IMPERIAL

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

# I-MOTION: A Modern Travel Diary Collection Application

*Supervisor:*
Aruna Sivakumar, Jacek Pawlak, Pancham Shukla

*Author:*
Haiyi Wang

*Second Marker:*
Ivan Procaccini

Submitted in partial fulfillment of the requirements for the MSc degree in Software Engineering of Imperial College London

September 2024

**Abstract**

The study of current urban systems is making changes because of advanced mobile technologies. The utilization of mobile devices by researchers has helped them to investigate the human behavior in urban environments. This transformation needs the development of data collection and analysis techniques, particularly in the areas of digital activities, time allocation, and travel patterns. Traditional data collection methods are still valuable. However, they usually fail to capture the dynamic characteristics of human behavior in urban environments. Besides, The current implementations of mobile-based systems still have many possibilities to improve. The objective of this project is creating a open-source modular mobile platform-based system named IMOTION to overcome these constraints.This report discusses the system architecture of IMOTION, including UI and front-end, middleware (deep learning), back-end, and data model. From the researcher's perspective, IMOTION helps them detect and collect information such as: travel start/end, travel origin/destination, travel activities, and travel modes. From the participant's perspective, IMOTION lowers the barrier to use, provides timely feedback, and protects privacy. provide timely feedback, and protect privacy. This allows users to be more actively engaged with IMOTION. In addition, IMOTION's open-source nature allows researchers to modify the solution according to their own specifications and share them.

# Acknowledgments

I want to express my heartfelt gratitude to my supervisors Dr Pancham Shukla, Dr Ivan Procaccini and Dr Aruna Sivakumar for their guidance and support throughout the project. Furthermore, I would like to thank Dr Jacek Pawlak whose valuable inputs informed the project's design. Lastly, my gratitude goes out to all the participants who dedicated their time to test the application and offer invaluable feedback.

# Contents


**1 Introduction** **2**
1.1 Motivation . . . 2
1.2 Aims & Objectives . . . 3

**2 Background** **5**
2.1 Overview of Traditional Survey Methods . . . 5
2.2 Smartphone-Based Travel Survey System . . . 6
2.3 Data Methodology and Classification . . . 6
2.4 Introduction of Future Mobility Sensing . . . 7
2.5 Architecture of FMS . . . 7
2.6 Existing Implementations . . . 8
2.6.1 MoveSmarter . . . 8
2.6.2 MEILI . . . 8
2.7 Summary . . . 9

**3 Design** **10**
3.1 Statement of Problems . . . 10
3.2 Design Requirements . . . 11
3.3 UI Design . . . 11
3.3.1 High Fidelity Wireframe . . . 12
3.3.2 Privacy Level . . . 14
3.4 Architecture . . . 15
3.4.1 System Context Diagram . . . 15
3.4.2 Container Diagram . . . 18
3.4.3 Component Diagram . . . 18

**4 Implementation** **21**
4.1 Technology Stack . . . 21
4.1.1 Mobile Framework: Swift . . . 21
4.1.2 Backend Framework: Java Spring & Python FastAPI . . . 21
4.1.3 Database: PostgreSQL . . . 22
4.1.4 Machine Learning: PyTorch . . . 23
4.1.5 Distributed microservice Framework: Docker& Kubernetes . . . 23
4.1.6 Reflection . . . 24
4.2 Transport Mode Detection . . . 24
4.2.1 Machine Learning Model . . . 24

4.2.2 Merging Algorithm . . . . . 27
4.3 Database . . . . . 27
4.4 Service Logic . . . . . 29
4.4.1 Setup . . . . . 29
4.4.2 Main Page . . . . . 30
4.4.3 Background Processing . . . . . 35

**5 Evaluation** **36**
5.1 User Testing . . . . . 36
5.1.1 Setup . . . . . 37
5.1.2 Result of UI Testing . . . . . 37
5.1.3 Result of Usability Testing . . . . . 38
5.2 Machine Learning Performance . . . . . 38
5.2.1 Setup . . . . . 38
5.2.2 Training Performance . . . . . 39
5.2.3 Test Performance . . . . . 39
5.2.4 Reflection . . . . . 41

**6 Conclusion** **42**
6.1 Limitations . . . . . 43
6.1.1 Third-party Plugin . . . . . 43
6.1.2 Implementation of Statistics and Community Page . . . . . 43
6.1.3 Watch Connectivity . . . . . 43
6.1.4 Languages . . . . . 44
6.2 Ethical and Professional Considerations . . . . . 44
6.3 Future Works . . . . . 45
6.3.1 Administration Web for researchers . . . . . 45
6.3.2 Progressive Learning Technique for Deep Learning Multi-Classification Network . . . . . 45
6.3.3 Message Queuing for Large Data Processing . . . . . 45
6.3.4 Big Data System . . . . . 46

**A User Testing Survey** **51**
A.1 UI Testing Open-Ended Survey . . . . . 51
A.1.1 When you first got into the start page, how would you describe your experience while trying to create a new account and log in? Were there any steps that felt unclear or confusing? . . . 51
A.1.2 When you started the primary survey, how intuitive did you find the process of answering questions? Were there any parts of the survey interface that you felt could be improved for better usability? . . . . . 51
A.1.3 In the main page, how intuitive did you find every features by icons or colors without telling you? Did you think statistics, community, and survey page help you to use this application more frequently? . . . . . 52

A.1.4 In the track page, do you think the layouts of design which is too complex or not? Is it easy to help you get your personal travel information in trip list view? Could you learn quickly how to use this page based on navigation? . . . . . 52
A.1.5 In the trip detail page, how smooth is your experience when entering details about your transportation and activities? Does the interface allow you to easily input and review your trip data? 53
A.1.6 What other feature do you think the app should have? . . . . 53

**B System Usability Scale** **54**

# Chapter 1

# Introduction

## 1.1 Motivation

The motivation of this project comes from the changing research environment and the need to address emerging challenges and opportunities for urban data collection and analysis. Although valuable, traditional methods of data collection for urban system research often fail to capture human behavior[1]. However, the capabilities of mobile devices open a new way for data collection and analysis. It allows researchers to delve further into the complexities of human behavior in many ways[2],such as transport planning, mobility services, social behavior research and land usage plan, etc. The utilization of mobile platforms is a solution that provides real-time insights into how to track individuals trips in different contexts.

Although in its early stage, mobile device-based travel diary collection already has many practical applications and technical implementations among major institutions and prestigious universities. Such systems can be broadly categorised into three types: memory-based systems, automated systems and semi-automated systems[3]. Of these, automated and semi-automated systems are the most popular, as memory-based systems are not very different in nature from traditional travel surveys. Automated systems are characterised by the fact that participants do not need to do anything with the application, but simply carry a smart device, and the system automatically records some of the travel diary information, such as destination, purpose of the trip, or mode of travel[3]. However, this approach leads to narrower data collection and less accurate data[3]. In contrast, semi-automated systems are more flexible, allowing for continuous improvement and refinement of the system by either automatically recording some of the travelling information via mobile devices, as in automated systems, or by providing interfaces that allow participants to modify or add information based on their memories that cannot be automatically accessed via their devices[3].

However, the current literature does not distinguish between automated and semi-automated travel diary collection systems, which makes it difficult to identify the strengths and weaknesses of each system and to provide clear guidance on best

practices and future research directions for travel diary collection systems[3]. In addition, the literature also lacks detailed discussions on the software design and code level, which leads to a relative closure of information and a scarcity of open source projects, and a consequent duplication of many studies. There is also very little discussion on how to improve user engagement through the design of user interfaces. In fact, user engagement decreases over the course of using a mobile system [4] because current research does not devote enough attention to system and user interface design. Besides, many systems still use traditional machine learning frameworks, which leads to higher costs and lower efficiency for maintenance and upgrades because traditional machine learning architectures lack the ability to self-learn.

## 1.2 Aims & Objectives

The project tries to address existing gaps in mobile data collection and analysis platforms and fulfill needs of Imperial's Urban Systems Lab. By developing a new mobile platform-based time use and travel data collection platform, this project helps to improve the breadth and depth for interdisciplinary research efforts. The core driver of the project is to fill gaps in existing research in this area and provide new directions for future research, helping researchers to work more effectively in this area. From the level of software design and software architecture, it provides new directions and new possibilities for the research in this field to help the research of this project, so as to further develop from different angles and help urban planning in a larger direction more smoothly.

The specific objectives of the project is to create a open-resource semi-automated mobile travel diary software system – IMOTION, which has:

- flexible microservice architecture
- user-centred UI design for increasing participation and stickiness
- effective and accurate deep learning model to support inference of mode of transport with less sensors of mobile devices
- conceptual logic of sharing users' travel journals
- novel means of preserving desired user privacy
- A small prototype for evaluations

This project tries to use the power of mobile technology to change the way understanding urban systems. By pushing the limits of data collection capabilities and fostering interdisciplinary collaboration, it will drive positive change in the urban system research.

The remainder of this paper is structured as follows: Section 2 reviews and discusses the history and current development and methodologies of travel diary collection

systems for finding possibilities, Section 3 demonstrates and explains UI and software design, Section 4 presents implementations of prototypes, Section 5 evaluate the prototype, and Section 6 conclude the report.

# Chapter 2

# Background

In order to build a robust travel diary mobile application with objectives in Section 1.2, this section will explore the following questions:

1. What is the traditional survey methods to collect participants' travel trails?

2. Is there any new survey system to take up the traditional one? What has been done?

3. What kinds of data can help building the robust and variable traffic models?

4. What existing solutions has been developed? What are their advantages and limitations?

## 2.1 Overview of Traditional Survey Methods

Traditional traffic survey methods, including paper-based diaries and interviews, computer-assisted telephone interviews (CATI), and GPS-based methods have long been the primary means to collect and analyze traffic travel behavior[2][5][6]. However, there are limitations in traditional methods.

They rely on respondents' memory heavily[6]. The data quality is unstable due to human memory limitations and subjectivity. So, It is challenging to support accurate modeling and decision-making processes[5]. Moreover, these approaches result in low participation rates and data bias by take up significant time and effort burdens on respondents[5]. Although GPS loggers can automatically record the location and movements of participants, it is necessary for participants to carry an additional device and ensure its continuous operation throughout the survey period in order to avoid missing any travel information[5]. Besides, relying solely on GPS data does not get detailed information such as trip purpose and traffic patterns. Such kind of information require additional input from participants or inference by algorithms, thus increasing the complexity of data processing [6].

## 2.2 Smartphone-Based Travel Survey System

The Smartphone-Based Travel Survey System(SBTSS) uses the mobile phone as its platform. It applies the GPS locator, a range of sensors , and machine learning techniques to predict subject trajectories. This kind of the survey system offers several advantages over traditional methods. SBTSS is cost effective because smartphones reduce the costs associated with traditional survey methods, such as printing, mailing, and manual data entry[6]. SBTSS provides more accurate and details of data compared to recall-based survey. SBTSS uses GPS and mobile sensing to automatically record trips and details without manually taking participant's memory to record travel details[5][6]. SBTSS captures a richer set of data, including socio-economic activities, multiple trips, and non-work-related trips[6][5]. SBTSS is adaptable to various contexts and can be continuously improved with additional sensors and machine learning algorithms. This adaptability is crucial for capturing the rapidly changing travel behaviors, especially in response to events like the COVID-19 pandemic[6]. With the verification of participants, SBTSS ensures more complete data collection[6][5]. Besides, the user-friendly interface and automated reminders in SBTSS help maintain higher participant engagement and response rates compared to traditional surveys.[6].

However, there are still some limitations which IMOTION can avoid and improve. A standardized test procedures for evaluating software designs and algorithms lacks in this domain[7]. Moreover, the performance of the software is influenced by the mobile system (iOS and Android), with greater stability observed on iOS devices[7]. Furthermore, variations in prediction accuracy exist due to differences in machine learning algorithms by different schools or organizations, ranging from 60% to 90% according to experimental findings[7]. So IMOTION should start in iOS devices and uses a stable and suitable machine learning algorithm.

## 2.3 Data Methodology and Classification

In the field of traffic modeling, the effective utilization of multiple data sources has significantly enhanced the accuracy and practicality of the model. Mahajan et al. present a comprehensive review of public and private data in transportation models, and highlight the impact of non-traditional data sources(mobile phones, social media, and smart cards) on traffic analysis and prediction[8]. By categorizing key data into four groups (commercial data, unavailable data, limited free data, and open data ), data accessibility is leveled and these classifications help people to figure clearly out the influence have on transportation models[8]. Current trends indicate an increasing reliance on hybrid approaches in open and proprietary datasets can lead to more robust and versatile traffic models. This trend represents an advancement from traditional methods primarily reliant on survey and census information while marking a evolution in transportation planning and policy-making.

Furthermore, Mahajan et al. discusses the availability of transportation data to the

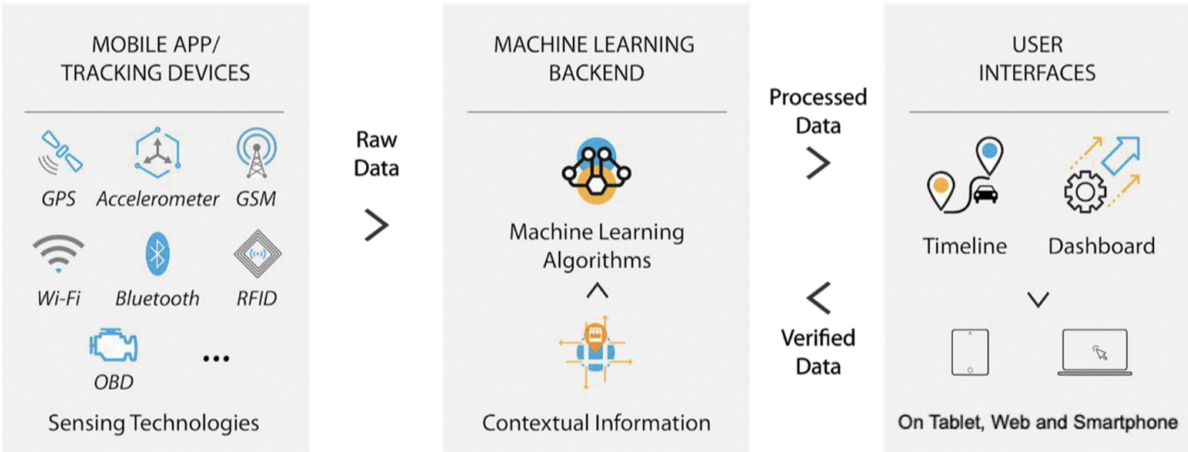


**Figure 2.1:** Future Mobility Sensing (FMS) platform architecture[5]

public. They critically analyze the central role played by open data in facilitating portability and applicability of research findings. SWOT( Strengths, Weaknesses, Opportunities and Threats)[8] analysis effectively reveals strengths, weaknesses opportunities, and challenges associated with different types of datasets. SWOT provides the insightful perspectives into their usage in traffic modeling. The research suggests that both public and private dataset owners can enhance sharing data while safeguarding privacy rights and commercial interests[8]. More accessible use of open data shall help propel transportation research towards a more inclusive and informed direction, laid upon the foundation for democratization of data access.

## 2.4 Introduction of Future Mobility Sensing

Future Mobility Sensing (FMS) is one of the popular architecture inspired by the Smartphone-Based Travel Survey System idea. FMS is a platform that collects and visualizes data by using mobile sensing technology, machine learning algorithms, and user verification to accurately record and present detailed travel and activity data[2][5]. The development of FMS technology marks a major shift in the way traffic data is collected, particularly the convergence of smartphone technology with location-based data collection. FMS utilizes GPS, GSM, Wi-Fi and accelerometers to collect high-resolution objective data on travel patterns and behavior in real time[2][5]. This technique not only improves the accuracy of the data, but also reduces the recall bias associated with traditional surveys.

## 2.5 Architecture of FMS

The FMS system (Figure 2.1) consists of three interrelated parts. It includes a mobile app or tracking device equipped with various sensing technologies to collect data[5]. The backend of the system is a server with a database and custom algorithms to infer the details of the trip, such as the trip site, mode and purpose,

thereby reducing the burden on the user[5]. In addition, a user interface accessible through mobile phones and web platforms allows participants to verify their activities. With validation, more information can be gathered and comprehensive data can be displayed[5]. Users can visually see their daily trips and activities on a geo-plotted timeline and can also confirm trips automatically detected by the system. For IMOTION and other implemented travel diary system, FMS is the important mature theoretical software design to start with.

## 2.6 Existing Implementations

Over the years, many applications have been developed to fulfil the needs of smartphone-based Travel Survey. This section will delve into a few noteworthy examples that provide valuable insights into the successes and limitations associated with existing solutions.

### 2.6.1 MoveSmarter

The MoveSmarter smartphone application, developed by the Dutch Mobile Traffic Panel (DMTP)[9], monitors travel patterns through automatic detection and data collection features. The system uses smartphone sensors to automatically detect where a journey begins and ends, as well as the type of transport used and the purpose of the journey. This data is then refined through a filtering and map matching process[9]. The system also includes a recall survey that provides recommendations. Participants had the ability to evaluate and modify their own trip data to improve the accuracy of the information collected. This automated approach simplifies the data collection process and minimises respondent workload. In addition, it increased the level of detail and accuracy of travel data over an extended period of time through constant monitoring[9]. Furthermore, it effectively captures dynamic fluctuations in travel behaviour that are difficult to capture in traditional one-off surveys.

However, there are many limitations to this particular application. Existing trip identification methods have a trip detection rate of only 75 - 80%[9] when the activity time is short and public transport is used. In addition, overuse of batteries may hinder user engagement and affect the accuracy of collected data[9]. In addition, the machine learning model is inaccurate in detecting similar transport modes, such as distinguishing between cycling and walking, and unreliable in identifying different types of public transport[9].

### 2.6.2 MEILI

The MEILI system is an advanced platform for the automatic collection and annotation of travel diaries. It is designed to capture and analyze travel behavior using GPS data from multiple agents. MEILI streamlines the process of collecting travel diaries and define trips and segments within a trip, known as ”triplegs” to put every

trips in a fine grain. MEILI is its open-source and modular architecture. It can be customized and adjusted by researchers based on specific needs. MEILI can minimize manual input from participants and reduce user burden through automated data . Moreover, it supports various active re-learning machine learning techniques to improve the overall efficiency of travel behavior research.

However, there are several limitations with the MEILI system. The continuous GPS tracking leads to significant battery consumption. Additionally, the accuracy of automatic trip detection algorithms needs improvement particularly for short trips and public transport modes. Furthermore, as the number of users increases, there is a high latency time or overhead for CRUD operations along with database capacity. Recognizing these shortcomings, the authors also acknowledge that such an application should not solely focus on recording travel but should collect additional data to enable more comprehensive analysis of human mobility.

## 2.7 Summary

In addition to the MoveSmarter and MEILI, many organisations have produced similar software such as TimeUse+ in ETH, MOTUS in VU Brussels, Future Mobility Sensing in MIT, and Motion-Tag in TUM. But most of the applications, because of their copyrights, it is not possible to understand their specific implementation theories and methods. So this is what this project needs to address. This project is completely open source. And although smartphone-based travel survey systems have made great strides in travel data diary, there are still challenges in their implementation. The digital literacy of some users, unstable system performance, high battery consumption, limited data usage, and restricted content exploration are significant barriers that need to be overcome. Future research should build a open-source software system in the foundation of FMS, and prioritise improving user-friendly interfaces for a wider audience, enhancing data privacy protection, improving system stability and scalability, minimising battery consumption while utilising diverse data for analysis, and exploring not only travel, but also other domains, such as user health and transport energy utilisation.

# Chapter 3

# Design

This chapter will explore the design considerations for developing I-MOTION. As highlighted in Chapter 2, the design must address several limitations of current technology and methodology, including high power consumption and limited information gathered. To guide the design process, the following key questions are considered:

1. **Statement of Problems**: What problem is the application trying to solve?

2. **Design Requirements**: What are expected functions of the application?

3. **UI Design**: What are the UI design of the application, and what principles should it follow?

4. **Architecture**: What are the system architecture of the application?

## 3.1 Statement of Problems

Most of the products on the market are primarily designed to serve research purposes, so their design originally focuses on researchers, often neglecting the user experience. This could potentially lead to poor user experience and low user engagement. Furthermore, for a platform that can collect sensitive user information, users cannot actively choose what information to share and what information not to share. This also results in the platform primarily using an invitation system to find users, making it difficult to get users to voluntarily download the app to expand the survey population. Additionally, because phone tracking and prediction require the use of multiple sensors and GPS in the phone, this leads to significant battery consumption, which greatly affects users' enthusiasm for using the app. Traditional machine learning has to use multidimensional detection to improve accuracy, further exacerbating this issue. Lastly, there is the limitation of information collection. Merely collecting personal activity information is insufficient; researchers need more diverse information to further depict a more specific and complete picture of urban activities. Finally, due to the complexity and volume of the collected information,

designing a backend structure that can run stably while reducing backend complexity to allow more people to easily participate in this open-source project is also a challenge.

## 3.2 Design Requirements

Based on problems and challenges above, the application should satisfy the following requirements:

- The UI design of frontend should interact and communicate with users nicely to increase users' stickiness.
- The application should focus on how to help users to choose how high privacy they want.
- The modern machine learning model should predict modes of transports by reducing senors while keeping accuracy.
- The user should easily edit activities to make them more accurate and real.
- The interactivity of user interfaces and backend logic to gather more information than only personal activities.
- The backend should be distributed and auto-scaled to be stable and low latency.

## 3.3 UI Design

Norman [10] states that the design of everything should pay attention to the affordance, visibility, mapping, and consistency. Besides, Rogers [11] emphasizes that an interactive design should be easy to understand, efficient in use, and offer a satisfying user experience. These principles emphasize that a good user interface design should have a low learning curve, with features like graphics and colors. A good design should be more intuitive to humans, taking precedence over more obscure elements like text. Additionally, the design should include the clear navigational logic to help users easily transition to the next interface. And appropriate feedback can enhance the overall user experience.

Tang[12] proposed that there are six main factors influencing users' stickiness: control, communication, responsiveness, context, mobile self-efficacy, and promotion. A design with high user engagement needs to exhibit dexterity, a good user experience, and consistency. It should effectively communicate with users through elements like charts and community feedback. Moreover, the design must display diverse and rich content. The UI should be both easy to use and useful. Moreover, the system's fast and stable responses and regular notifications can increase users' trust and reliance on the software.

A summary of the key guidelines for UI design is presented in Table 3.1:

| Component | Guidelines |
|---|---|
| Text | Limits the amount of texts on the single page. Care about the hierarchy between different pages |
| Hierarchy and layout | Use hybrid hierarchy including tab bar and navigation bar to map users between different parts. Users should clearly know which part they are in now |
| Graphics | Use simple graphic cues, like icons and symbols to help uses understand contexts. No abstract and complex design. For charts, the statistical information should be shown clearly |
| Design | Keep a simple and intuitive design. Do not put so many components in one page |
| Interfaces | Should let users actively use their fingers to make interfaces intuitively. |
| Error Feedback | Clear and useful error feedback to help users deal with any situations in the correct way |

**Table 3.1:** design guidelines

### 3.3.1 High Fidelity Wireframe

There are several HIFI wireframe of prototypes before implementations demonstrating how users interacts with this application. Figure 3.1 shows the wireframe of login and register pages in Figma.In the privacy page, app introduces how this privacy level works if users choose them. Figure 3.2 shows the wireframe of survey pages. Each buttons and graphs give users a linear flow and easily finish register, login, and survey. It takes short time for users to finish those set-up processes. Figure 3.3 shows examples of main pages. They are mapped by tab bars. Everything is displayed by rich of colors and simple icons, only a few words for understanding. Because every clickable things are designed in grinds, so users can click them easily. Figure 3.4 shows pages of track section. Users can see their trails in the map and activities in the trip lists below. Users can click calendar and setting icons to select which date they want to see and change settings. Trip lists can drag longer for more rich descriptions. Finally, users can click each trip rows into trip details to edit by themselves.

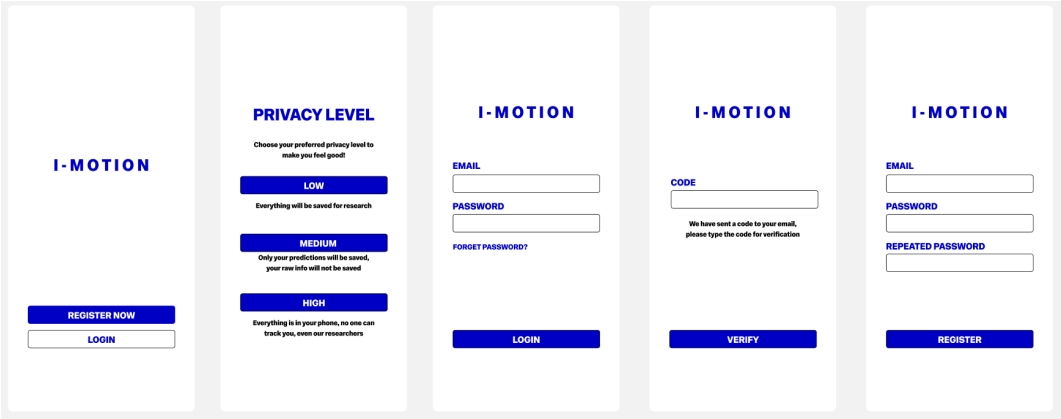


**Figure 3.1:** IMOTION Login and Register Pages

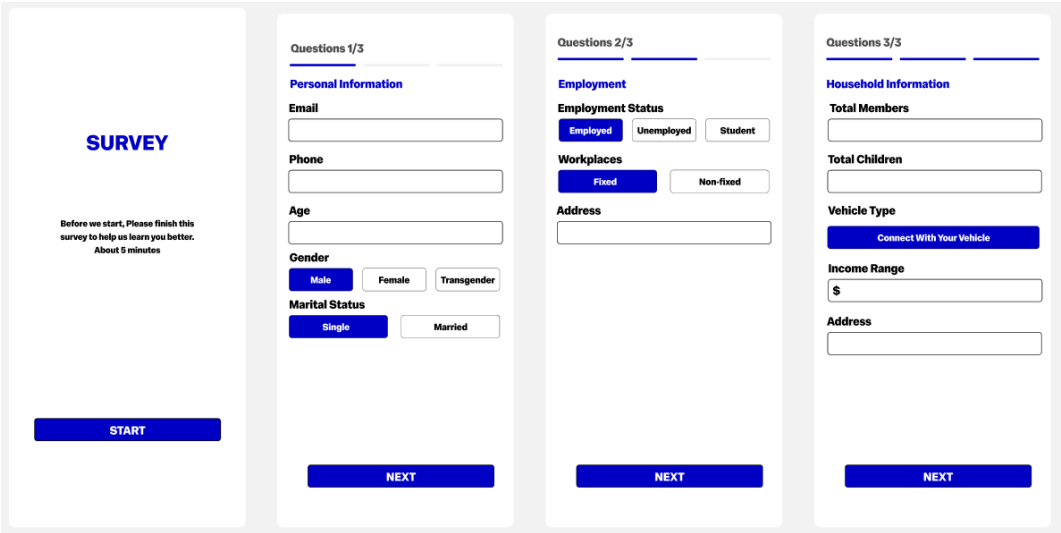


**Figure 3.2:** IMOTION Survey Pages

**Figure 3.3:** IMOTION Main Pages

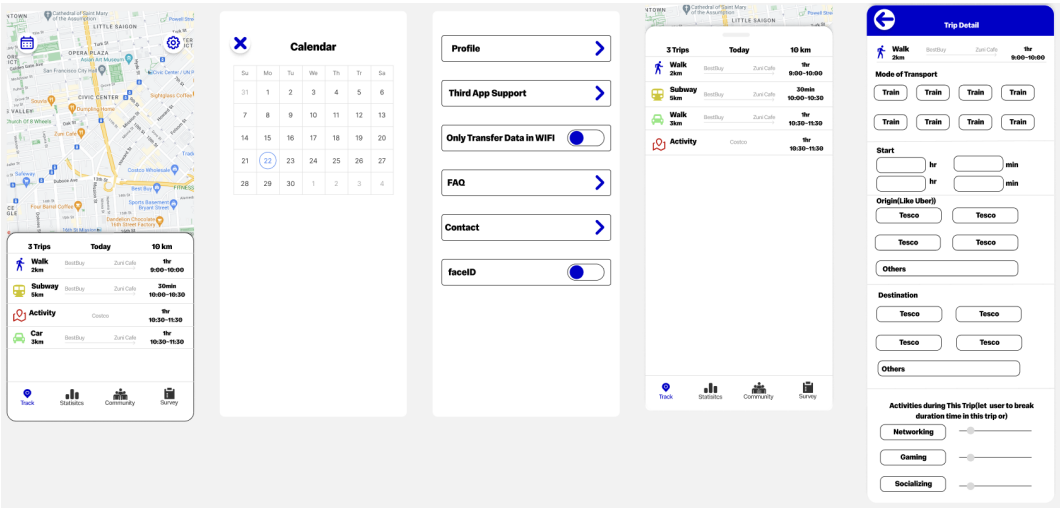


**Figure 3.4:** IMOTION Track Pages

### 3.3.2 Privacy Level

An important difference between IMOTION and other current implementations is the provision of privacy levels to help users have more flexibility in choosing at what level their travel data is made available to researchers. In general, some people refuse to use a data collection application because of security and privacy concerns. However, these concerns are more diverse. People's willing to allow researchers to track them based on many considerations: age, income level, gender, political affiliation, and education level[13]. For example, young and low-income, highly educated, male, and left-leaning groups are more likely to agree to and eventually participate in a study[13]. And if the data are highly personal (like audio recording), people are less willing to participate in this research[14]. Therefore, a privacy-level mechanism is necessary to get a more diverse group of people willing to use IMOTION. Under this mechanism, both the mobile device and the back-end servers should have the ability to infer raw data. the current design provides three levels: LOW, MEDIUM, and HIGH. in the case of LOW, the participant agrees to give all of raw data to the researcher, and the back-end is in charge of prediction and recording. In the MEDIUM case, participants do not agree to give sensitive data such as GPS data to the researcher, but agree to release other data for research purposes. In the HIGH case, all the data will not be given to the researcher and the mobile phone will be responsible for storing data and embedding the deep learning model for prediction.

## 3.4 Architecture

This section will demonstrates the conceptual architecture of the application in the three-layers diagrams: Context diagram, Container diagram, and Component diagram.

### 3.4.1 System Context Diagram

Figure 3.5 illustrates a comprehensive IMOTION system architecture. Five components consists of this architecture: 1.personal users with accounts, 2.a verification system, 3.a third-party API system, a 4.frontend mobile system, and 5.a backend server system. Users interact with the mobile frontend to view tracking data, access statistics, edit timelines, and manage privacy settings. The verification system authenticates users via email or phone. The frontend tracks trips and activities through sensors and APIs and then communicates with the backend server. Backend server stores data, makes inferences, and predicts user behaviors. The third-party API system provides additional tracking data.This design of the architecture makes sure a user-friendly experience, detailed activity statistics, and robust data privacy and security.

Figure 3.6 outlines the conceptual process for verifying and merging multi-person activities between users A and B. When user A adds user B to an activity, the backend updates the activity records to include B's participation. User B then refreshes their timeline, which triggers a backend check for any unmerged activities. If B user finds any unmerged activity within the relevant time frame, B will decide whether to merge it based on A's participation. After merge, the activity is updated to be synchronized between two users with the shared activities.

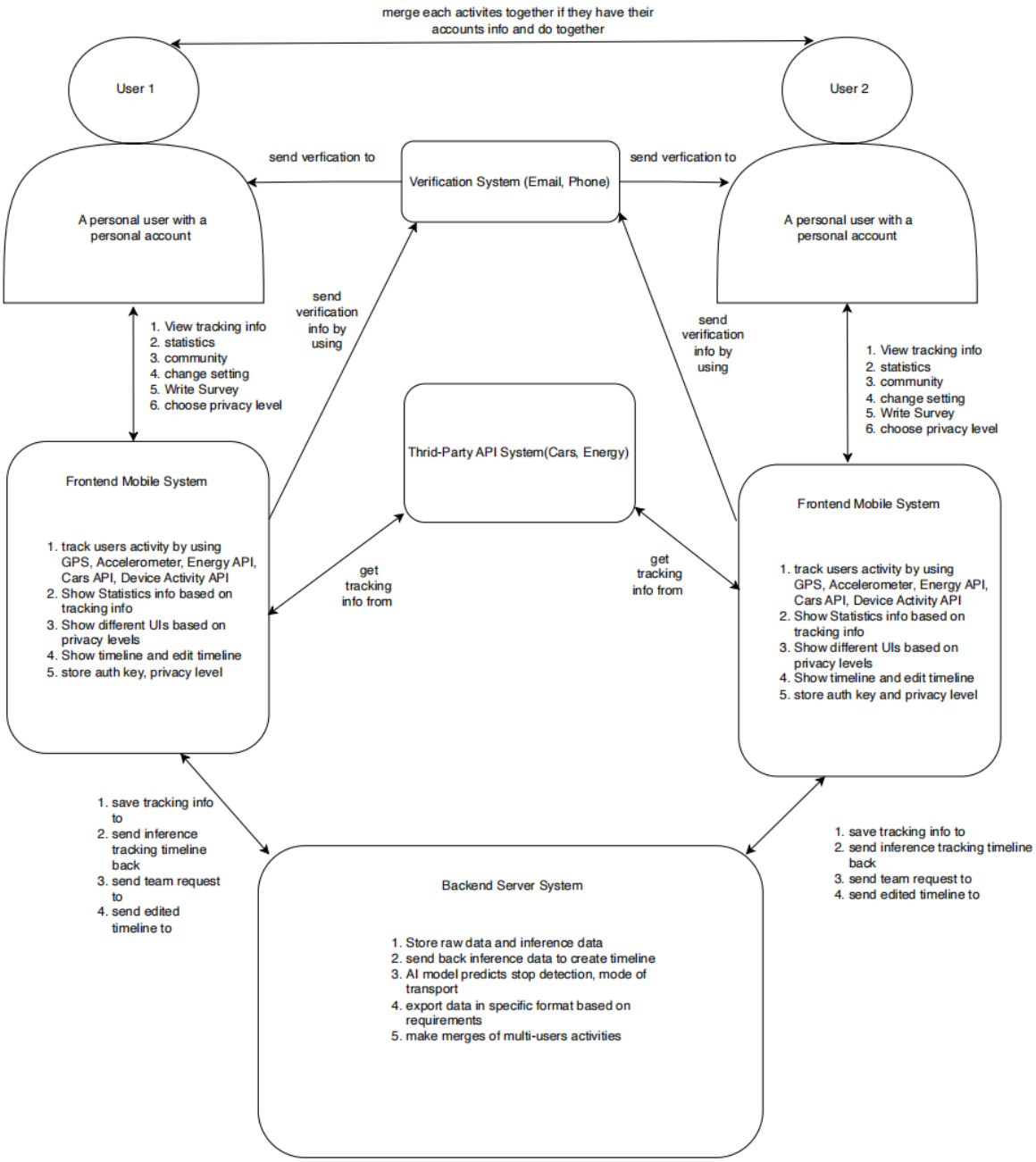


**Figure 3.5:** IMOTION System Context Diagram

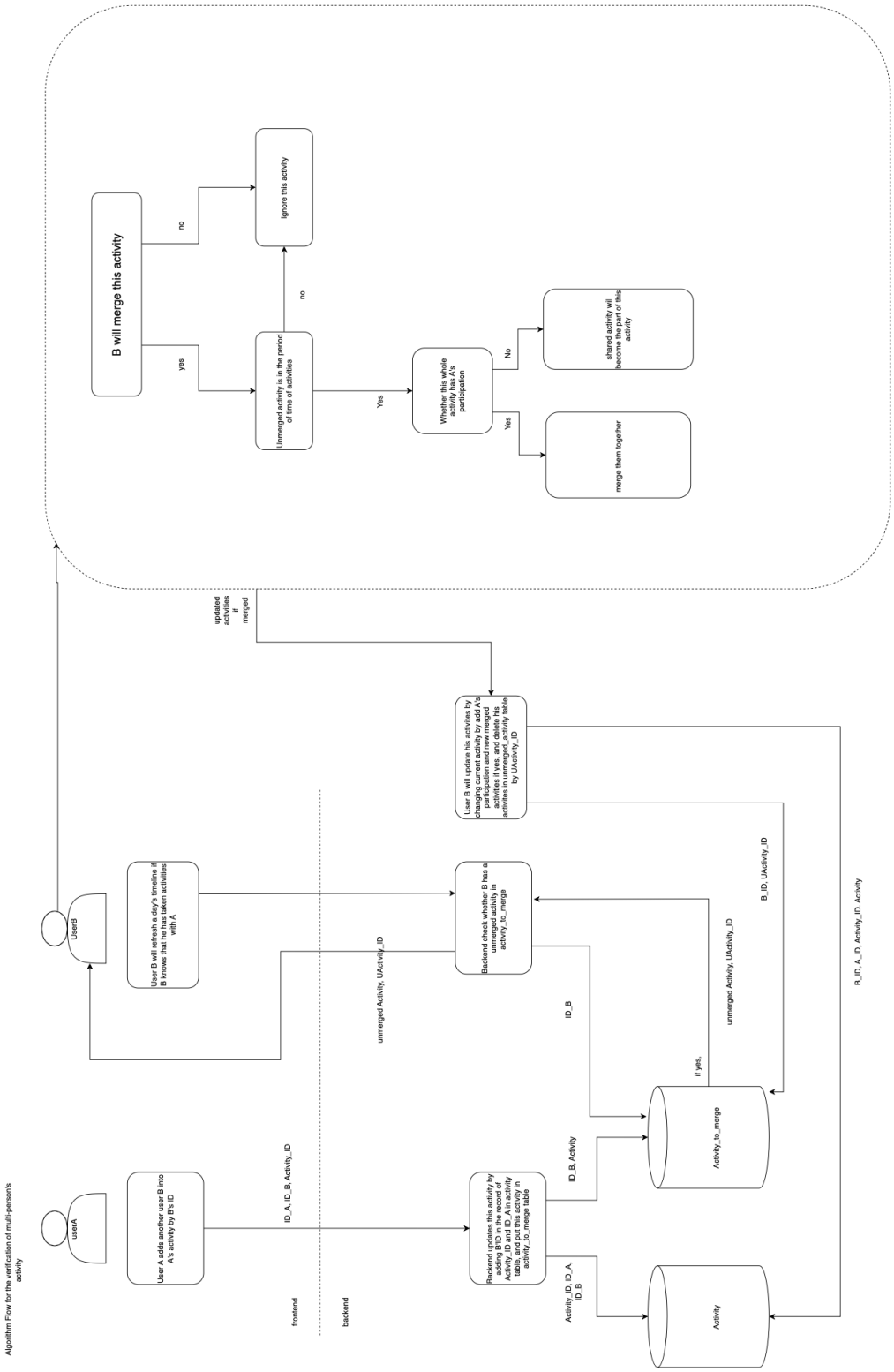


**Figure 3.6:** IMOTION System Context Diagram

### 3.4.2 Container Diagram

The container diagram in Figure 3.7 presents a comprehensive architecture for IMOTION system, involving multiple interconnected containers. Users interact with the system through the mobile application and access rich features. The mobile app collects data from various sensors (GPS, accelerometer, heart rate) and integrates with external sources like cars and energy systems. Data is processed and stored in a mobile database, then sent through an API gateway to the backend server, which includes AI models for data analysis. The backend server also interfaces with databases, community APIs, data collection APIs, and researcher APIs. Researchers access the system through a web client to manage privacy settings and export data. The system ensures robust data collection, processing, and user interaction through seamless integration of frontend, backend, and third-party services.

### 3.4.3 Component Diagram

The component diagram in Figure 3.8 depicts a multi-tier architecture of IMOTION system. It includes components like User API, Data Collection API, Statistics Survey API, and Survey API interacting with a PostgreSQL database and an AI Model via REST APIs. The system is designed for both mobile (Apple) and web clients. The Apple client architecture consists of mobile and watch apps. There are many SwiftUI frameworks for user interactions, scheduling, and handle data exchanges . In Kubernetes, The backend ensures secure communication and efficient data processing. Additionally, Redis is used for caching and storing live updates for mobile and web applications.

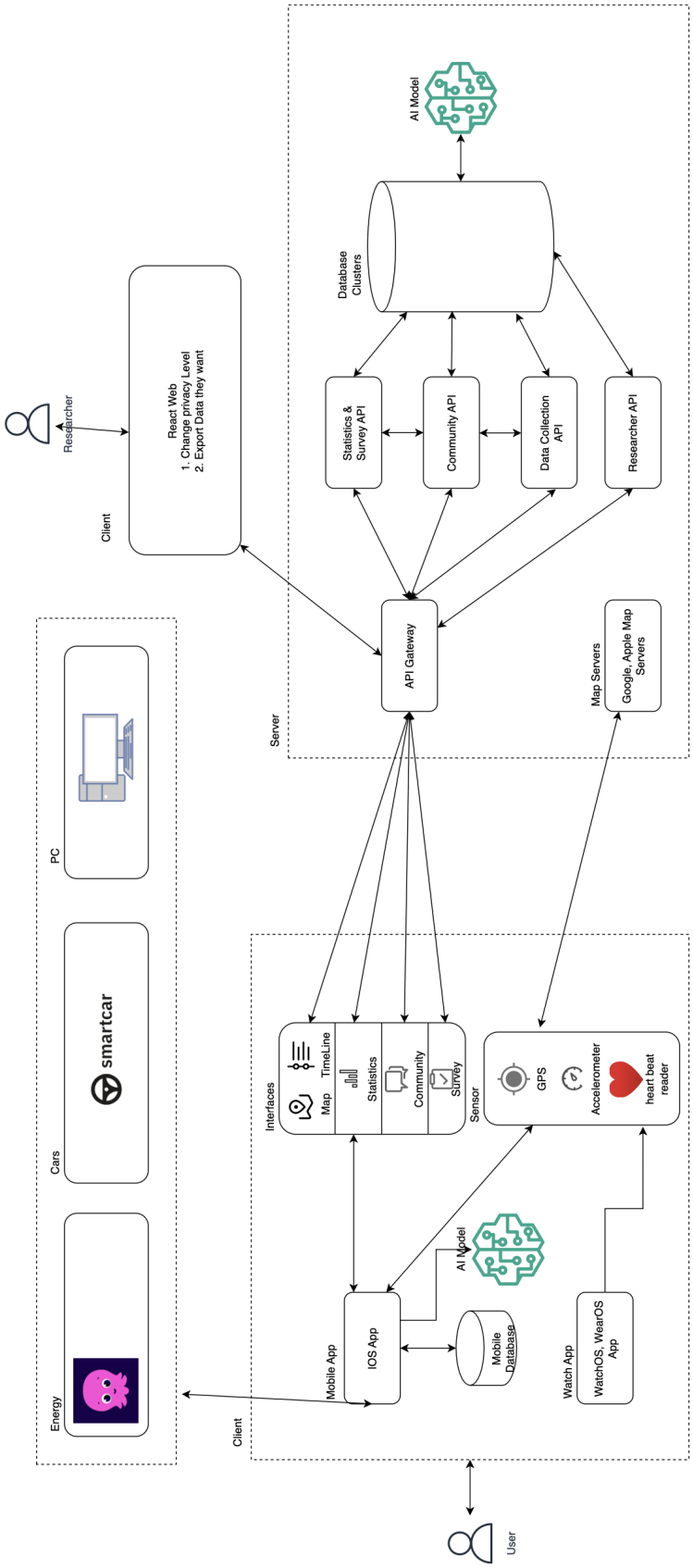


**Figure 3.7:** IMOTION Container Diagram

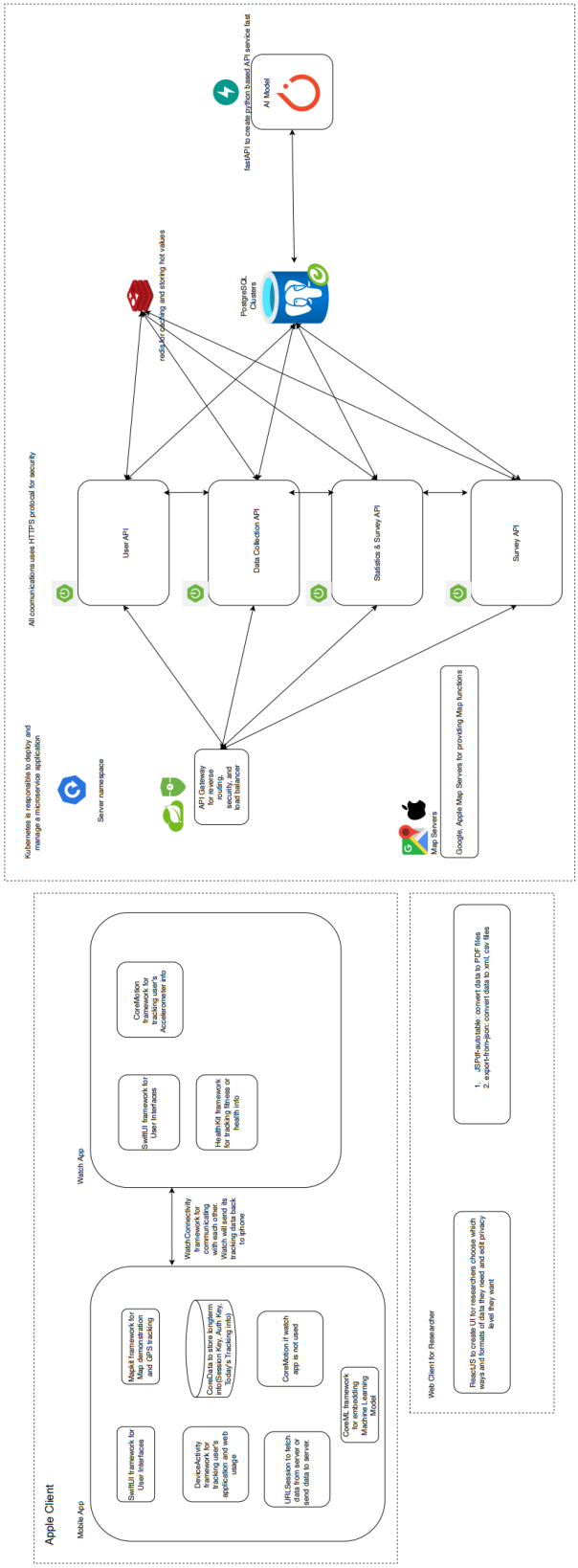


**Figure 3.8:** IMOTION Component Diagram

# Chapter 4

# Implementation

## 4.1 Technology Stack

For this project, I chose to build a microservice based full stack application, including frontend, backend, deep learning algorithms, data management, microservices architecture management. They consists of Swift for IOS frontend, Java Spring Boot/Cloud Framework and Python Fastapi for backend, PyTorch for machine learning, PostgreSQL for database, and Kubernetes and Docker for microservice management and deployment. This allows to split each parts of development in smaller grains. So each parts of backend and frontend will be tested and deployed separately. It will decrease the maintenance cost thanks to decoupling.

### 4.1.1 Mobile Framework: Swift

Swift is the programming language developed by Apple specifically for creating applications for Apple's operating systems[15]. Its security features, speed, and current syntax make it a strong and user-friendly tool for developers. Swift's efficiency optimizations and robust error handling in mobile development facilitate the creation of efficient and dependable applications.The citation for the source is "Swift". Swift programming in Xcode's integrated development environment allows for hot deployment and real-time updates[15]. Furthermore, Swift offers many packages to use Apple devices' functionalities seamlessly. These include SwiftUI for user interface design, UIDevice and NWPathMonitor for monitoring the status of iPhone, Mapkit for GPS functionality, CoreMotion for accelerometer usage, URLSession for retrieving data from or sending data to servers, Core ML for transferring PyTorch deep learning models to MLPackage for iOS app integration, and SwfitData and JWTChain for secure and persistent data storage. Those packages will also be used in this application of the project[15].

### 4.1.2 Backend Framework: Java Spring & Python FastAPI

The Java Spring Framework and Python FastAPI are two popular back-end frameworks today that are widely used to build high-performance and easily extensible

server-side applications.

Java Spring stands out for its versatility and high degree of integration, and it is suitable for the full range of development from simple stand-alone applications to complex enterprise applications[16]. Spring provides developers with a powerful and flexible programming and configuration environment that covers tasks as diverse as dependency injection, aspect oriented programming, and transaction management. Its modular design allows developers to accurately select the required components according to the needs of the project, thereby increasing the efficiency of application development[16]. In addition, Spring has a large developer community that provides rich framework support for a variety of features[16]. Spring's database connectivity capabilities, powerful security configuration capabilities, and extensive ecosystem of cloud-native microservices have made it a great convenience to develop microservice-based system.

Python FastAPI is a modern API framework designed for high-performance application development. It makes clever use of Python's async/await asynchronous features, type annotations, and automated data model validation, which not only greatly reduces the amount of code, but also significantly improves runtime efficiency[17]. Because of its support for asynchronous programming, FastAPI shows excellent performance when dealing with large data volumes or high concurrency scenarios, making it particularly suitable for IO-intensive applications[17]. When developing machine learning services, FastAPI is able to handle the huge amount of training and prediction data with ease, which is perfect for needs.

### 4.1.3 Database: PostgreSQL

PostgreSQL is widely used in their respective areas of industry to build full-featured, efficient, and secure database management systems.

PostgreSQL is a cutting-edge open source object-relational database system recognized for its stability, high performance, and ability to handle complex queries[18]. It supports a wide variety of data types and shows high scalability, enabling users to create personalized functions and data types, and to write server-side code in a variety of programming languages. PostgreSQL adheres to the ACID standard, making it a useful tool for transaction processing systems with strict data integrity requirements[18]. At the same time, it provides excellent concurrency support in environments with frequent read and write operations through multi-version concurrency control (MVCC) technology, thereby improving efficiency. In addition, PostgreSQL is very scalable, so it is ideal for handling large amounts of data in enterprise applications. Since the data that IMOTION needs to store, such as latitude, speed, username, etc., relates only to the underlying data types, and there are associations between these data, a relational database becomes first choice with the need for security and extensibility.

### 4.1.4 Machine Learning: PyTorch

Developed by Facebook's AI Research Lab, PyTorch is an open-source machine learning platform distinguished by dynamic compute graphs, Python interface, GPU acceleration, large ecosystem of tools and modules, and active community [19]. Its dynamic computing graphs enable flexible model creation and runtime changes, hence simplifying debugging and enabling experimentation. Through CUDA's GPU acceleration, PyTorch greatly increases performance while effortlessly interacting with Python to simplify development and prototyping [19]. That qualifies it for training deep neural networks on vast amounts of data. Its large ecosystem and active community offer tools and resources for a range of activities and projects, which also assist to create deep learning models for this prototype.

### 4.1.5 Distributed microservice Framework: Docker& Kubernetes

Originally developed by Google, Kubernetes—also known as K8s—is an open-source orchestrating tool for containers. Large-scale seamless scalability, automated deployments, and competent containerized application handling are made possible by it[20]. Thanks to its integrated load balancing algorithms that help in the management, discovery, and access to application instance clusters[20], its potential to generate and oversee many containers accelerates operational and maintenance chores. Within the microservices space, Kubernetes offers administrative capabilities, scalability, and automated deployments that change the volume of service instances in real-time to effectively handle varying loads while simultaneously preserving best availability and reducing resource waste. Its sophisticated scheduling system also guarantees best use of resources by allocating them depending on the real demands of every container and service.

An open-source containerizing tool, Docker helps developers pack apps and their dependencies into agile, transferable containers ready for use on any widely used Linux system. Among numerous impressive capabilities of Docker are a secure container environment, little performance impact, virtualization, app isolation, and quick deployment[21]. Docker's containerizing in a microservices architecture lets every microservice run independently inside its own container, therefore guaranteeing continuous deployment and operations. This self-sufficiency increases the portability and responsiveness of microservices, therefore improving the fault tolerance and scalability of the system at once.

Microservices call for both Kubernetes and Docker as absolutely essential tools. By grouping microservices into separate containers, Docker simplifies their fast deployment and isolation. While Kubernetes manages and coordinates these containers, automating deployments, load distribution, and service identification Combining these technologies guarantees exceptional durability and flexibility, greatly simplifies microservices' operation and monitoring, thereby increasing developmental and deployment efficiency. Microservices architectures can quickly change to fit corporate demands by means of this symbiotic mix, scaling up or down as needed.

Consequently, running a variety of services on my server-based applications is easy; gradual launches of new services are just a matter of business.

### 4.1.6 Reflection

Building a microservices architecture for this application is justified by its open source nature. This method avoids the development of closed systems catering only to certain user groups and removes the additional expenses linked with outside services[22]. Given its closely tied character with many linked services and components, a monolithic program presents difficulties for other researchers who desire to change the software code[22]. Under a microservices design, however, researchers are free to add or change features without affecting the functionality of other areas. By using circuit breakers and auto scaling systems that dynamically modify application size depending on current demand, microservices also provide strong stability and disaster recovery features[23]. Though at the expense of higher complexity and a steeper learning curve, microservices allow programs to use several programming languages in areas where they are most appropriate; this trade-off produces a more flexible architectural design.

## 4.2 Transport Mode Detection

Transport mode detection is using a bottom-up strategy with a combination of convolutional neural network and merging algorithm. This strategy predict transport mode in every seconds based on the neighbour history of accelerometer data and merge into trip legs.

### 4.2.1 Machine Learning Model

The machine learning model implemented is inspired by CNN model designed by Liang in 2019[24]. This model efficiently and accurately identifies various transportation modes solely based on accelerometer data from mobile phones. Transport modes for predictions include still, walking, running, biking, bus, car, subway, and train[25]. In the experimental verification conducted in the paper, this model achieved an impressive accuracy of 94.48%[24], surpassing traditional machine learning techniques. Notably, due to its reliance only on accelerometer data, it exhibits minimal resource consumption on the phone system.

#### Data Processing & Preprocsessing

1. Data collection: This application determines the sampling frequency of the accelerometer based on user-defined privacy levels with a default setting of 100 samples per second(100hz).

2. Gravity Subtraction: Considering Earth's gravity as a universal influence affecting all objects uniformly; Because gravity is a stable value, using high-pass filter to eliminate gravity since it is irrelevant for traffic pattern recognition.

3. Data Smoothing: By employing the central moving average algorithm, acceleration data effectively reduce fluctuations caused by unexpected movements of mobile phones.

4. Data Window Sliding: To enhance adaptability to changes in phone position and orientation, acceleration magnitude is applied instead of specific axis data. Additionally, time segmentation of series data into fixed-size windows to meet real-time detection requirements.

**Model Construction**

This section demonstrates the design of the convolutional neural network (CNN) (Figure 4.1 and 4.2) for processing one-dimensional acceleration data, aiming to accurately identify users' transportation modes in different time periods.

1. Input layer: The network receives a one-dimensional acceleration data vector with a scale of 600×1 per time window.

2. Convolutional layers: Multiple convolution kernels are utilized to extract features from the input data. The first convolution layer consists of 32 convolution kernels sized at 15×1, and the convolution operation is performed with a step size of 1. Subsequently, the data undergoes dimensionality reduction through the Max pooling layer.

3. Fully connected layer: After multiple convolution and pooling operations, the data is reduced to 9×64 and then fed into a fully connected layer that further compresses its dimensions to 200×1.

4. Output layer: Finally, the transformed data becomes a 8×1 output vector representing probabilities associated with various transportation modes.

5. Network features: This network employs convolution and pooling operations for feature extraction in acceleration data while utilizing the fully connected layer for classification purposes, enabling accurate detection of users' transportation modes. To enhance model convergence speed, ReLU is adopted as an activation function along with Admin and CrossEntropyLoss algorithm and L2 regularization techniques to prevent overfitting.

6. Inference Process: Since the data is collected at 100 Hz, the 600 x 1 input represents 6 seconds of accelerometer magnitude data. When performing time-series prediction, the data will slide in 6-second windows, using the 6 seconds of data to predict one second of transport modes.

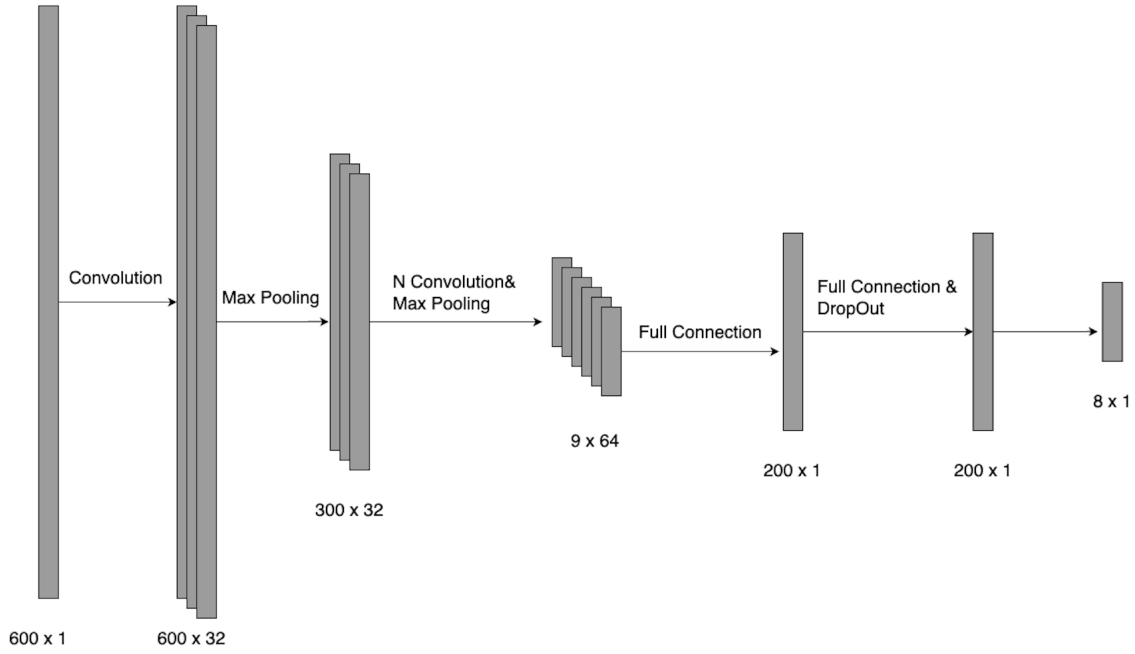


**Figure 4.1:** IMOTION Model Design

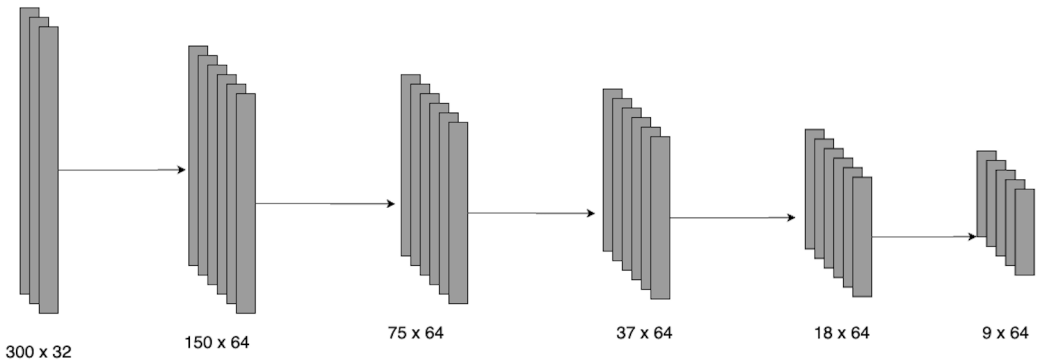


**Figure 4.2:** Hidden Layer Details

### 4.2.2 Merging Algorithm

After the predictions, data is changed to the sequence of transport modes with every seconds. Next step is to merge them to a bigger time sequences. However, because of the nature of incorrect predictions, some sequences of transport modes will be irregular. A merging algorithm in table 4.1 tries to address this problem.

The algorithm starts by encoding the initial transport mode sequence $\{m_1, m_2, \ldots, m_n\}$ and the corresponding time sequence $\{t_1, t_2, \ldots, t_n\}$ into the form $\{(m_j, t_j, t_j, w_j)\}$, where each $w_j$ is initialized to 1. Consecutive sequences of the same transport mode are merged into new tuples $(m'_j, t_{\text{start}}, t_{\text{end}}, w'_j)$, with $w'_j$ representing the total duration of the merged sequence. These merged sequences are classified as core sequences if $w'_j \geq$ MinTime and as normal sequences otherwise. Normal sequences are then modified to match the nearest core sequence with the largest $w'_j$. This process repeats until all sequences are core sequences. Finally, the core sequences are decoded into a format suitable for database storage.

| | |
|---|---|
| **Input:** | MinTime |
| **Step 1:** | Encode the initial transport mode sequence $\{m_1, m_2, \ldots, m_n\}$ and time sequence $\{t_1, t_2, \ldots, t_n\}$ to the form of $\{(m_j, t_j, t_j, w_j)\}$, where each $t_j$ are start time and end time and $w_j$ are initialized to 1(1 second). |
| **Step 2:** | Merge each neighbour sequence into a new $(m'_j, t_{start}, t_{end}, w'_j)$, where $m'_j$ is the transport mode label of the merged sequence and $w'_j$ is the number of seconds in the merged sequence. |
| **Step 3:** | The new merged sequence is a core sequence if ($w'_j \geq$ MinTime) or normal sequence if ($w'_j <$ MinTime). |
| **Step 4:** | Normal sequences modify their labels to the same as the neighbour core sequence with the larger $w'_j$. |
| **Step 5:** | Repeat step 4 until all remaining labels are core labels. |
| **Step 6:** | Decode the final core label sequence $\{(m'_p, t_{start}, t_{end}, w_p), \ldots, (m'_q, t_{start}, t_{end}, w_q)\}$ into the JSON string prepared to database. |

**Table 4.1:** design guidelines

## 4.3 Database

There are three main groups of tables in the database: general entity, raw data, and inference information.

In figure 4.3, there are two tables in general entity group:`User` and `Survey`. `User` table stores all identification information for authorization and authentication. The `houseHold` column stores encoded string for different family. Users can have the same encoded string if they are family. `privacyLevel` stores which privacy level

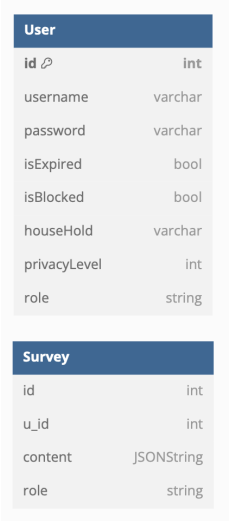


**Figure 4.3:** General Entity Tables

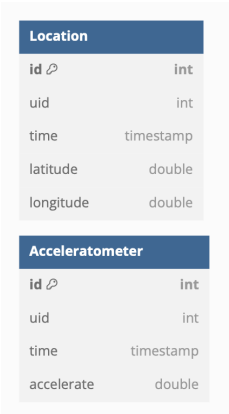


**Figure 4.4:** Raw Data Tables

users want to be. `Survey` table stores every survey where users answered or researchers proposed. `role` columns is to distinguish between normal users and researchers for different authentications. `u_id` in all tables in the database is `id` in `User` table to identify data for which user.

In figure 4.4, there are two examples of tables in the raw data group: `Location` and `Accelerometer`. These are only two raw data collecting in the prototype, but more data can be collected further in the future.

In figure 4.5, `Activity` and `Purpose` are tables in inference information group. `Activity` stores trips and travel modes with any information predicted by the model and put by users which researchers are interested in. `Purpose` stores any purpose or activities happened in each trip in `Activity`.

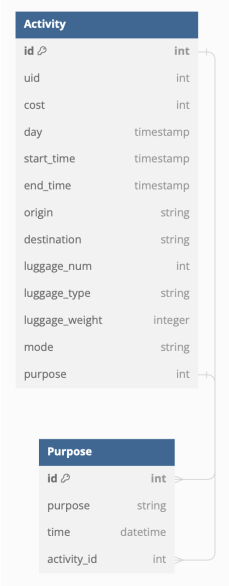


**Figure 4.5:** Inference Information Tables

## 4.4 Service Logic

### 4.4.1 Setup

When the user first enters the app, a loading page (Figure 4.6) featuring the "I-MOTION" logo will appear. During this page's display, the mobile system will prompt the user for network access permission and check if there is a stored token. If no token information is found, the page will redirect to the start page for registration or login.

If the user does not have an account, they can click the register button to proceed to the register page in Figure4.6. Users will first choose their desired privacy level. Then, they can register by entering their email address and a correct password.

The current prototype offers two privacy levels: self-use and research (Figure 4.6). When the privacy level is set to self-use, all data will be stored on the phone and predictions will be made using an embedded deep learning model. This ensures the confidentiality and security of the user's sensitive information but also means that backend services, such as community ranking, cannot be accessed. Additionally, it implies that GPS and accelerometer data will cause the app to be significantly larger than the research-level version. When the privacy level is set to research, all data will be sent to the backend server for further analysis and research. This also means that the service will be more comprehensive and the model's predictions more robust.

Once the user has their own account, they can log in via the login interface (Figure 4.7). If the backend detects that the user has not yet completed the primary survey, the page will redirect to the survey page for completion (Figure 4.7). After the sur-

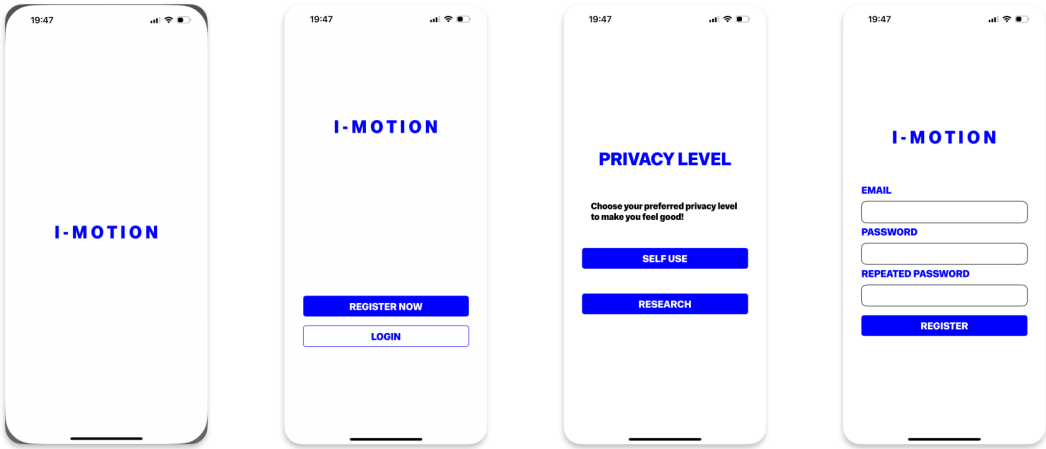


**Figure 4.6:** Launching and Register Pages

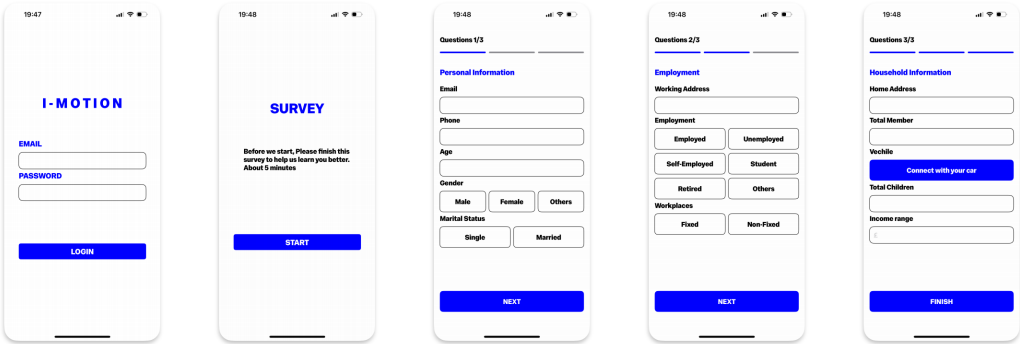


**Figure 4.7:** Login and Survey Pages

vey is completed, the user will be taken to the notification page to authorize the app to remind them daily to check and edit their activities from the previous day.

The user will then enter the main page for the first time. The system will request permission to use GPS and the accelerometer, and allow the app to continue using these sensors in the background.

### 4.4.2 Main Page

#### Track Page

The Track page(Figure 4.8) is the core page of this app. It consists of three parts: the calendar and location buttons at the top, the map in the middle, and the list view displaying trips at the bottom. Most of part of this page is interactive. So users will feel dominance and control of this application, which will positively affect users' evaluation of usability of this app[26].

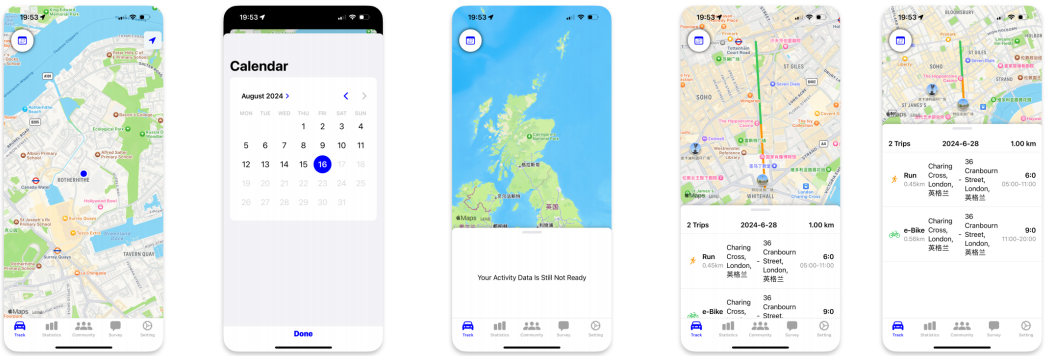


**Figure 4.8:** Track Page

### List View

The list view(Figure 4.8) will not show when the calendar date is set to today since the data for today is still under collecting. Should the user choose any past date, the Track page will load a loading indication and retrieve the data. Should data retrieval fail, the page will show ”Your activities are still not available” to let the user know the activity information for that particular day hasn’t yet been examined. The travels will show themselves in the list view once the data has been effectively obtained. A summary displaying the overall distance and trip count will be found at the top of the list view. With each row in the list view representing a journey and displaying approximate activity information including transportation options, start point, end point, and time interval, the trips in the list view are sorted chronologically morning through evening. To show more or less simultaneous trips, users can slide across the list view. Clicking any row will access the trip detail page.

### Trip Detail

In the trip detail page (Figure 4.9), users can see more detailed trip information and make modifications. Currently, the transportation modes that can be predicted include:

- Still
- Walk
- Run
- Bike
- Car
- Bus
- Subway
- Train

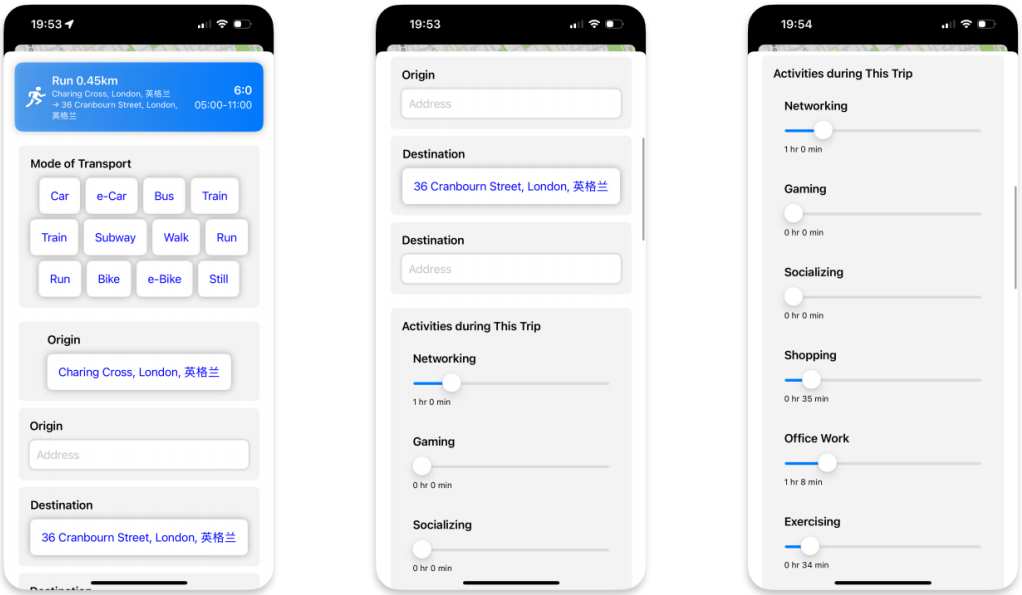


**Figure 4.9:** Trip Detail Page 1

Extra choices for e-car and e-bike have been made available for users in the trip details. Users in the origin and destination fields can either manually create a more exact location name or click on an already-existing location name.
There are 22 in-person activities(Figure 4.10) available:

- Networking
- Gaming
- Socializing
- Shopping
- Office Work
- Exercising
- Reading
- Dropping off/picking up children from school
- Traveling
- Visiting Parks/Playgrounds
- Studying
- Watching TV/Movies
- Listening to Music

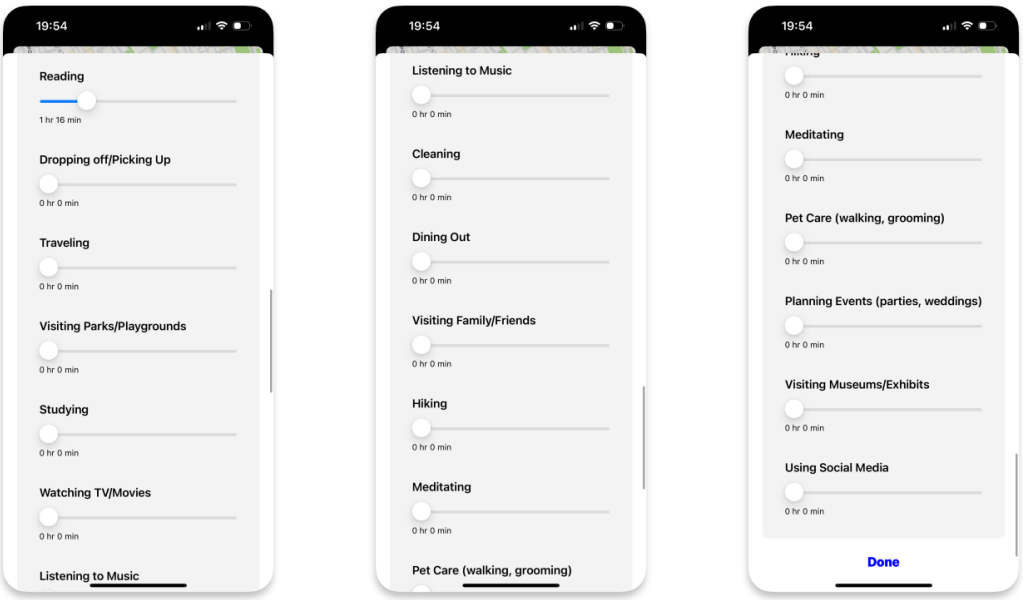


**Figure 4.10:** Trip Detail Page 2

- Cleaning
- Dining Out
- Visiting Family/Friends
- Hiking
- Meditating
- Pet Care (walking, grooming)
- Planning Events (parties, weddings)
- Visiting Museums/Exhibits
- Using Social Media

Users can drag buttons to confirm which activities they engaged in during the trip and how much time they spent on each. Finally, they can click “Done” to submit their changes.

### Statistics Page

Users of the Statistics Page(Figure 4.11) can graphically see their travel trends over a week, a month, and a year. It also shows other information such CO2 emissions and the expenses connected with their operations.

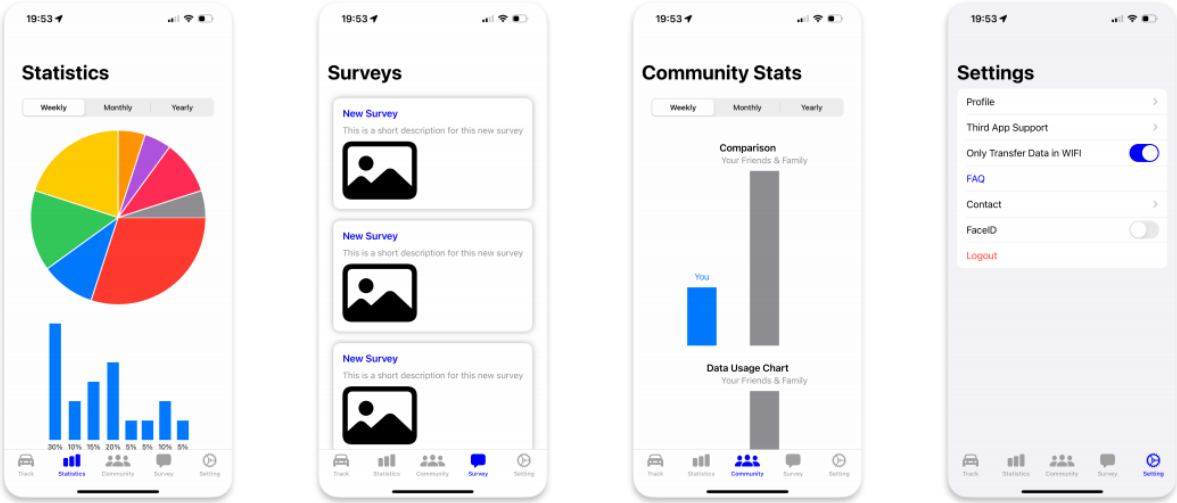


**Figure 4.11:** Statistics & Surveys & Community & Setting Page

### Community Page

Users of the Community Page(Figure 4.11) can graphically compare themselves with other people living in the same region or with individuals they know closely. This covers contrasts of public transit use, CO2 emissions, and electric vehicle use. The Community Page generates a virtual community or rating, therefore enhancing user communication [12]. With the aim of these comparisons improving user engagement and motivating more active and long usage of the app [26], it promotes a feeling of belonging.

### Survey Page

The Survey Page(Figure 4.11) lets researchers distribute different surveys or feedback requests to consumers, therefore letting interested people to finish them. Either the surveys are internal to the app or they link to outside survey sites. Apart from giving researchers more in-depth or targeted information, the Survey Page raises user involvement since consumers find feedback fascinating and significant [27]. Moreover, good feedback collecting can help users become less reluctant to provide GPS data[28].

### Setting Page

The user can manage their information, seek help, and manage the app's status through the settings page. There are seven features available: **Profile**, **Third Party Support**, **Only Transfer in WIFI**, **FAQ**, **Contact**, **FaceID**, and **Logout**.

- **Profile**: Allows users to manage their email address and password, as well as change their password.

- **Third Party Support**: Helps users manage connections to third-party APIs, such as the Smartcar API and Octopus Energy API, and potentially more APIs in the future to make it easier for users and researchers to access vehicle and energy usage information.
- **Only Transfer in WIFI**: A toggle switch that, when enabled, ensures that the user's data is only sent to the backend when connected to Wi-Fi, thus saving mobile data.
- **FAQ and Contact**: Direct users to the official IMOTION website where they can seek help and find more information.
- **FaceID**: When enabled, this feature allows users to bypass entering their password each time they start the app. It uses biometric information, such as face or fingerprint recognition, to automatically log in, enhancing security and providing a smoother user experience.
- **Logout**: When clicked, this option deletes all sensitive user information from the app, including account details, cache, GPS data, and accelerometer information.

### 4.4.3 Background Processing

When the user is using the app on the homepage or when the app is running in the background, the app continuously collects GPS information and 100Hz accelerometer data, storing it in the phone's storage. To save battery, the app is configured to update GPS data only when the movement distance exceeds 100 meters. If the user's movement remains under 100 meters for an extended period, the GPS data collection will automatically stop. When the system determines that it has collected sufficient data for the day, such as from 6 AM to 12 AM, and if the user's privacy level is set to research, the app will automatically send the data to the backend for further analysis and predictions. Once the app confirms that the backend has received the data, it will delete the stored data from that day to reduce storage pressure. If the privacy level is set to self-use, the data will be processed by the embedded model, and the results will be stored on the phone. Because all these processes are automatic, the user remains unaware of them, reducing the usage barrier and burden, thereby enhancing the user experience.

# Chapter 5

# Evaluation

The prototype and machine learning model have been evaluated through usability and performance testing to demonstrate the current user experience, system stability, and latency of the application, thereby providing direction for future development.

## 5.1 User Testing

The user testing in this project are based on a formative study with a small group of participants in order to provide continuous feedback and improvement during the development process[29]. The user testing consisted of two parts: UI testing and usability testing.

UI testing was conducted by creating a virtual front-end interface using a GUI tool such as Figma, and participants validated the design and layout through open-ended questionnaire feedback[30]. The main goal of the testing was to ensure that the UI design was user-centred, focusing on three key areas: consistency, responsiveness and accessibility[30].

Usability testing was conducted in an unsupervised remote testing environment[31], as the software is a travel diary application, and the remote testing environment was able to more realistically simulate the actual user experience. Although unsupervised testing may make it difficult for the researcher to capture and understand the motivations of the participants' behaviour[29], which in turn increases the complexity of the activity, being able to cope with complex operations is also one of the important metrics to be evaluated for this software system. Usability testing will use a closed-ended questionnaire based on the System Usability Scale (SUS) [29] to calculate scores for each metric as well as a total score to determine room for improvement in each feature and overall usability.

### 5.1.1 Setup

This test is composed of four participants, including 2 men and 2 women. These four participants will simultaneously perform both UI Testing and Usability Testing. During the testing, the participants are required to follow a fixed app usage process to ensure that no UI functionality is left untested. The process includes the following steps: **register** → **login** → **primary survey** → **track** → **trip detail** → **statistics** → **community** → **surveys** → **setting**.

After completing the UI test journey, the participants will be asked to answer 6 post-task questions to provide feedback. The result will be showed in Appendix A. These 6 questions are as follows:

1. When you first got into the start page, how would you describe your experience while trying to create a new account and log in? Were there any steps that felt unclear or confusing?
2. When you started the primary survey, how intuitive did you find the process of answering questions? Were there any parts of the survey interface that you felt could be improved for better usability?
3. In the main page, how intuitive did you find every feature by icons or colors without guidance? Did you think the statistics, community, and survey pages encouraged you to use this application more frequently?
4. In the track page, do you think the layout design is too complex or not? Is it easy to help you get your personal travel information in the trip list view? Could you quickly learn how to use this page based on the navigation provided?
5. In the trip detail page, how smooth was your experience when entering details about your transportation and activities? Does the interface allow you to easily input and review your trip data?
6. what other feature do you think the app should have?

After completing the Usability test journey, the participants will be asked to finish a System Usability Scale(SUS). The format of SUS and resulted value will be in Appendix B.

### 5.1.2 Result of UI Testing

The general UI experience is good for users but there is still a lot to improve. Overall, there are clear and intuitive design elements and the operation is simple and easy to understand. Users generally agreed that the account creation and login process was simple, but noticed that overly technical terms in terms of privacy Settings were confusing, and wanted more color differentiation in the interface. The survey process was considered to be intuitive. But some users felt that the answer options were

too limited, especially in terms of gender and employment status. The home page and its ICONS were well received and perceived as intuitive and recognizable, possibly encouraging frequent use, but users indicated that more detailed explanation of functionality was needed. The track page was praised for its clean layout. However, almost everyone found the trip details page too complex and worried about too much information and complicated input. As a non-research-oriented user, there is resistance to these complex trip details input. So a simpler UI should be designed for such target users. Finally, it is suggested to add some features, including a social function to connect with friends and a history function to avoid duplicate data viewing.

### 5.1.3 Result of Usability Testing

The IMOTION prototype's feedback and system usability scale (SUS) findings reveal that users typically feel the program has a good interface, easy to learn and high functional integration. Most users said they were confidence in their ability to operate simply without technical help and were ready to use the program regularly. Features like privacy settings and community involvement capabilities, which inspired users to use the service more. Some users, meantime, expressed worries about the intricacy of the application—including battery waste resulting from background running—as well as usability problems such not being able to instantly check same-day trip notes. Though the app is useful among users, if these particular problems can be resolved, it would assist to raise user experience.

## 5.2 Machine Learning Performance

### 5.2.1 Setup

Sussex-Huawei Locomotion Challenge 2018 [25] provides the data resource for this evaluation. The training data is a matrix with dimensions $16,310 \times 6,0000$. This matrix represents 16,310 frames, where each frame contains 6,000 samples (equivalent to 1 minute of data recorded at a sampling rate of 100 Hz), which equals to 11 days. The corresponding label matrix shares the same dimensions ($16,310 \times 6,000$), with each value indicating the transportation activity associated with each sample. There are 8 different activity classes, represented by the following numbers:

- 1 – Still
- 2 – Walk
- 3 – Run
- 4 – Bike
- 5 – Car
- 6 – Bus

- 7 – Train
- 8 – Subway

Apart from its size— $5698 \times 6000$—the testing dataset has the same structure as the training dataset. This comes to 4 days because 5,698 frames, each with 6,000 samples, correspondingly. The respective label matrix has the same dimensions.

### 5.2.2 Training Performance

Effective learning is demonstrated by CNN model's training performance shown by the loss curve in Figure 5.1. The early epochs' fast drop in loss suggests that the model picks up the fundamental trends in the data fast. The curve starts to level as training goes on, implying that the model is getting close to its ideal performance—where more study produces little effect. This behavior along with the CNN architecture of the model and the application of ReLU activation, Admin optimizer, and L2 regularization point to a well-tuned model. The training process of the model seems strong and shows that it has acquired the required characteristics to carry out its purpose efficiently.

### 5.2.3 Test Performance

Prior to combining the Train (Class 7) and Subway (Class 8) classes in figure 5.2 and figure 5.3, the model's test performance demonstrated moderate effectiveness in distinguishing between the various modes of transportation, which included Still (Class 1), Walk (Class 2), Run (Class 3), Bike (Class 4), Car (Class 5), Bus (Class 6), Train (Class 7), and Subway (Class 8). The accuracy was around 73.08%, with precision of 74.17% and recall of 73.08%. The F1 score, which balances precision and recall, was 73.43 percent. The confusion matrix shows that, while the model was usually successful at identifying stationary states (Still) and typical movement patterns (Walk, Run, Car, Bus), it struggled more with distinguishing between similar transport modes such as Train and Subway. These two classes most likely share many similar properties in the acceleration data, resulting in a higher risk of misclassification between them.

The performance measures of the model much improved once Train and Subway were merged into a single class in figure 5.4 and figure 5.5. With comparable rises in precision and recall to 79.09% and 78.73%, respectively, the accuracy climbed to 76.30% and the F1 score jumped to 78.88%. This development implies that the model found it simpler to classify the combined Train and Subway class than it would have separating between the two. The fewer classes simplified the categorization process, therefore lowering ambiguity and improving general performance. The model increased in its capacity to differentiate between the remaining motorized modes, such Car and Bus, and still performed well in spotting non-motorized modes, such Still and Walk. This shift emphasizes the difficulty of precisely categorizing rather comparable transit options and shows how purposeful class consolidation could result in stronger models.

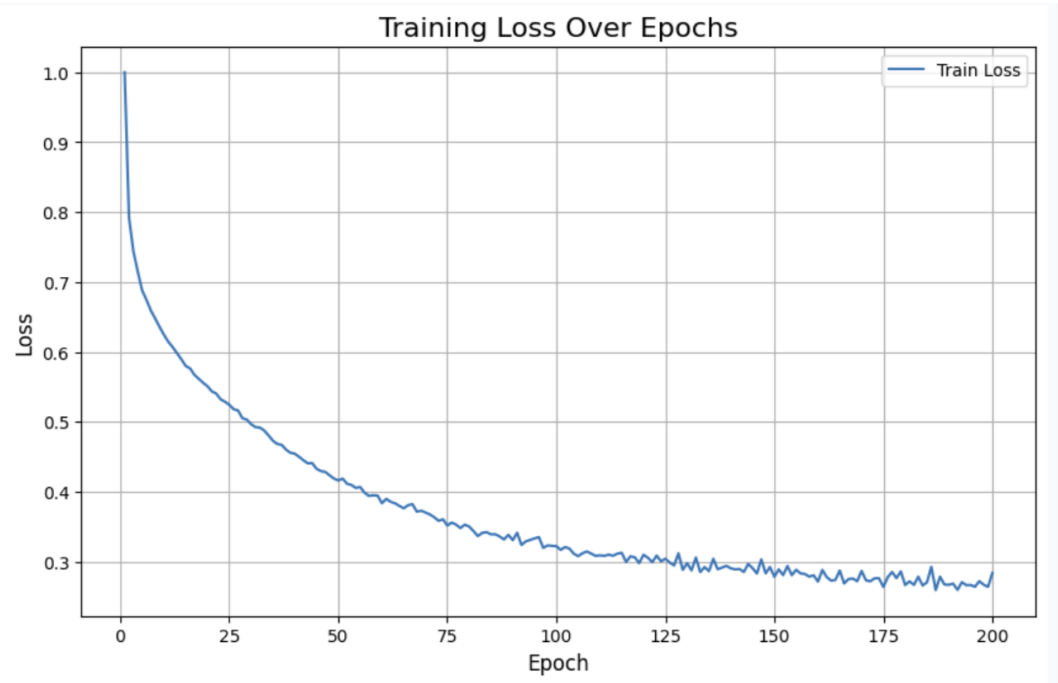


**Figure 5.1:** Loss Curve Over Epochs

```
Results  - Accuracy: 0.7308, Precision: 0.7417, Recall: 0.7308, F1 Score: 0.7343
```

**Figure 5.2:** Results of Test Before Combination

| | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Class 7 | Class 8 |
|---|---|---|---|---|---|---|---|---|
| 0 | 72.890317 | 1.337342 | 0.020734 | 1.109268 | 2.488078 | 1.513581 | 9.205888 | 11.434792 |
| 1 | 1.787925 | 92.256911 | 0.261312 | 1.939211 | 0.453858 | 0.495118 | 0.921469 | 1.884197 |
| 2 | 0.118343 | 2.100592 | 95.680473 | 1.804734 | 0.000000 | 0.147929 | 0.059172 | 0.088757 |
| 3 | 0.936768 | 2.283372 | 0.195160 | 92.661983 | 1.307572 | 1.229508 | 0.644028 | 0.741608 |
| 4 | 0.689169 | 0.070483 | 0.015663 | 0.250607 | 78.847208 | 15.568956 | 1.621114 | 2.936800 |
| 5 | 1.814337 | 0.812556 | 0.033393 | 1.179875 | 19.813001 | 64.781834 | 4.263134 | 7.301870 |
| 6 | 12.359204 | 1.388675 | 0.015430 | 0.740626 | 11.525999 | 9.813300 | 40.719025 | 23.437741 |
| 7 | 19.127820 | 0.812030 | 0.030075 | 0.571429 | 5.563910 | 4.210526 | 27.849624 | 41.834586 |

**Figure 5.3:** Confusion Matrix Before Combination

```
Results  - Accuracy: 76.30%, Precision: 79.09%, Recall: 78.73%, F1 Score: 78.88%
```

**Figure 5.4:** Results of Test After Combination

| | Class 1 | Class 2 | Class 3 | Class 4 | Class 5 | Class 6 | Class 7 |
|---|---|---|---|---|---|---|---|
| 0 | 72.890317 | 1.337342 | 0.020734 | 1.109268 | 2.488078 | 1.513581 | 20.640680 |
| 1 | 1.787925 | 92.256911 | 0.261312 | 1.939211 | 0.453858 | 0.495118 | 2.805666 |
| 2 | 0.118343 | 2.100592 | 95.680473 | 1.804734 | 0.000000 | 0.147929 | 0.147929 |
| 3 | 0.936768 | 2.283372 | 0.195160 | 92.661983 | 1.307572 | 1.229508 | 1.385636 |
| 4 | 0.689169 | 0.070483 | 0.015663 | 0.250607 | 78.847208 | 15.568956 | 4.557914 |
| 5 | 1.814337 | 0.812556 | 0.033393 | 1.179875 | 19.813001 | 64.781834 | 11.565004 |
| 6 | 14.654293 | 1.193147 | 0.020396 | 0.683255 | 9.504385 | 7.913522 | 66.031001 |

**Figure 5.5:** Confusion Matrix After Combination

### 5.2.4 Reflection

The great performance of the training loss suggests that CNN model is competent in evaluating and interpreting acceleration data of time-series. The test findings, however, imply that the model finds it difficult to distinguish between modes of transportation with like operation patterns. Two scenarios exist now that might cause this performance drop. The first scenario is that the sampling period is far too short. According to experimental data, even if six seconds is enough to precisely differentiate between several forms of transportation, it might not be enough to record variations across like modes, such train and subway. Increasing the sampling length might help to enhance accuracy. The second alternative is that the approach applied for gravitational removal is stationary. A static filter might not efficiently eliminate gravity and might potentially contribute erroneous data since the phone continuously changes orientation. Improved model performance results from more effective early stage differentiation of the data using a more complicated hybrid filter.

# Chapter 6

# Conclusion

The main objective of this report is to provide an off-the-shelf, easy to modify and improve travel diary system solution for mobile phones called **IMOTION**. The open-source nature of IMOTION allows the research community to easily deploy and use it for purposes such as travel diary extraction, destination or usage analysis, trajectory segmentation, and any other research that uses trajectory data or travel diaries. The design and principles of the IMOTION architecture and its use are explained in detail in the report. A series of tests are provided to demonstrate its usability and effectiveness.

IMOTION has four innovations over other similar projects:

1. **Open Source Modular Microservices System:** IMOTION is an open-source modular microservices system. The advantage of this system design is its convenience and flexibility. Developers can add more components to meet their needs without affecting the overall system operation. All modules are packaged in Docker containers and scheduled by Kubernetes, allowing developers to deploy their solutions in essentially any development language and on any operating system environment. Front-end, back-end, and data developers can focus on their own development separately.

2. **User Experience Focus:** IMOTION is one of the few research-based travel diary collection systems that emphasises user experience. Its UI design enhances user experience and application engagement through easy-to-understand linear guides, recognisable colours and graphics, and timely feedback.

3. **Transport Pattern Detection:** IMOTION's transport pattern detection is a bottom-up strategy that combines a convolutional neural network (CNN) with a merging algorithm, using only accelerometer data. Reduced sensor usage translates to lower power consumption for the phone. From evaluations, CNN is proficient at detecting time-series data. Moreover, the model's reliance solely on CNN facilitates its embedding into mobile devices. The merging algorithm effectively mitigates the issue of irregular sequences due to inaccurate predictions.

4. **Privacy Level Mechanism:** The introduction of a privacy level mechanism encourages participation from users of diverse backgrounds while safeguarding their security and privacy. In this framework, deep learning models are deployed both on the mobile device and backend, enabling the system to function at varying privacy levels.

## 6.1 Limitations

While I-MOTION basically fulfilled objectives in Section 1 and Designs in Section 3, it is important to announce that I-MOTION is still a in-progress prototype and still remains some limitations for further improvement.

### 6.1.1 Third-party Plugin

Mobile device data is limited, hence different third-party plugins are necessary to provide more varied data collecting in product design. For most car brands, Smartcars can deliver information on fuel and power use; Octopus can offer statistics on home energy consumption. One of the goals for next development is to have the present prototype incorporate these interfaces, which it does not presently do.

### 6.1.2 Implementation of Statistics and Community Page

Some participants could be reluctant to participate in research if they are not sure how it will help society [32]. Consequently, one should create statistical and communal functions. These features will give participants easy feedback including the monetary and data costs produced over a time as well as the percentage of public green transportation use against private car use over a given period. This comments enables people to evaluate how their acts affect society as well as oneself.

The community function will also include comparisons between the participant's energy consumption and sustainability practices and those of others in the same area or with whom they have close relationships. This comparison will generate positive feedback, giving participants a sense of purpose in using the software and encouraging them to actively engage in the research process or develop daily habits of using the software. As a result, participants will develop a positive impression of the research and be more inclined to participate in future studies.

### 6.1.3 Watch Connectivity

Based on smart watch UK market report by statista in 2024[33], the user base and product penetration of smartwatches are steadily increasing each year. Regardless of gender, age, or income, the usage rate of smartwatches is almost equal, indicating that smartwatches are becoming as common as smartphones. Smartwatches are valuable hardware because their configuration and usage differ from smartphones.

Data collected from a watch's accelerometer can complement or replace smartphone data, leading to more accurate predictions and reducing the phone's battery usage. This report emphasizes the use of smartwatches in design, although the prototype does not yet include smartwatch integration. Future development aims to incorporate smartwatches as a new hardware tool.

### 6.1.4 Languages

Based on Population Census in 2021 [34], there are total 19 ethnic groups in United Kingdom, which 81.7% are white ethnic group and 28.3% are other ethnic groups. This demonstrates that the population of the UK is highly diverse. Despite being an English-speaking country, the significant multilingual population structure reminds us that providing multilingual translation in software is essential to obtain a larger experimental sample for comprehensive analysis.

## 6.2 Ethical and Professional Considerations

This part will talk about the professional and moral issues that were thought about while the project was being planned and evaluated. First, it's important to collect and store user info. To make a profile, users must verify their identity via email and put personal details like their name, job title, home address, location, and personal connections. According to GDPR rules from the UK Information Commissioner's Office (ICO) [35], people have the right to know how their personal data is collected and used. Personal data is any information that can be used to find out who someone is, like their name or where they live [36]. The IMOTION project meets these standards because it tells users clearly before they sign up what the purpose of collecting data is. Protecting personal information is just as important in usability tests. End-user test surveys only asked about age range and device details from participants so as not to collect personal information.

Second, GPS tracking and accelerometer data collection must be done carefully when it comes to security and privacy problems [35]. Users must explicitly give permission for the app to use GPS trackers, and it shows a real-time location-tracking pointer on its interface to let users know that it is collecting information about their location.

Finally, it is essential to conduct rigorous scrutiny of algorithms, particularly in the context of machine learning APIs, in order to identify and mitigate any potential biases. It is crucial that technology ensures inclusivity and equity by delivering fair outcomes irrespective of gender, race, age or socioeconomic status[37]. Simultaneously, transparency regarding the employed AI and machine learning models should be upheld through public availability of their decision-making processes and comprehensive documentation for review purposes. Furthermore, supporting external audits becomes essential to guarantee adherence to ethical standards[38].

## 6.3 Future Works

In addition to the limitations mentioned above, design, implementation, and testing have revealed many opportunities to improve the application's usability and stability. It is hoped that these improvements can be realized in future development.

### 6.3.1 Administration Web for researchers

Researchers often need to customize software to meet specific research requirements, such as adjusting data collection methods, tailoring participant feedback, and aligning the interface with study goals. They also require the ability to manipulate the database, including extracting and analyzing data to uncover patterns or trends and adapting data collection based on initial findings. Additionally, monitoring system performance and usage statistics is crucial to ensure the software's reliability and the study's integrity. To address these needs, an administrator web page is necessary, providing a centralized platform for customization, data management, and system monitoring, thereby enabling efficient and effective research.

### 6.3.2 Progressive Learning Technique for Deep Learning Multi-Classification Network

The IMOTION system will allow the user to modify as many modes of transport as possible in the future because transport modes change rapidly. Therefore, neural networks should evolve with the times.Progressive learning is one of the best approaches. It dynamically adjusts the structure of the neural network when there is a new label to learn new categories while retaining previous knowledge[39]. This solves the weakness of the two common learning methods (batch and sequential learning)[40]. Batch learning requires a large amount of known data to enable the neural network to learn the optimal parameters in a batch[39]. However, for IMOTION system, each modification by the user means a new real data is generated. In this situation, Batch learning produces a serious delay because a sufficient amount of data needs to be collected first before the model can be retrained. The user can not get the latest accurate model in time. Although Sequential Learning can learn new real data itratively, it is able to learn new labels because the model structure is static[39].

### 6.3.3 Message Queuing for Large Data Processing

Because gathering 24-hour GPS and accelerometer data creates a big amount of information, delivering it all at once might put a load on the backend system. During testing, it was discovered that the machine learning service sometimes gets overloaded, resulting in system failures in which the service crashes and restarts. This kind of overload not only affects the flow of data processing, but it also increases the danger of losing crucial data or delaying the research process.

To overcome this issue, establishing a message queuing mechanism for massive data processing can greatly reduce the backend workload[41]. Message Queuing works by dividing down enormous amounts of data into smaller, more manageable chunks that are transmitted progressively rather than all at once. This managed data flow keeps the backend system from getting overburdened, lowers the likelihood of service failures, and increases overall system stability. It also provides scalability and fault tolerance by guaranteeing that data is handled even when momentary challenges develop, hence maintaining the integrity and continuity of the data gathering process.

### 6.3.4 Big Data System

As the number of people and the duration of the research grows, it inevitably leads to an increase in the size of the data, an increase in the cost of maintaining the database, and an increase in the number of non-hot-spot data that increasingly dominate the database, leading to an inevitable decrease in the speed of retrieval. So there is a need for an easily developed and maintained big data system to store and manage the huge amounts of non-hot but meaningfully analyzed data (e.g. accelerometer data) for subsequent extraction and analysis by researchers. There are many options here. Big data systems can be created using the open source Apache Hadoop[42], which provides a complete solution for storage (Hadoop Distributed File System) and retrieval (Apache Hive[43]), and has capabilities for data analysis and machine learning (Apache Spark). The good thing is that the code is completely autonomous and controlled, the bad thing is that it is expensive to maintain. Or use private cloud technologies like amazon's DynamoDB[44] and google's BigQuery[45]. the good thing is that the maintenance cost is very low, but it involves data sensitivity issues.

# Bibliography


[1] McCool D, Lugtig P, Mussmann O, Schouten B. An app-assisted travel survey in official statistics: Possibilities and challenges. Journal of Official Statistics. 2021;37(1):149-70. pages 2

[2] Alho A, Cheng C, Hieu DT, Sakai T, Zhao F, Ben-Akiva M, et al. Online and in-person activity logging using a smartphone-based travel, activity, and time-use survey. Transportation Research Interdisciplinary Perspectives. 2022;13:100524. pages 2, 5, 7

[3] Prelipcean AC, Susilo YO, Gidófalvi G. Collecting travel diaries: Current state of the art, best practices, and future research directions. Transportation Research Procedia. 2018;32:155-66. pages 2, 3

[4] Keusch F, Bähr S, Haas GC, Kreuter F, Trappmann M, Eckman S. Non-participation in smartphone data collection using research apps. Journal of the Royal Statistical Society Series A: Statistics in Society. 2022;185(Supplement_2):S225-45. pages 3

[5] Cottrill CD, Pereira FC, Zhao F, Dias IF, Lim HB, Ben-Akiva ME, et al. Future mobility survey: Experience in developing a smartphone-based travel survey in Singapore. Transportation Research Record. 2013;2354(1):59-67. pages 5, 6, 7, 8

[6] Hong S, Zhao F, Livshits V, Gershenfeld S, Santos J, Ben-Akiva M. Insights on data quality from a large-scale application of smartphone-based travel survey technology in the Phoenix metropolitan area, Arizona, USA. Transportation Research Part A: Policy and Practice. 2021;154:413-29. pages 5, 6

[7] Harding C, Faghih Imani A, Srikukenthiran S, Miller EJ, Nurul Habib K. Are we there yet? Assessing smartphone apps as full-fledged tools for activity-travel surveys. Transportation. 2021;48:2433-60. pages 6

[8] Mahajan V, Kuehnel N, Intzevidou A, Cantelmo G, Moeckel R, Antoniou C. Data to the people: a review of public and proprietary data for transport models. Transport reviews. 2022;42(4):415-40. pages 6, 7

[9] Geurs KT, Thomas T, Bijlsma M, Douhou S. Automatic trip and mode detection with move smarter: First results from the dutch mobile mobility panel. Transportation research procedia. 2015;11:247-62. pages 8

[10] Norman D. The design of everyday things: Revised and expanded edition. Basic books; 2013. pages 11

[11] Kurniawan S. Interaction design: Beyond human–computer interaction by Preece, Sharp and Rogers (2001), ISBN 0471492787. Universal Access in the Information Society. 2004;3:289-9. pages 11

[12] Hsu TH, Tang JW. Development of hierarchical structure and analytical model of key factors for mobile app stickiness. Journal of Innovation & Knowledge. 2020;5(1):68-79. pages 11, 34

[13] Gil-López T, Christner C, de León E, Makhortykh M, Urman A, Maier M, et al. Do (not!) track me: Relationship between willingness to participate and sample composition in online information behavior tracking research. Social science computer review. 2023;41(6):2274-92. pages 14

[14] Di Matteo D, Fine A, Fotinos K, Rose J, Katzman M, et al. Patient willingness to consent to mobile phone data collection for mental health apps: structured questionnaire. JMIR mental health. 2018;5(3):e9539. pages 14

[15] Inc A. Swift Programming Language; 2023. Accessed: 2023-10-23. `https://developer.apple.com/swift/`. pages 21

[16] Pivotal Software. Spring Framework – Home; 2024. Accessed: 2024-05-12. Available from: `https://spring.io`. pages 22

[17] Tiangolo S. FastAPI;. Accessed: 2023-10-27. `https://fastapi.tiangolo.com`. pages 22

[18] Group PGD. PostgreSQL; 2023. Accessed: 2023-10-23. `https://www.postgresql.org/`. pages 22

[19] Contributors P. PyTorch; 2023. Accessed: 2023-10-23. `https://pytorch.org/`. pages 23

[20] Kubernetes;. Accessed: 2023-10-27. `https://kubernetes.io`. pages 23

[21] Docker;. Accessed: 2023-10-27. `https://www.docker.com`. pages 23

[22] Al-Debagy O, Martinek P. A comparative review of microservices and monolithic architectures. In: 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics (CINTI). IEEE; 2018. p. 000149-54. pages 24

[23] Razzaq A, Ghayyur SA. A systematic mapping study: The new age of software architecture from monolithic to microservice architecture—awareness and challenges. Computer Applications in Engineering Education. 2023;31(2):421-51. pages 24

[24] Liang X, Zhang Y, Wang G, Xu S. A deep learning model for transportation mode detection based on smartphone sensing data. IEEE Transactions on Intelligent Transportation Systems. 2019;21(12):5223-35. pages 24

[25] Sussex-Huawei. SHL Dataset - Activity Recognition Challenge; 2018. Accessed: 2024-08-06. `http://www.shl-dataset.org/activity-recognition-challenge/`. pages 24, 38

[26] Furner CP, Racherla P, Babb JS. Mobile app stickiness (MASS) and mobile interactivity: a conceptual model. The Marketing Review. 2014;14(2):163-88. pages 30, 34

[27] Wenz A, Jäckle A, Burton J, Couper MP. The effects of personalized feedback on participation and reporting in mobile app data collection. Social Science Computer Review. 2022;40(1):165-78. pages 34

[28] Tsai JY, Kelley P, Drielsma P, Cranor LF, Hong J, Sadeh N. Who's viewed you? The impact of feedback in a mobile location-sharing application. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems; 2009. p. 2003-12. pages 34

[29] Barnum CM. Usability testing essentials: Ready, set... test! Morgan Kaufmann; 2020. pages 36

[30] Hotjar. UI Testing: What It Is and How to Do It; 2023. Accessed: 2024-08-21. Available from: `https://www.hotjar.com/ui-design/testing/`. pages 36

[31] Group NN. Usability Testing 101; 2024. Accessed: 2024-08-21. Available from: `https://www.nngroup.com/articles/usability-testing-101/`. pages 36

[32] Winkler C, Meister A, Axhausen KW. The TimeUse+ data set: 4 weeks of time use and expenditure data based on GPS tracks. Transportation. 2024:1-27. pages 43

[33] Statista. Smartwatches - United Kingdom — Statista Market Forecast; 2024. Accessed: 2024-08-07. `https://www.statista.com/outlook/hmo/digital-health/digital-fitness-well-being/fitness-trackers/smartwatches/united-kingdom`. pages 43

[34] Office for National Statistics. Population of England and Wales; 2024. Accessed: 2024-08-07. `https://www.ethnicity-facts-figures.service.gov.uk/uk-population-by-ethnicity/national-and-regional-populations/population-of-england-and-wales/latest/`. pages 44

[35] Office IC. Right to be informed;. Accessed: 2024-05-31. `https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/personal-information-what-is-it/what-is-personal-information-a-guide/`. pages 44

[36] Office IC. What is Personal Information? A Guide;. Accessed: 2024-05-31. https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/personal-information-what-is-it/what-is-personal-information-a-guide/. pages 44

[37] AI H. High-level expert group on artificial intelligence. Ethics guidelines for trustworthy AI. 2019;6. pages 44

[38] Office IC. Artificial Intelligence and Data Protection;. Accessed: 2024-05-31. https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/artificial-intelligence/. pages 44

[39] Venkatesan R, Er MJ. A novel progressive learning technique for multi-class classification. Neurocomputing. 2016;207:310-21. pages 45

[40] Vibrantish. Understanding Batch and Online Learning in Machine Learning. Medium. 2024. Accessed: 2024-09-02. Available from: https://medium.com/@vibrantish/understanding-batch-and-online-learning-in-machine-learning-84817e9dd9fd. pages 45

[41] Apache Software Foundation. Apache Kafka; 2024. Accessed: 2024-08-16. https://kafka.apache.org/. pages 46

[42] Apache Software Foundation. Apache Hadoop; 2024. Accessed: 2024-08-16. https://hadoop.apache.org/. pages 46

[43] Apache Software Foundation. Apache Hive; 2024. Accessed: 2024-08-16. https://hive.apache.org/. pages 46

[44] Amazon Web Services, Inc . Amazon DynamoDB; 2024. Accessed: 2024-08-16. https://aws.amazon.com/cn/dynamodb/. pages 46

[45] Google Cloud. BigQuery: Cloud Data Warehouse; 2024. Accessed: 2024-08-16. https://cloud.google.com/bigquery. pages 46

# Appendix A

# User Testing Survey

## A.1 UI Testing Open-Ended Survey

### A.1.1 When you first got into the start page, how would you describe your experience while trying to create a new account and log in? Were there any steps that felt unclear or confusing?

- ”Everything seems pretty clear to me as I see the images of the start page flow. However, I am unsure about the details of the privacy level in the second image.”
- ”All steps are clear, its easy to creature new account. Could be better if back buttons are available.”
- ”I think the privacy level is a bit unclear, may be use bullet points to show the difference?”
- ”I think the color is consistent that is a very good point. but i don’t know how to say IT is seems like too simple. and the bluesim boys really like the symbol of the imperial college. maybe we can use two or three colors to represent the function and the choice?”

### A.1.2 When you started the primary survey, how intuitive did you find the process of answering questions? Were there any parts of the survey interface that you felt could be improved for better usability?

- ”All of the questions are clear and straight, but I feel the choices may be too restricted for participants to answer. For example, the gender part should give another choice (like the other and leave a blank for them to write their gender representation) since some people are very sensitive to that and to avoid any

future conflicts. The workplace (hybrid) and employment status (part-time) should also be considered for giving more choices."

- "Answers are intuitive. Address input could be splited into several lines to fit long input."
- "I think may be add more explanations about "total member/children"
- "i don't know if there is a go back function to the previous question but i believe it already have."

### A.1.3 In the main page, how intuitive did you find every features by icons or colors without telling you? Did you think statistics, community, and survey page help you to use this application more frequently?

- "I found this main page interesting. I like the icons and color; it's pretty intuitive. I also feel those four interfaces/functions will attract me more frequently, but I need to know their accuracy and detailed functions instead of just seeing pictures or names."
- "Icons are easy to understand. Each page is meaningful and helpful."
- "I think its clear and very intuitive"
- "Yes. They are clear and easy to understand, which encourage me to use"

### A.1.4 In the track page, do you think the layouts of design which is too complex or not? Is it easy to help you get your personal travel information in trip list view? Could you learn quickly how to use this page based on navigation?

- "On the track page, I think the layouts of the design are pretty straightforward; the information on the page is not very ambiguous or hard to understand."
- "Layout is good, trip list is clear and comprehensive. Navigation do help to get started."
- "I think its good"
- "The trip details are a little complex. Too many fields to fill"

### A.1.5 In the trip detail page, how smooth is your experience when entering details about your transportation and activities? Does the interface allow you to easily input and review your trip data?

- ”I am not sure of the purpose of this detail page. Is it for data correction, or does it require me to enter every information? Since I think this app is auto-collecting data from my phone, I believe this page is for data showing and correcting. If this page requires me to write, I would not prefer to use this app.”
- ”It is easy to fill all necessary info. Interface is user-friendly.”
- ”sorry to say that but i think i will be lazy to do the second figure things is too complex for me and i always like to do the things in app within a few minutes so if one page cannot let me to use in few seconds. i think i may skip it directly. for me i do not think the different color for different travel way is a good choice but maybe you like it? i do not think IT is a very good point to put a lot of information in one page. i think the second figure can be switch to the interface that put in the previous question, a progressive one,once customer have answer one question were to question they jumped to the another interface. this strategy make people willing to do the questionnaire or much more complex information.”
- ”A little complex but easy to understand so it’s good”

### A.1.6 What other feature do you think the app should have?

- ”I feel this app also could add a friend function like Zenly; you can share information with your friends and communicate with them. Building a community in this app could help it be more successful.”
- ”Travel info history, such that users don’t need type same info over and over again.”
- ”All good.”
- ”No suggestion”

# Appendix B

# System Usability Scale

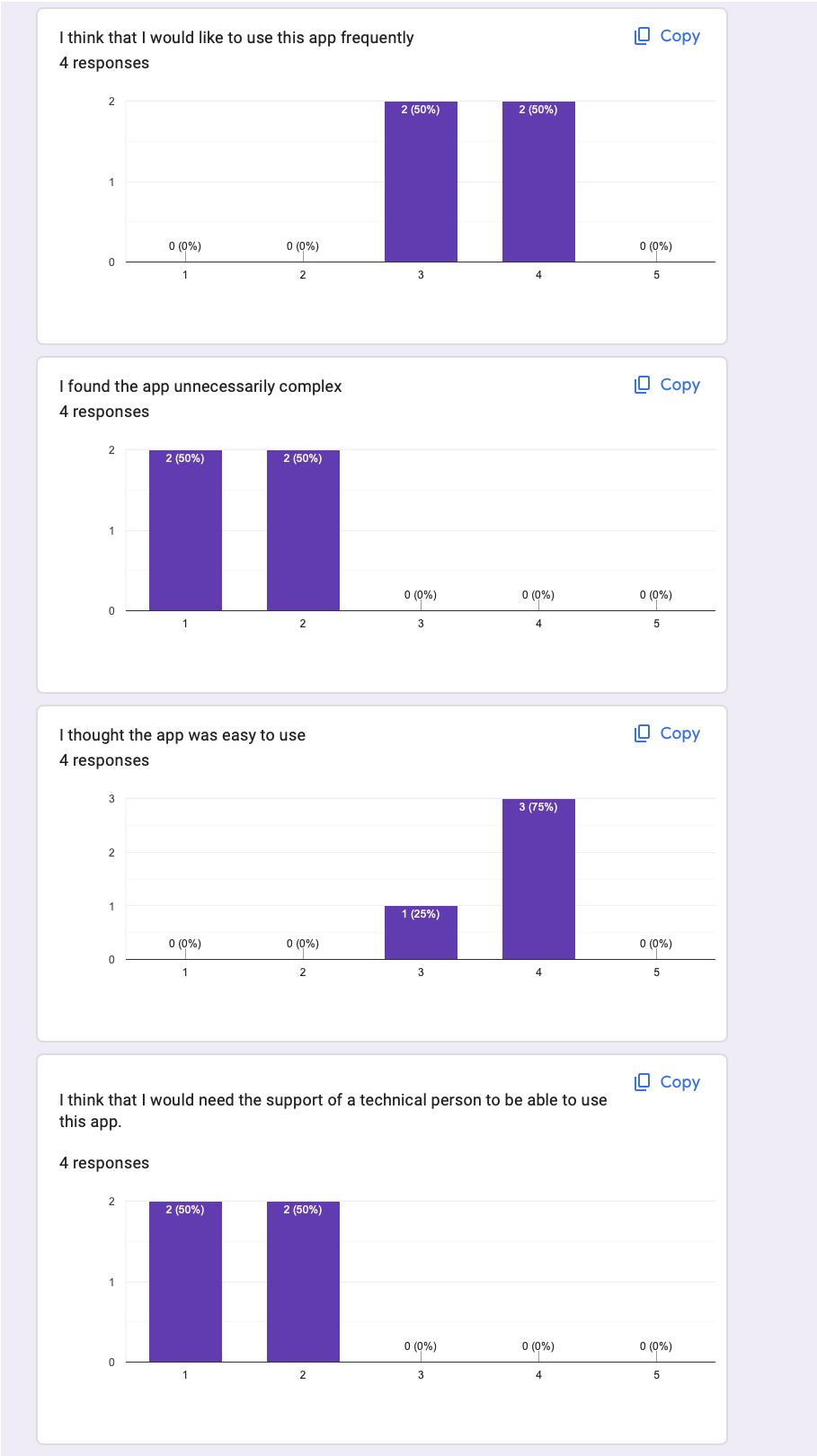


**Figure B.1:** SUS-1

**Figure B.2:** SUS-2

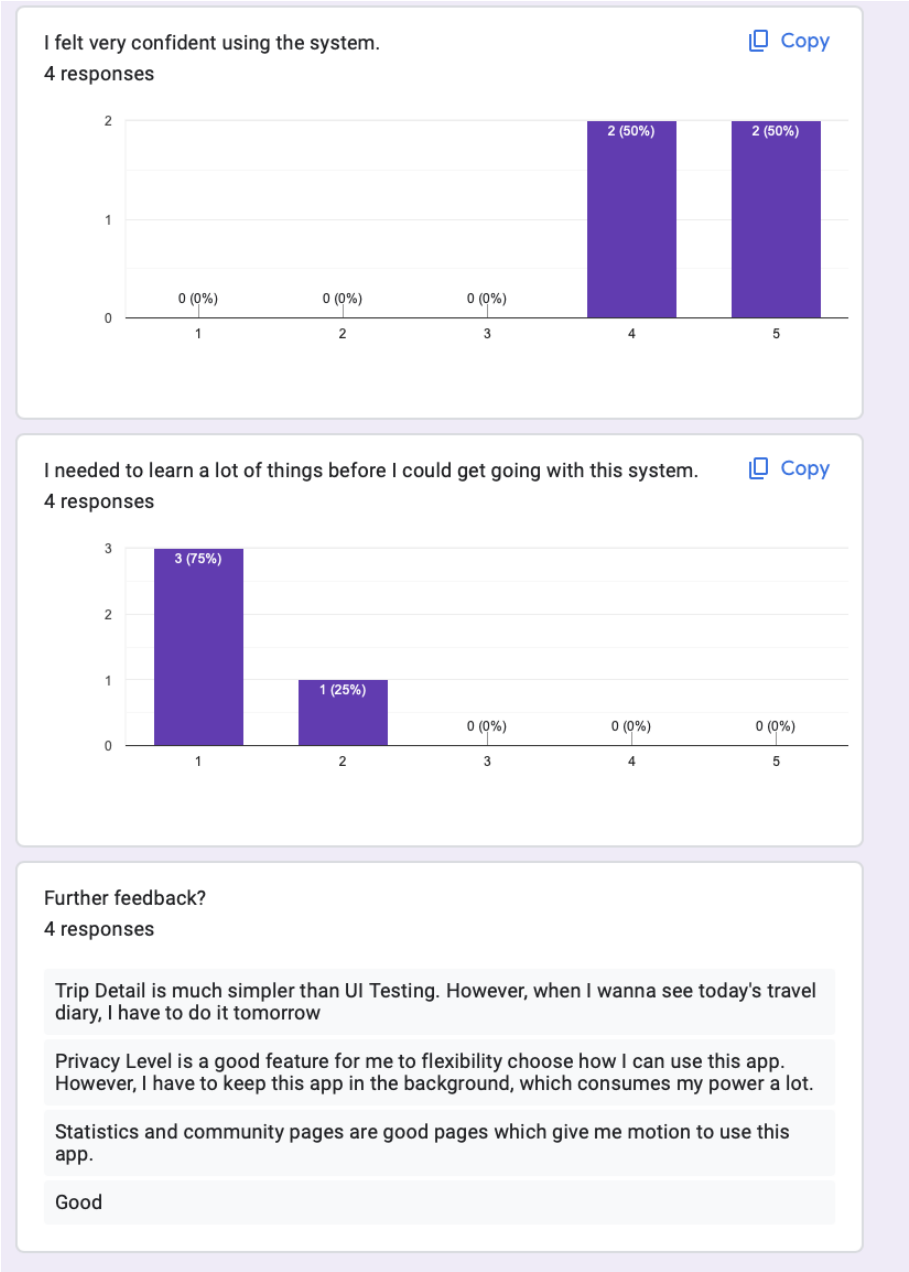


**Figure B.3:** SUS-3