entities (instances of those entity types). A relationship

is how the data is shared between entities. There are three types of

relationships between entities.

By defining the entities, their attributes, and showing the relationships

between them, an ER diagram illustrates the logical structure of databases.

ER diagrams are used to sketch out the design of a database.

Limitations: -

* Limited relationship representation: ER model represents limited

relationship as compared to another data models like relational model etc.

* No representation of data manipulation: It is difficult to show data

manipulation in ER mode

A screenshot of a computer

Description automatically generated with medium confidence

1. **Deployment Diagram:**

Deployment diagram is a structure diagram which shows architecture of the

system as deployment (distribution) of software artifacts to deployment

targets.

Artifacts represent concrete elements in the physical world that are the result

of a development process. Examples of artifacts are executable files, libraries,

archives, database schemas, configuration files, etc.

Deployment target is usually represented by a node which is either hardware

device or some software execution environment. Nodes could be connected

through communication paths to create networked systems of arbitrary

complexity.

Note, that components were directly deployed to nodes in UML 1.x

deployment diagrams. In UML 2.x artifacts are deployed to nodes, and artifacts

could manifest (implement) components. Components are deployed to nodes

indirectly through artifacts.

Deployment diagrams could describe architecture at specification level (also

called type level) or at instance level (like class diagrams and object

diagrams).

Specification level deployment diagram shows some overview

of deployment of artifacts to deployment targets, without referencing specific

instances of artifacts or nodes.

Instance level deployment diagram shows deployment of instances

of artifacts to specific instances of deployment targets. It could be used for

example to show differences in deployments to development, staging or

production environments with the names/ids of specific build or deployment

servers or devices.

Purpose:

The term Deployment itself describes the purpose of the diagram. Deployment

diagrams are used for describing the hardware components, where software

components are deployed. Component diagrams and deployment diagrams are

closely related. Component diagrams are used to describe the components and deployment

diagrams show how they are deployed in hardware.

UML is mainly designed to focus on the software artifacts of a system. However,

these two diagrams are special diagrams used to focus on software and hardware

components. Most of the UML diagrams are used to handle logical components but deployment

diagrams are made to focus on the hardware topology of a system. Deployment

diagrams are used by the system engineers.

The purpose of deployment diagrams can be described as −

• Visualize the hardware topology of a system.

• Describe the hardware components used to deploy software components.

• Describe the runtime processing nodes

A screenshot of a computer

Description automatically generated

1. **Activity Diagram**

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc

Purpose of Activity Diagrams: The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another

Graphical user interface

Description automatically generated

1. **Use Case Diagram**

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

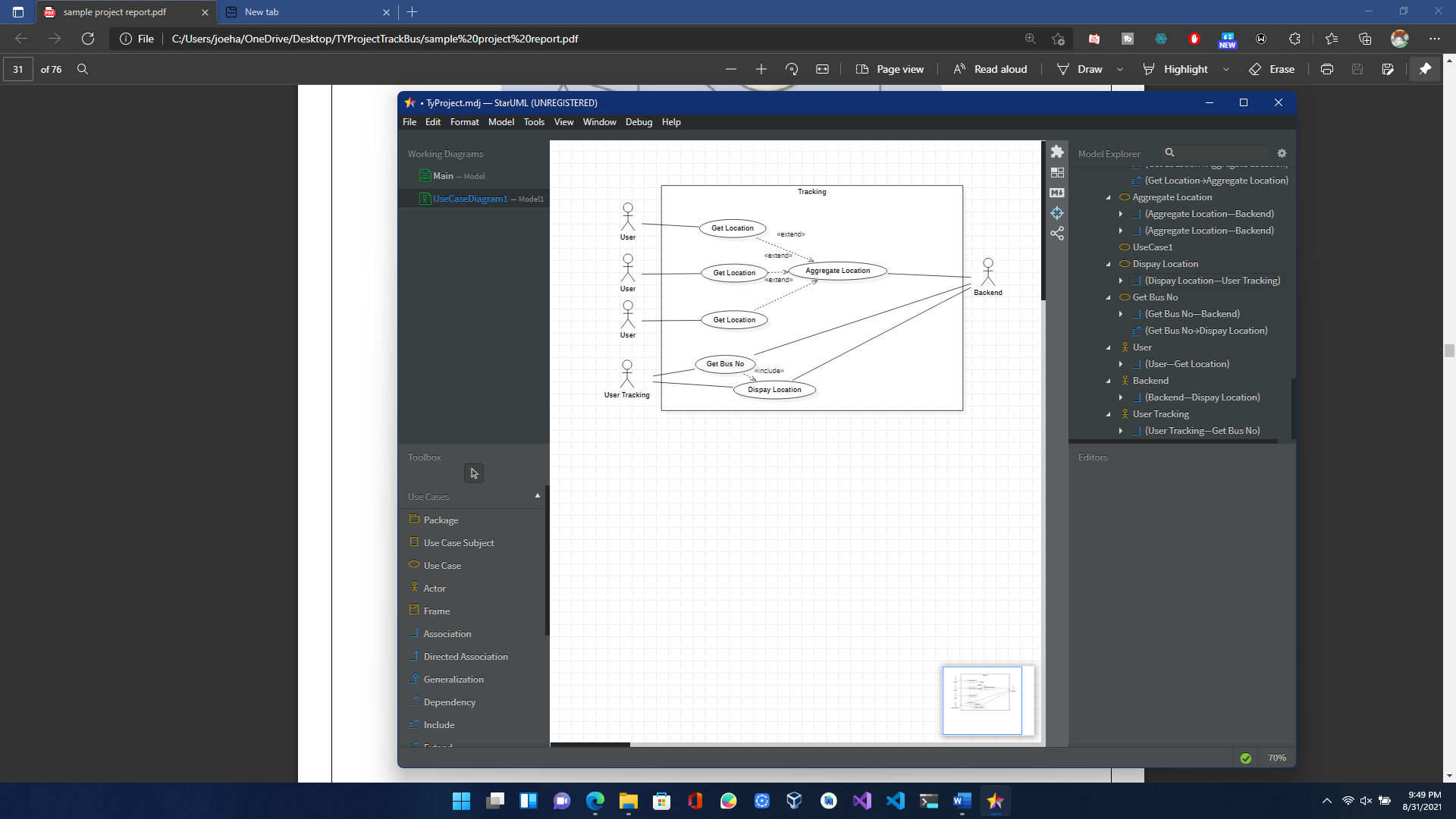
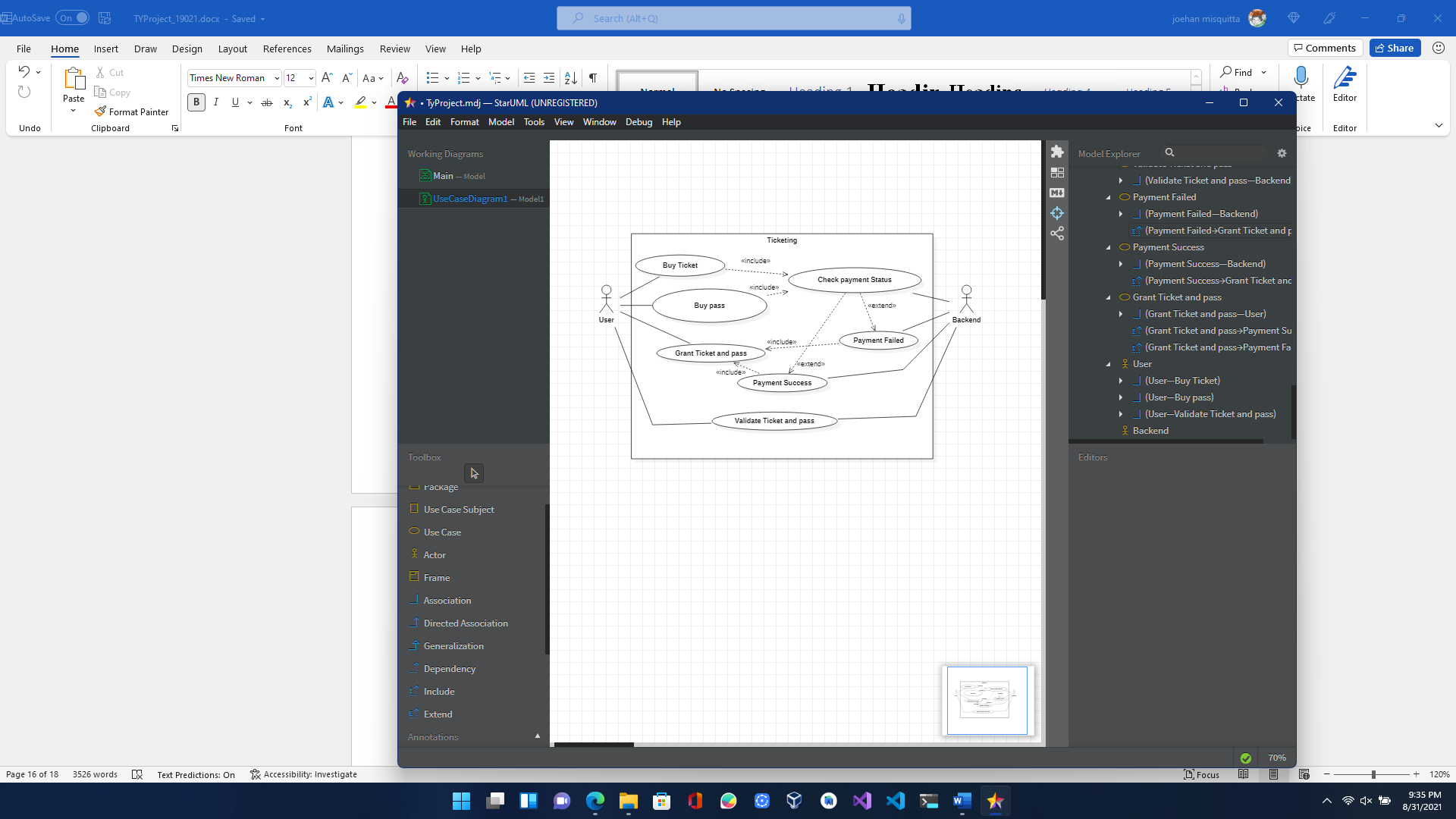
A use case diagram is usually simple. It does not show the detail of the use cases:

* It only summarizes some of the relationships between use cases, actors, and systems.
* It does not show the order in which steps are performed to achieve the goals of each use case.

**Graphical user interface

Description automatically generatedA picture containing text, screenshot, monitor, computer

Description automatically generated**

****

1. **Class Diagram:**

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

Purpose of Class Diagrams:

1. Shows static structure of classifiers in a system.

2. Diagram provides a basic notation for other structure diagrams prescribed by UML.

3. Helpful for developers and other team members too.

4. Business Analysts can use class diagrams to model systems from a business perspective

A UML class diagram is made up of:

* A set of classes and A set of relationships between classes

Diagram

Description automatically generated

**Chapter 4**

### System Design

* 1. **Basic Modules**

Modules:

The project is divided into more manageable parts to develop each part or module separately. When all modules are ready, we integrate all the modules into one system.

These are the Modules of the Project:

* **Plan:**

1. By Destination
2. From location
3. To Location
4. By Route
5. Route No.
6. Map

* **Explore:**

1. Enter locality
2. Map

* **Ticketing:**

1. Buy Tickets:
2. Select route
3. Payment
4. Buy Bus Pass:
5. Select route
6. Select Pass duration
7. Payment
8. My Tickets/Pass

* **Menu:**

1. Account:
2. Login/logout using Google Account
3. Account Info
4. Settings:
5. Location access
6. Feedback
   1. **Data Design**
      1. **Schema Design**

The schema design deals with the type and the name of the variable present in the module

**User:**

UserID: Int

Email: Varchar

Password: Varchar

**Tickets:**

UserID: Int

Ticketno: Int

From: String

To: String

Dateofpurchase: DateTime

**Passes:**

UserID: Int

PassId: Int

From: String

To: String

Dateofpurchase: DateTime

Dateofexpiry: Datetime

**Tracking:**

UserID: Int

Busno: Int

Longitude: Int

Latitude: Int

**BusCurrentLocation:**

Busno: Int

Longitude: Int

Latitude: Int

**4.2.2 Data Integrity and Constraints**

The Login page will have only 2 parameters

Email

Password

* 1. **Procedural Design**

Software Procedural Design (SPD) converts and translates structural elements into procedural explanations. SPD starts straight after data design and architectural design. This has now been mostly abandoned mostly due to the rise in preference of Object-Oriented Programming and design patterns The procedural design is often understood as a software design process that uses control commands such as: sequence, condition, repetition, which are applied to the predefined data. Sequences serve to achieve the processing steps in order that is essential in the specification of any algorithm. Conditions provide facilities for achieving selected processing according to some logical statement. Repetitions serve to achieve looping during the computation process. These three commands are implemented as ready programming language constructs. The programming languages that provide such command constructs are called imperative programming languages. The software design technique that relies on these constructs is called procedural design, or also structured design

* + 1. **Logic Diagrams**

Logic diagrams are diagrams in the field of logic, used for representation and to carry out certain types of reasoning.

**Control Flow Diagram**

**A screenshot of a computer

Description automatically generated**

* 1. **User Interface Design**

This is one of the main task of the developer to design a graphical user interface that user attracts to and can use easily i.e. in one word it should be user-friendly. There should be a better understanding of customer’s likes and dislikes and the features that are in trend and mesmerize the public easily. It is concerned with everything from starting the system or logging into the system to the eventually presentation of desired inputs and outputs.

Functions performed by the users

**Plan**:

User enters his current location and his destination and clicks on the plan button

The system gets the location of the closest bus/vehicle and displays it to the user

**Explore:**

User enters his current location and clicks on the explore button, The system displays the closest bus stops to the user.

**Ticketing:**

User can buy tickets and passes using this application

* 1. **Security Issues**

This Application Contains Location information of the user and Buses, which can be used by malicious actors for the purpose of harming the users or the vehicle.

* 1. **Test Case Design**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test**  **Scenario** | **Test Steps** | **Test Data** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| Check  User login with valid Data | 1.Enter Email  2.Enter  Password  3.Click  Submit | Email  =joehanm10@gmail.com  Password  = joehan | User should Login into an application |  |  |
| Check  User login with invalid data | 1.Enter Email  2.Enter  Password  3.Click  Submit | Email =joehan43@gmail.com  Password  = g33rcw | User should not Login into an application |  |  |

## References

1. <https://en.wikipedia.org/wiki/Android_(operating_system)>
2. <https://en.wikipedia.org/wiki/Kotlin_(programming_language)>
3. https://www.tutorialspoint.com/mysql/mysql-introduction.htm
4. <https://www.mapspeople.com/mapsapis/#:~:text=With%20the%20Maps%20SDK%20for%20Android%2C%20you%20can,downloading%2C%20map%20display%2C%20and%20response%20to%20map%20gestures>.