

Project Report
On
Arcane hub - Multiplatform Mobile Uber
Application

Submitted in partial fulfillment of the requirements for the award of degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING



Submitted to:

Dr. Sumedha Arora (E9941)



Submitted by:

Deepam Kumar / 18BCS1522

Saurabh Kumar / 18BCS1517

Deepakshi Sharma / 18BCS1537

Gaurav Sharma / 18BCS1528

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHANDIGARH UNIVERSITY, GHARUAN
June 2021

CERTIFICATE

This is to certify that the work embodied in this Project Report entitled “ ” being submitted by “ ” - UID “ ” , 5th Semester for partial fulfillment of the requirement for the degree of “ **Bachelor of Engineering in Computer Science & Engineering** ” discipline in “ **Chandigarh University** ” during the academic session Jan-Jun 2021 is a record of bona fide piece of work, carried out by student under my supervision and guidance in the “ **Department of Computer Science & Engineering** ”, Chandigarh University.


APPROVED & GUIDED BY:

Dr. Sumedha Arora

DECLARATION

We, student of **Bachelor of Engineering in Computer Science & Engineering, 5th Semester** , session: **Jan – June 2021, Chandigarh University**, hereby declare that the work presented in this Project Report entitled “ ” is the outcome of my own work, is bona fide and correct to the best of my knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any other university or anywhere else for the award of any degree or any professional diploma.

Student details and Signature

| <u>Name</u> | <u>Uid</u> | <u>Signature</u> |
|-------------------------|-------------------|---|
| Deepam Kumar | 18BCS1522 |  |
| Saurabh Kumar | 18BCS1517 | |
| Deepakshi Sharma | 18BCS1537 | |
| Gaurav Sharma | 18BCS1528 | |

APPROVED & GUIDED BY:

Dr. Sumedha Arora

To our parents, teachers and all the well-wishers out there . . .

ABSTRACT

In this Project, we had used the basic idea of UBER app but with a different approach. We used some different technologies this time for the ease and smooth working of our new application. We are building this project in such a way that it should run on both android and ios, so that irrespective of platform, every user can use this application very easily. Also, we are building this project mainly, to help drivers to provide a completely a commission free platform for their earnings. As we are building this project from scratch, it's very much fun and at the same time, we came to know about various new technologies and learnt various new skills.

TABLE OF CONTENTS

| S.NO | TOPIC | PAGE NO |
|------|--|---------|
| | CONTRIBUTORS / CONTRIBUTIONS | 8 |
| | ACKNOWLEDGEMENTS | 9 |
| | List of Figures & Tables | 10 |
| | Glossary | 11 |
| 1 | INTRODUCTION | 13 |
| 2 | BACKGROUND | 13 |
| | 2.1 - MOTIVATION | 13 |
| | 2.2 – SPECIFICATION & GOALS | 14 |
| | 2.3 - SUMMARY | 14 |
| 3 | SYSTEM MODELLING | 15 |
| | 3.1 - PROJECT OVERVIEW | 15 |
| | 3.1.1 - THE RESEARCH STAGE | 15 |
| | 3.1.2 – THE INTERFACE DESIGNING & BUILDING STAGE | 16 |
| | 3.1.2.1 – System Design | 20 |
| | 3.1.3 – The Tuning & Calibration Stage | 25 |
| | 3.1.4 - The Programming Stage | 25 |
| 4 | Experimental Results Discussions | 25 |
| | 4.1 – Discussion Difficulties | 25 |
| 5 | CONCLUSION | 25 |
| 6 | BIBLIOGRAPHY | 26 |

CONTRIBUTORS / CONTRIBUTIONS



Deepam Kumar

Leader (18BCS1522)

- Development of Architecture and framework and system designs
- Front end development (react native)
- Backend development (AWS AMPLIFY)
- Controlling Pull and Push requests of Git.



Saurabh Kumar

18BCS1517

- Front end development (react native)
- Controlling Pull and Push requests of Git.
- Framework Design



Deepakshi Sharma

18BCS1537

- Testing work (Jest)
- Documentation
- UI Design



Gaurav Sharma

18BCS1528

- Documentations
- Testing work(UX)
- Presentations

ACKNOWLEDGEMENTS

I would like to express my special thanks of gratitude to my mentor Dr. Sumedha Arora mam who gave me the golden opportunity to do this wonderful project on the topic (**Arcane hub – Multiplatform Cloud based Uber Application**), which also helped me in doing a lot of Research and i came to know about so many new things I am really thankful to them. Secondly I would also like to thank my friends who helped me a lot in finalizing this project within the limited time frame.

List of Figures

| S.NO | Figure | PAGE NO |
|-------------|--|----------------|
| 1 | 3.1.2.1: Application Home Page | 14 |
| | 3.1.2.2: Application Location Search Page | 15 |
| | 3.1.2.3: Application mode of payment Page | 15 |
| | 3.1.2.4: Application Booking Confirmation Page | 16 |
| | 3.1.2.5: Application Navigation Menu Page | 17 |
| | 3.1.2.1.1: System Design Diagram | 18 |
| | 3.1.2.1.2: supply-demand-node-distribution-system-design | 19 |
| | Figure 3.1.2.1.4: Data Flow Diagram | 22 |
| | | |

Project Description

1. INTRODUCTION

In the present situation, travel and transportation is becoming very popular amongst millions of travelers around the world. It is also one of the biggest demands of many people since it is compulsory for them to travel around the world. We need to travel from one place to another place in our daily life due to many factors.

In the past, it was very difficult to enjoy travels around the world. However in later, the mechanical modes of transportation were introduced and today we see different types of automobiles on the roads, each designed to fulfill precise travel requirements. The trend of private car hire and cabs began soon after the advent of cars and other automobiles in our society. Hiring a private car usually means a car which is not a cab and a driver for managing travel requirements. There are lots of car hire online companies in every part of the world offering private car rental services. Hiring of private car or cab can be done with ease with the help of few clicks of mouse before you are leaving for the journey.

For transportation, it began from animals and till today, it has reached taxis and limousines. People utilize cab services for traveling from one place to another. Aside from public transportation that includes local buses and coaches, taxis are considered as the most convenient and comfortable means of transportation. They give you the ease and convenience of departing from your doorsteps to the exact destination within a short period of time.

We are going to make a Complete Cab Application for both the rider and driver. Also an admin panel from everything can easily be controlled.

2. Background

This section includes the general background information of the framework of our application. This section also covers our goals and specifications for the project undertaken.

2.1 Motivation

We made this application to provide drivers the best application with commission free environment. We tried to implement all the skills which are currently in demand and we can have proper hands on new technologies so we just tried to put our efforts to make users happy.

2.2 Specifications and Goals

The previous work of this already exists. The similar application can be found on the project either Android market. This project will focus on providing high quality usability experiences to users mainly following Google's user interface guideline.

Apart from basic features of Cab Application, we are adding some advance features for both rider and driver applications:-

- Split Payment (For Rider app)
- Interactive map (For both Rider and Driver app)
- Voice Recognition (For both Rider and Driver app)
- Panic Button (For both Rider and Driver safety)
- Heat Map (For Driver app)
- Forward Dispatch (For both Driver app)

In the later stage of development we can add features like:-

- Secret secure chats with messages that self-destruct on both devices within a specified time after being read
- Scheduling messages in advance
- Temporary content like status and stories
- Games (provide more fun)
- Ecommerce features (for example, Chabot's)

2.3 Summary

The aim of this project was to build an application that can be used to help drivers to provide simple application to use with commission free environment. It also open up the possibilities to broaden the understanding and applications of control systems, stabilization, artificial intelligence and computer image processing as it will be upgraded in future stages.

3. System Modelling

This section explains in details how the project was approached. All hardware and software components that will be used will be explained in this section of the report. Problems encountered and solutions to these problems will not be mentioned in this section of the report.

3.1 Project Overview

The ultimate goal of this project is to design a cab application that will provide both driver and user a commission free platform with a very simplistic approach. This application can be used on both android and ios. We are building this project in such a way that it can compete with the current market and can contribute some values to the society.

Agile is a set of techniques to manage software development projects. It consists in:

- Being able to respond to changes and new requirements quickly.
- Teamwork, even with the client.
- Building operating software over extensive documentation.
- Individuals and their interaction over tools.

We believed it was a perfect fit for our project since we did not know most requirements beforehand. By using the Agile, we were able to focus only on the features which had the most priority at the time.

Therefore, it was decided that we are going to use agile method. This project was split into four main stages:-

the research stage, the interface designing and building stage, the tuning and calibration stage, and the programming stage. Each stage will be explained more thoroughly in the following sections of this document.

3.1.1 The Research Stage

The research stage was a critical stage that provided our team with the knowledge necessary to complete the other stages of our project. This stage was an ongoing process that our team had to return to many times during the development process to gain the knowledge needed to continue on with the project. Our research encompassed a wide range of sources, which included studies done at different universities and hobby enthusiast sources. Our research included the system design of android applications, theory and principle of each backend components we are using in this project, and how our application should be modified to fit the purposes of our project to meet the requirements of user in order to compete with other applications in the market.

3.1.2 The interface designing and building stage

This stage concerns the designing part of our Project, which includes designing interface, the framework of ui designs and most importantly making an overview of project and diagrammatically present the project on papers before getting started. This Phase also include the System Design of Our Project, UML diagram, ER diagram, various data flow diagrams used in our project. One thing to notice, this phase taught us a very good understanding on software design part our subject.

While designing interface we used various prototyping components like Figma, Sketch and Balsamiq.

Initially we have decided that we'll download all the small components like icons and design palates from internet but then after few discussion with our team, we thought why not let's make it from complete scratch. So every component of project is made by us,

The interface consist of few pages which are following:-

- Home Page
- Location Search
- Payment mode
- Location Confirmation
- Navigation Menu

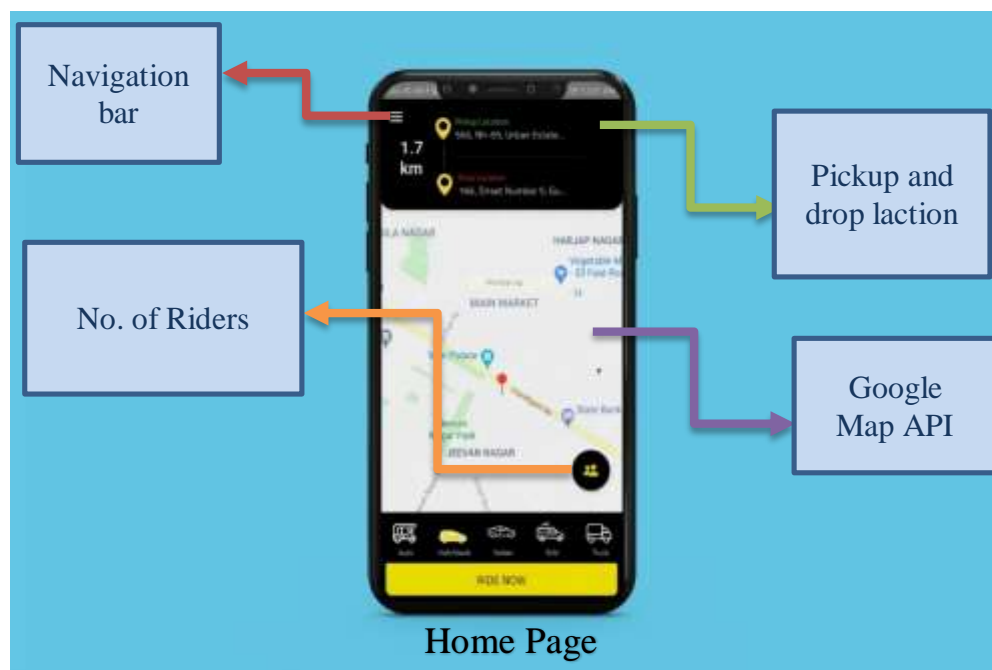


Figure 3.1.2.1: Application Home Page

This page consist of various components like Navigation menu bar (hamburger style) and also provided the vehicle selection option at starting which will help the user to select the riding options even before the ride .

We tried our best to keep the design very simple and aesthetic so that every user can use this application without getting confused. We kept in our mind that the design should not match with the current applications in the market as we used the idea of uber application design.

The next page consist of searching the location or pinning the location for both pickup and drop location. We also used the autocomplete feature which is powered by rankbrain, a key component of google's **machine learning search algorithm** that uses deep learning that helps users to get better results.

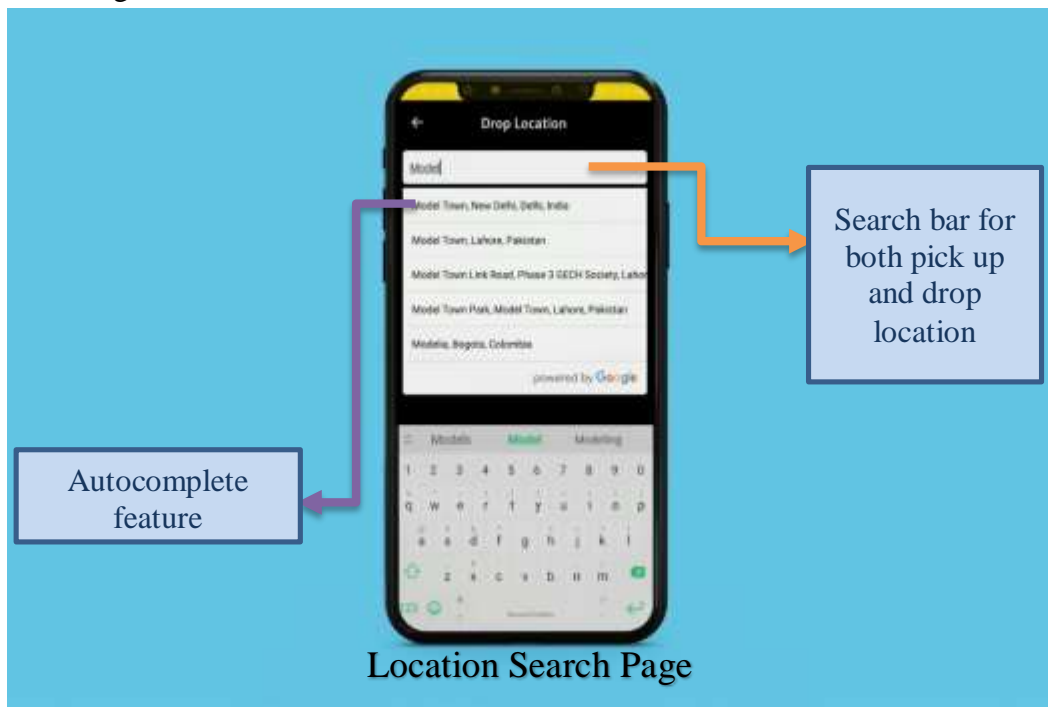


Figure 3.1.2.2: Application Location Search Page

And then we created the mode of payment page for both driver and rider in order to make no conflict between them that whether driver wants online or cash payment method to receive the payment and vice versa for rider.

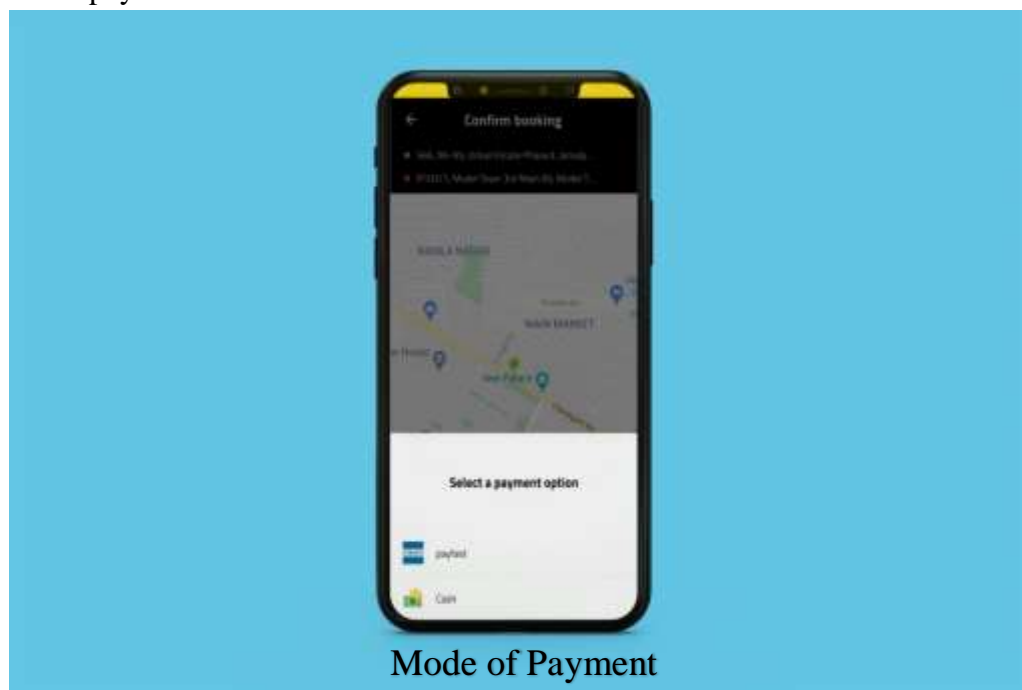


Figure 3.1.2.3: Application mode of payment Page

The next page is booking confirmation page which consist various components like location verification, Fare verification and again the mode of payment option.

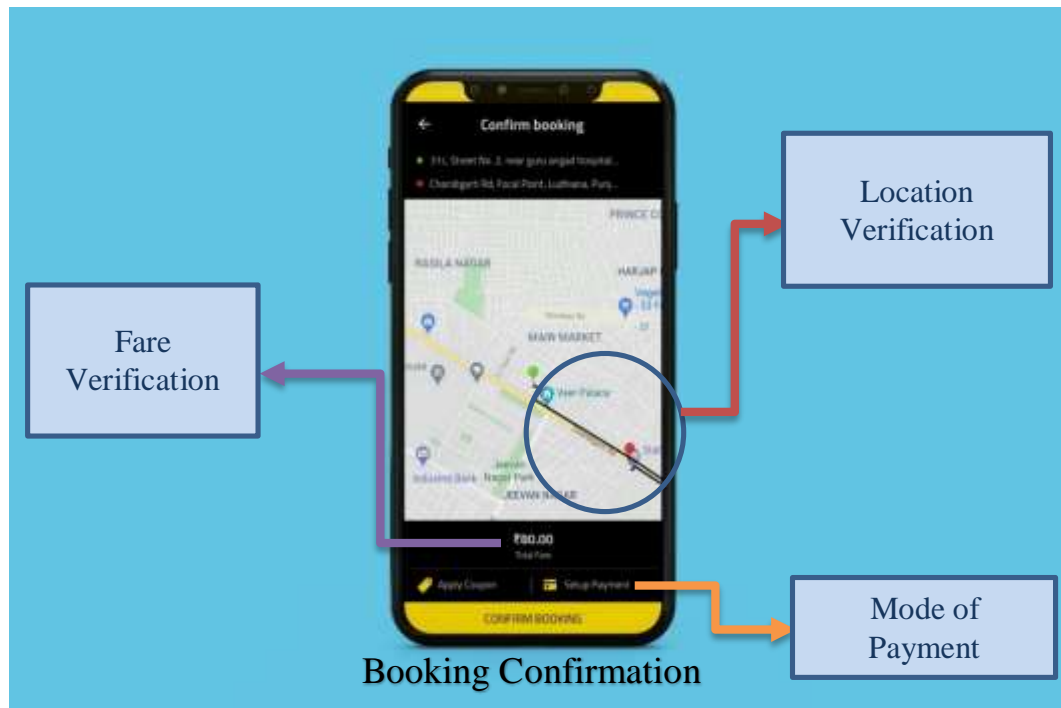


Figure 3.1.2.4: Application Booking Confirmation Page

And finally the navigation menu for user which consist various components like:-

- Balance you have added/ have in your account
- Button to get on Home Page
- Profile Settings
- Wallet to add money to your application account or withdraw money to your bank account
- Notification history
- Rewards
- Refer and Earn Rewards for the growth of application
- SOS Setting for emergency
- FAQ
- Profile Picture and details of user

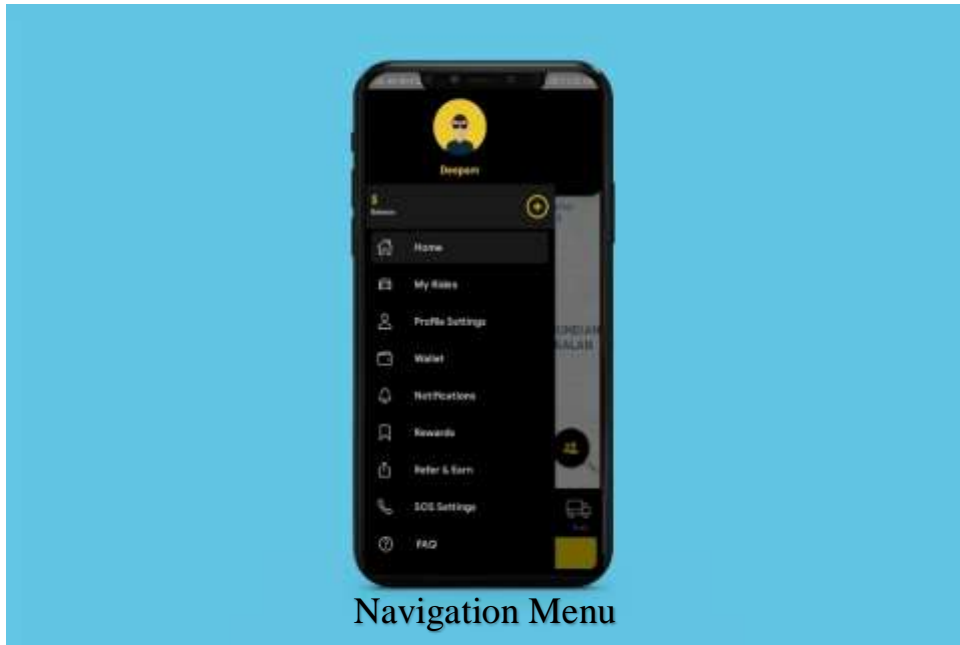


Figure 3.1.2.5: Application Navigation Menu Page

During this stage we focused on verifying and testing each component thoroughly. The testing process will be explained in greater detail in a later section in this document. After each part was verified to be working correctly, we combined the components together and started designing for our application backend framework system designs. This stage is very important as we have to decide that what backend technologies we are going to use in our project. As we have decided that we are not going to copy this project from uber so we thought why not use new technologies this time.

So we decided that we will be building the application from scratch, starting from setting up a React Native project and finishing with connecting the application with a GraphQL backend using AWS Amplify. We will use React Native, an open-source mobile application framework created by Facebook, Inc. It is used to develop applications for Android, iOS, Web.

We will also implement navigation between screens, header bars, and tab bars for the Uber app menu using React Navigation.

For the backend, we will be using AWS Amplify, which is an amazing combination of tools and services from AWS, that helps us build mobile and web applications faster. It offers pre-made authentication components and flows, database, API (REST and GraphQL), storage, and much more.

So now the problem is where to start with, so we begin with designing our system design.

- Earth has a spherical shape so it's difficult to do summarization and approximation by using latitude and longitude. To solve this problem we used **Google S2 library**. This library divides the map data into tiny cells (for example 3km) and gives the unique ID to each cell. This is an easy way to spread data in the distributed system and store it easily.
- S2 library gives the coverage for any given shape easily. Suppose you want to figure out all the supplies available within a 3km radius of a city. Using the S2 libraries you can draw a circle of 3km radius and it will filter out all the cells with IDs lies in that particular circle. This way you can easily match the rider to the driver and you can easily find out the number of cars(supply) available in a particular region.

Supply Service And How it Works?

- In our case cabs are the supply services and it will be tracked by geolocation (latitude and longitude). All the active cabs keep on sending the location to the server once every 4 seconds through a web application firewall and load balancer. The accurate GPS location is sent to the data center through Kafka's Rest APIs once it passes through the load balancer. Here we use **Apache Kafka** as the data hub.
- Once the latest location is updated by Kafka it slowly passes through the respective worker nodes main memory.
- Also a copy of the location (state machine/latest location of cabs) will be sent to the database and to the dispatch optimization to keep the latest location updated.
- We also need to track few more things such as number of seats, presence of a car seat for children, type of vehicle, can a wheelchair be fit, and allocation (for example, a cab may have four seats but two of those are occupied).

How Dispatch System Match the Riders to Drivers?

- We have discussed that DISCO divides the map into tiny cells with a unique ID. This ID is used as a sharing key in DISCO. When supply receives the request from demand the location gets updated using the cell ID as a shard key. These tiny cells' responsibilities will be divided into different servers lies in multiple regions (consistent hashing). For example, we can allocate the responsibility of 12 tiny cells to 6 different servers (2 cells for each server) lies in 6 different regions.

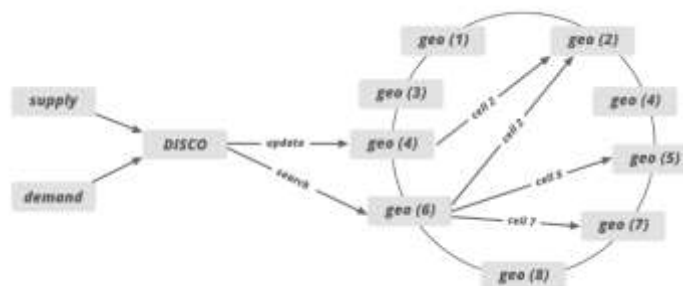


Figure 3.1.2.1.2: supply-demand-node-distribution-system-design

- Supply sends the request to the specific server based on the GPS location data. After that, the system draws the circle and filter out all the nearby cabs which meet the rider's requirement.
- After that, the list of the cab is sent to the ETA to calculate the distance between the rider and the cab, not geographically but by the road system.

- Sorted ETA is then sent back to the supply system to offer it to a driver.

If we need to handle the traffic for the newly added city then we can increase the number of servers and allocate the responsibilities of newly added cities cell IDs to these servers.

How Uber Builds the Map?

Uber uses third party map service provider to build the map in their application. Earlier Uber was using Mapbox services but later Uber switched to Google Maps API to track the location and to calculate ETAs.

1. Trace coverage: Trace coverage spot the missing road segments or incorrect road geometry. Trace coverage calculation is based on two inputs: map data under testing and historic GPS traces of all Uber rides taken over a certain period of time. It covers those GPS traces onto the map, comparing and matching them with road segments. If we find missing road segments (no road is shown) on GPS traces then we take some steps to fix the deficiency.

2. Preferred access (pick-up) point accuracy: We get the pickup point in our application when we book the cab in Uber. Pick-up points are really important metric in Uber especially for large venues such as airports, college campuses, stadiums, factories, or companies. We calculate the distance between the actual location and all the pickup and drop-off points used by drivers.

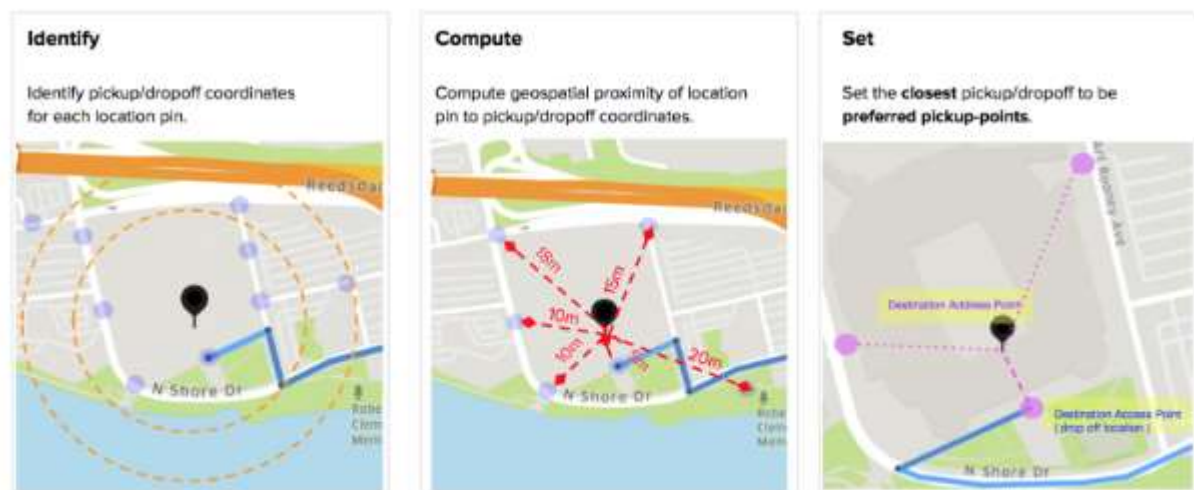


Figure 4: Refining access points is composed of three main steps: identify actual pick-up and drop-off locations used by drivers for a place or address (left), compute the distances from those locations to the place or address (middle), and then set the preferred access point based on the shortest distance (right).

The shortest distance (closest pickup point) is then calculated and we set the pin to that location as a preferred access point on the map. When a rider requests the location indicated by the map pin, the map guides the driver to the preferred access point. The calculation continues with the latest actual pick-up and drop-off locations to ensure the freshness and accuracy of the suggested preferred access points. Uber uses machine learning and different algorithms to figure out the preferred access point.

How ETAs Are Calculated?

ETA is an extremely important metric in Uber because it directly impacts ride-matching and earnings. ETA is calculated based on the road system (not geographically) and there are a lot of factors involved in computing the ETA (like heavy traffic or road construction). When a rider requests a cab from a location the app not only identifies the free/idle cabs but also includes the cabs which are about to finish a ride. It may be possible that one of the cabs which are about to finish the ride is more close to the demand than the cab which is far away from the user. So many uber cars on the road send GPS locations every 4 seconds, so to predict traffic we can use the driver's app's GPS location data.

We can represent the entire road network on a graph to calculate the ETAs. We can use AI simulated algorithms or simple **Dijkstra's algorithm** to find out the best route in this graph. In that graph, nodes represent intersections (available cabs), and edges represent road segments. We represent the road segment distance or the traveling time through the edge weight. We also represent and model some additional factors in our graph such as one-way streets, turn costs, turn restrictions, and speed limits.

Once the data structure is decided we can find the best route using [Dijkstra's search algorithm](#) which is one of the best modern routing algorithms today. For faster performance, we also need to use OSRM (Open Source Routing Machine) which is based on [contraction hierarchies](#). Systems based on contraction hierarchies take just a few milliseconds to compute a route — by preprocessing the routing graph.

Databases

Uber had to consider some of the requirements for the database for a better customer experience. These requirements are...

- The database should be horizontally scalable. You can linearly add capacity by adding more servers.
- It should be able to handle a lot of reads and writes because once every 4-second cabs will be sending the GPS location and that location will be updated in the database.
- The system should never give downtime for any operation. It should be highly available no matter what operation you perform (expanding storage, backup, when new nodes are added, etc).

Earlier Uber was using the RDBMS PostgreSQL database but due to scalability issues uber switched to various databases. Uber uses a NoSQL database (schemaless) built on the top of the MySQL database.

- Redis for both caching and queuing. Some are behind [Twemproxy](#) (provides scalability of the caching layer). Some are behind a custom clustering system.
- Uber uses schemaless (built in-house on top of MySQL), Riak, and Cassandra. Schemaless is for long-term data storage. Riak and Cassandra meet high-availability, low-latency demands.
- MySQL database.
- Uber is building their own distributed column store that's orchestrating a bunch of MySQL instances.

Analytics

To optimize the system, to minimize the cost of the operation and for better customer experience uber does log collection and analysis. Uber uses different tools and frameworks for analytics. For log analysis, Uber uses multiple Kafka clusters. Kafka takes historical data along with real-time data. Data is archived into Hadoop before it expires from Kafka. The data is also indexed into an Elastic search stack for searching and visualizations. Elastic search do some log analysis using Kibana/Graphana. Some of the analyses performed by Uber using different tools and frameworks are...

- Track HTTP APIs
- Manage profile
- Collect feedback and ratings
- Promotion and coupons etc
- Fraud detection
- Payment fraud
- Incentive abuse by driver
- Compromised accounts by hackers. Uber uses historical data of the customer and some machine learning technique to tackle with this problem.

How To Handle The Datacenter Failure?

Datacenter failure doesn't happen very often but we still maintains a backup data center to run the trip smoothly. This data center includes all the components but Uber never copies the existing data into the backup datacenter.

It actually uses driver phones as a source of trip data to tackle the problem of data center failure.

When The driver's phone app communicates with the dispatch system or the API call is happening between them, the dispatch system sends the encrypted state digest (to keep track of the latest information/data) to the driver's phone app. Every time this state digest will be received by the driver's phone app. In case of a datacenter failure, backup data center (backup DISCO) doesn't know anything about the trip so it will ask for the state digest from the driver's phone app and it will update itself with the state digest information received by the driver's phone app.

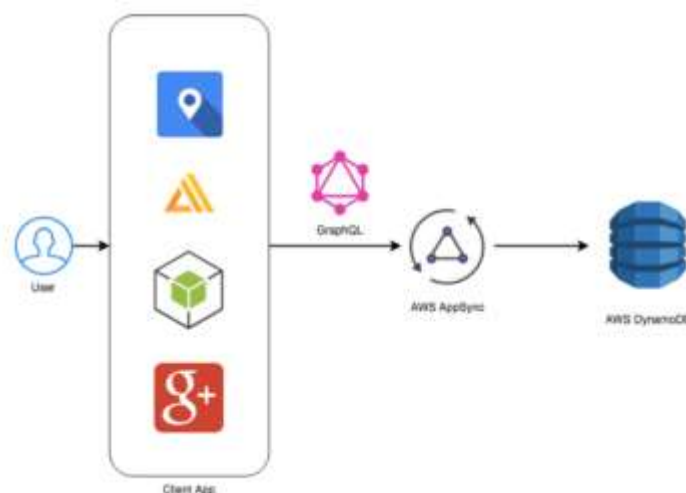


Figure 3.1.2.1.4: Data Flow Diagram

The next step was to tune and calibrate our application.

3.1.3 The Tuning/Calibration Stage

During this stage the application control system and design was tuned. This was a tedious process. After some initial tuning, the application was ready to go. The Jest testing process and results of the test will be explained in greater detail later section in this document.

3.1.4 The Programming Stage

In this stage of our project is moved to further stage i.e. coding stage. The preliminary designs for these commands have been drawn out and will be explained in greater detail in a later section of this document. Unfortunately we did not get very far in this stage and programming for these commands has yet to be started.

4. Experimental Results and Discussion

In this section of the document we will be discussing the verification and testing of each hard-ware and software component. All problems will be described in detail and the solutions we made to solve these problems. In this section we will also discuss our overall results of the project and what we could have done to improve upon our project. Future work for this project will also be mentioned in this section of the document.

4.3 Discussion Difficulties

We are facing one difficulty that by default the application starts taking the location permission even the user close it.

So this is one of the privacy issue we are resolving the issue.

Conclusion

This is very simple app and in reach of everyone. So that anyone from anywhere can use it. Drivers are partners, they are not our employees. The whole application is developed considering both rider's and driver's safety. This app will help to find the necessary services within a short period of time and with few efforts spent.

Bibliography

Cheok, D. (2015). The current state of Uber's global challenges, mapped. Bloomberg Business. Web.

Doole, I. & Lowe, R. (2008). International marketing strategy: Analysis, development and implementation. London, UK: Cengage Learning EMEA.

Higson, C. (2015). The value of Uber. Forbes. Web.

McAlone, N. (2015). Here's how Uber got its start and grew to become the most valuable startup in the world. Business Insider. Web.

Shontell, A. (2015). Uber CEO explains his company's highly ambitious goal to end car ownership in the world. Business Insider. Web.