# DRIVER SOURCING PORTAL (UBER)

*Project report submitted in partial fulfilment of the requirement for the degree of*

## BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING


By

Shivam Dhawan (151455)

UNDER THE SUPERVISION OF

**Mr. Sumeet Kumar**




Department of Computer Science & Engineering and Information Technology

**Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh**

# CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in this project work entitled "Driving Sourcing Portal (Uber)" in partial fulfillment of the requirements for the award of the degree of B.Tech in IT, JUIT, Waknaghat is an authentic record of my own work carried out during the period February to May 2019 under the supervision and guidance of Mr Sumeet Kumar.

I have not submitted the matter embodied in this project work anywhere for the award of any degree or diploma

**Shivam Dhawan**

# ACKNOWLEDGEMENT

I would like to make use of this opportunity to give my deepest and most sincere thanks to our faculty and the team of management for providing me with their guidance, knowledge and wisdom. The guidance and facilities provided to me at the institution has been of great value and has greatly helped in my career.

I would also like to thank **Prof. Dr. Satya Prakash Ghrera, FBCS, SMIEEE** Professor, Brig (**Retd.) & Head of Department of CSE and IT** , Department of Computer science and Engineering at Jaypee University of Information Technology, for allowing us to include the information in out reports.

I would also like to show my sincere appreciation to **Grappus Technologies Pvt. Ltd.** for furnishing me with a chance to seek after my modern preparing, as it is a significant piece of the B .Tech course and the one opens you to the business benchmarks and influences you to adjust to the most recent patterns and advancements. In the meantime, it gives an encounter of taking a shot at a venture.

I also would like to show my deepest appreciation to **Mr. Sumeet Kumar** for his motivation, helpful recommendation, plan investigation and friendly direction in my work, without which this task work culmination would have been unthinkable for me. Genuine gratitude to one and all of my seniors and associates and colleagues at Grappus for their help through the venture.

**Shivam Dhawan**

# ABSTRACT

**Uber** wants us to develop a platform where it can provide service to DTI (Driving Training Institute) and Driver to register themselves with **Uber** and start working as their employee.

Goal is to design, develop and build a low cost driver lead channel or portal where an individual driver or driver training schools can register and upload leads for Uber. This would allow them to simplify their process of onboarding and attract more partners to connect with them. It would also reduce their cost of operations that previously involved training of representatives who would interact and help onboard their partner drivers or connect with training agencies.

# LIST OF FIGURES

# LIST OF TABLES



# LIST OF ABBREVIATIONS

| Abbreviation | Description |
|---|---|
| ERD | Entity Relationship Diagram |

| | |
|---|---|
| H/w | Hardware |
| S/w | Software |
| GHz | Giga Hertz |
| MB | Megabyte |
| GB | Gigabyte |
| RAM | Random Access Memory |
| API | Application Programming Interface |
| ER | Entity Relationship |
| MVC | Model View Controller |
| HTTP | Hyper Text Transfer Protocol |
| SQL | Structured Query Language |
| AJAX | Asynchronous JavaScript And XML |
| URL | Uniform Resource Locator |
| UI | User Interface |
| DFD | Data Flow Diagram |
| DTI | Driver Training Institute |
| DSP | Driver Sourcing Panel |

# C O N T E N T S

**CHAPTER 4: TESTING**

## CHAPTER 5: CONCLUSION AND REFERENCES



## CHAPTER 6: ANNEXURES

# CHAPTER 1
# INTRODUCTION

1.1. Company’s Introduction
1.2. Problem Statement behind the project
1.3. Solution proposed through the project
1.4. Deliverables

## 1.1 Company's Introduction

Grappus Technologies Pvt Ltd. was set up in 2013 by Dhruv Goel and Anuj Birla with the purpose of taking App Development for mobiles and the general customer experience of a thing to another measurement in India. Maintained by Ex-IITians and NSIT-ians, Grappus has worked with various huge brands, for instance, Mercedes-Benz, PVR Cinemas, Star India (constrained by Tata), Uber, Aon Hewitt and that is only the start. Besides, Grappus has been starting late featured in Top 50 Mobile App Companies in India.

Since the start, Grappus has gone for structure items that handle true issues, however with style. Each task that Grappus has dealt with has a specific plan embodiment and client experience that keeps the customers upbeat and makes them return for additional.

With 46 workers, Grappus has a committed group for each undertaking and each group performs multiple tasks on different activities. Aside from undertaking group, there are groups dependent on the piece of the task that they take a shot at, for example, Front-End Team, Backend Team, iOS Team, Android Team, Testing Team and Design Engineering Team. Grappus centers around Designing and Development of items, however it likewise gives specific administrations dependent on Testing, Estimation, Maintenance and Research.

For each undertaking, the whole group chose is given a legitimate instructions about the prerequisites of the customers and the courses of events all together. Grappus has confidence in conveying quicker, on time applications with better quality than the customers, subsequently supporting the significance of conveying dashes.

## 1.2 Problem Statement behind the project

A DTI in Sholapur is going to graduate a group of 40 drivers and every one of them are scanning for employments with their recently discovered ability. The DTI needs to add to that as employment ensure is a main consideration in verifying future clusters for the

establishment. DTI needs to send every one of the drivers data to Uber and check whether we can help in finding the driver a line of work.

A jobless individual with a driving permit is looking for employments and might want to drive on Uber full time or low maintenance. Since he doesn't have a vehicle, he is searching for different approaches to drive on Uber under another person's vehicle.

### 1.3 Solution proposed through the project

We want to design and develop a portal where individual drivers or driver training schools can register and upload driver leads for Uber.

Goals

- Build a low-cost driver leads channel
- To develop a scalable solution and to reach the drivers & DTIs in hinterlands

### 1.4 Deliverables

Working as part of the backend team in the given project I was responsible for adding various features to the product based on the updated scope and requirements of project and client. Besides that I was working mostly to optimise the existing code by examining the code by means of various tools and optimising it by means of updating queries, reusing extracted data and fixing piece of code that was previously delivered.

| S. No. | Phase | Deliverables |
|---|---|---|
| **1.** | Requirement Analysis | Use Case |
| **2.** | System Design | Database Schema |
| **3.** | System Design | Data Flow Diagrams (Level 0 and Level 1) |
| **4.** | Construction and | Test Cases |
| **5.** | Testing Unit Testing | Test Results |

**Table 1.1: Deliverables**

**Amazon Web Services (AWS) Server:** This deliverable was dealt with utilizing Amazon Web Services (AWS). The task has been deteriorated into 5 reliant small scale benefits, each facilitated on an Amazon Web Services (AWS) EC2 Instance. It additionally included getting versatile IP Addresses for every one of these servers and enabling them to associate with one another.

**Database Setup:** The task utilizes NoSQL controlled MongoDB that permits adaptable capacity of information in an application because of nonattendance of connections and quicker ordering of expansive unstructured information. The database is facilitated on Amazon Web Services' (AWS) EC2 case that must be gotten to by any of the 5 Micro Services in the undertaking.

**Firebase:** Firebase by Google is a cloud based portable and web application improvement stage. It gives vast number of administrations utilized in the task including Crashalytics and Firebase Cloud Messaging for Push Notifications.

**API Deliverables:** Backend Development of this undertaking has been actualized utilizing Django for Python, NodeJS with ExpressJS and MongoDB. A sum of 121 API Endpoints

have been conveyed and more are in the improvement procedure. Similar API Code base is utilized for a Multi-Functional Admin Panel/CMS and for the two iOS and Android Mobile Applications.

**Redis Caching Server:** Since the measure of information the application will manage is expansive and the reaction time should have been limited, Redis Caching Server have been set up with the backend to guarantee that the API has quicker information get to and the task works consistently.

**MSG91 SMS Service:** MSG91 is a SMS Messaging administration that can be fused with the backend APIs and can be utilized for sending SMS to the clients of the undertaking. This administration gives uncommon OTP Requesting and Verification tasks with IVR Call office for checking the legitimacy of the clients. This administration is additionally utilized in this venture for sending SMS Notifications to enlisted clients.

**AWS S3 Service:** Amazon Web Services (AWS) gives distributed storage administration called S3 that is worked to store and recover any measure of information from anyplace. The undertaking gives an API Endpoint where the client can transfer numerous documents utilizing a solitary endpoint and accordingly, would get a variety of URLs produced by S3 in the wake of transferring the records on mists stockpiling.

**AWS SES Service:** Amazon Web Services (AWS) Simple Email Service (SES) is a cloud email administration which is utilized for sending messages for the clients of use/CMS. The SES API is utilized for sending email warnings to the enrolled clients of the application and is likewise utilized for Email Verification of the enlisted client.

# CHAPTER 2
# PROJECT DESCRIPTION

## 2.1 Interface for the System

The Driver Sourcing Portal has the following interfaces designed for the Users' interaction:

1. **Signup Screen**
    a. In this Screen, the User will be given option to signup either as DTI or Driver.
    b. Signing up as DTI, user has to provide necessary detail and confirm his mobile number.
    c. Signing up as Driver, user has to provide necessary detail and confirm his mobile number.
2. **Login Screen**
    a. This interface will allow users onboarded on the system to provide their credentials to system.
    b. An OTP verification option has also been provided to users to authenticate themselves.
    c. User will either get redirected to dashboard screen or shown an error based on the submitted data.
3. **Driver Dashboard Screen**
    a. In this screen, a *Driver* can view its profile and application status.
    b. In this screen, a *Driver* can choose to edit its profile.
4. **DTI Dashboard Screen**
    a. In this screen, DTI will be able to view its profile and application status of lead uploaded by him.
    b. In this screen, DTI can choose to edit its profile or select any of lead to update its profile.
    c. In this screen, DTI can filter lead by their status.
5. **DTI Add Lead Screen**
    a. In this screen, the DTI can add new lead.

b. After creating a lead, DTI can review its profile and update to Uber portal.

**6. Lead Status Screen**

a. In this screen, the Lead can check its status.

**7. Admin Dashboard Screen**

a. In this screen, admin can view all the DTI registered with Uber.

b. In this screen, admin can filter DTI based on their annual strength, class strength and status.

c. In this screen, when admin click on any DTI, it will take admin to DTI profile where it can see leads referred by that DTI.

**8. Admin Lead List Screen**

a. In this screen, admin can view list of all leads referred on portal.

b. In this screen, admin can filter lead based on their location, status, referred by DTI.

c. On clicking on any lead will take admin to lead profile.

## 2.2 Specifications for the system

### 2.2.1 Requirements - Hardware

For development, each server instance used for hosting the project is an AWS Instance Type T2-Medium and has the following hardware specifications:

- Architecture – X86 or X86/64-bit h/w based architecture – Intel Xeon processor is provided with compatible Motherboards
- Assembled with power of 3.15-megahertz (MHz) processor or higher
- Memory should at least be of 4 gigabytes (GB) of RAM.
- Secondary Storage be provided of at least 30 gigabytes (GB) of space on HD besides OS files.

Users need to have atleast the following configuration.

- Architecture of X86 or X86/64-bit h/w type architecture
- Assembled with 1-gigahertz (GHz) type processor or higher
- Memory should be of at least 1 gigabytes (GB) of RAM.
- Secondary Storage be provided of at least 150 megabytes (MB) of space on HD besides OS files.

### 2.2.2 Requirements - Software

Driver Sourcing Panel uses the following software requirements **For development**:

- Python 3.6.4
- NPM
- Node JS
- PostgreSQL 10
- Postman
- PyCharm

Users need to have the following kind of s/w configuration:

- Android 6.0 (Marshmallow) or IOS 8 or higher
- Web Browser with an active internet connection.

## 2.3 Behind The Project - Methodologies and used Tools

### Django (v2.0.0) and REST APIs

Django is a free and open source web application system written in Python. It is in all respects broadly utilized just for the sake of it's adaptability and information situated Web Applications and APIs as indicated by the REST architecture design. Serene web administrations are worked to work best on the internet. Authentic State Transfer (REST) is an engineering style that helps determine the requirements, for example, an interface that

is uniform, however connected to the website's administration may prompt attractive properties like execution, versatility, and modifiability, that empower developers to build fast and efficient products.

***Django Rest Framework***

It is a framework that provides easy to use interface over django and makes your work extensively simple and yet it is completely flexible. It offers an appealing, web based rendition of your API, and the alternative of returning crude JSON. It gives incredible model serialization, show information utilizing standard capacity based perspectives, or get granular with ground-breaking class-based perspectives for increasingly complex usefulness.

***Postgres (v10)***

It is amazing, open source, object-social database framework that provides utilization and broadening of the SQL language combined with multiple highlights that securely store and scale extensively most of the confounded information and outstanding burdens.

It provides access to numerous highlights intended to enable designers to assemble applications, executives to ensure information uprightness and manufacture shortcoming tolerant conditions, and help you deal with your information regardless of how huge or little the dataset. Notwithstanding being free and open source, it is exceedingly extensible.

***React JavaScript (v16.3.2)***

Respond is a JavaScript library that makes use of structured UI designed explicitly for SPAs. It has been utilized for taking care of the view layer of the MVC for web and portable applications. It likewise enables us to make reusable UI parts.

It allows developers to create simple and module based components that can be easily plugged in into one another and combined together to create a well-structured, fully scalable and fast UI. It's main performance factor is it's ability to make use of virtual DOM to render only the parts that need to be changes and not everything.

***Lifecycle Model***

Dexterous type programming makes way for improvement and alludes to a gathering of programming advancement strategies dependent on reiteration based improvement, where necessities and settings may proceed through coordinated effort between self-sorting out cross functional groups. Deft techniques or Agile proceedings may lead to developments in a restrained undertaking the executives procedure that empowers visit investigation and adjustment, an administration reasoning that energizes collaboration, self-organization & responsibility, a lot of creating of prescribed procedures planned to take into account the rapid conveyance of fantastic programming, and organisational procedure that leads to improvements in client demands and friends objectives.

Scrum is a subset of Agile. Scrum is done every now and then and used to direct complex problems headway, using iterative and enduring practices. Scrum basically assembles the team, helps in brainstorming, improves proficiency and reduces time to benefits in regard to extraordinary "course" shapes. Scrum shapes enable relationship to change effectively to rapidly advancing necessities, and produce a thing that meets creating business goals.

**2.3.1 Phase 1 - Requirement**

The Requirements stage centers around what the framework will do in an exertion that sees all partners, including supporters and potential clients, as significant wellsprings of data.

- **Use Case model**: Use case models allows us to take into account how users will interact with your framework. An use case chart in its simplest is a portrayal of a client's interaction with your framework & delineating the determinations of the diagram. It's

outline can depict various kinds of clients of a framework and the different ways that they wish to collaborate with your framework.

- **Initial Domain Model:** A domain model recognizes principal business element types and the their inter-connections. A general E.R diagram can also be created to show relations of an Entity with others.
- **Website Structure Model:** This diagram is designed to show the overall structure of the systems and list their sitemap depicting how website flow is structured.

Necessities have characterized for first forms which incorporates a framework which can get Users in the framework, client level and school level collaborations. The necessities have been sorted beneath under Functional, Usability, Reliability prerequisites.

***Functional Requirements:***

- Creating models for database and linking them with appropriate views and controllers.
- Configure and maintain servers to host the system.
- Brainstorming the flow of the system and creating relevant API for user's login, signup and profile view.
- Determining the structure and working for APIs of DTI CRUD
- Determining the structure and working for APIs of Driver CRUD
- Determining the structure and working for APIs of Lead CRUD
- Determining the structure and working for user's auth
- Determining the structure and working for APIs of Notification CRUD and it's delivery by means of Push notifications, Phone SMS and Electronic mail mechanisms.
- Determining the structure and working for APIs for Bulk CRUD, sms sending, sms listing and Push Notifications for each sent sms.

***Non-Functional Requirements:***

Just as practical prerequisites which are characterize explicit conduct or elements of our framework we can distinguish non-utilitarian necessities that can be made into use to

deliver views on processing of our system. By and large, we can recognize nonfunctional prerequisite of our framework DSP under the following classes:

- Smooth UI functionality
- Safety of our data
- Continuous readiness of our system
- Un-interruptibility
- Escalated system in terms of performance

***Usability Requirements***

They generally help us determine the validity of the data provided to us by end-user on the internet. They also help us determine auth of users communicating with our platform.

***Reliability Requirements***

Communications by end-user are saved over our framework in form of various states of the system.

### 2.3.2 Phase 2 - Design

Setup organize begins once basics finished after the establishment of basic run. Structured orchestrate endeavors to reveal different substances attracted with the framework and the nearest directed about what's more interactable for our framework.

The plan arrange takes its very basic of data as the prerequisites perceived towards confirmed necessities record. For every fundamental, a lot of at any rate one course of action fragments will be passed on in light of social affairs, workshops, similarly as model endeavors.

The portrayal of the ideal framework joins into detail, and everything considered wire accommodating chain of essentialness outlines, screen position diagrams, grid based presentation of our business, process plots, simple codes, and a flat out db appearing of the

segments included. DB Diagrams and DFDs for the structure are made to value the through and through stream and comfort of the framework. This stage also included seeing the on-screen characters of our framework.

**Actors involved in the system:**

- **Super Admin -** It is basically the possessor of the architecture. They have complete ownership of the system and may perform any kind of interaction with the contexts that belongs to our environment. They may also create new context for the same.
- **Uber Staff** - The **Uber Staff** is one which is made by the higher level SA and given only limited access of the environment. A Uber staff can create another User (DTI, Driver, Lead). They have permission to change Status of lead, driver or DTI.
- **DTI –** DTI or Driver Training Institute has right to create and verify detail of driver or lead before uploading them to DSP. DTI can also upload Leads in bulk via CSV.
- **Driver –** Driver can also register himself individually by registering and uploading his detail on DSP.
- **Lead –** Lead are Driver whose detail are uploaded by DTI. A lead can check his status of approval as driver in Uber.

### 2.3.3 Phase 3 - Development

Based on a triple tier architectures with lower-up way of working, it is structured in the following way:

#### 2.3.3.1 Presentation Layer

It contains arrangement of signing into & investigating client's based specific board. The client may submit any kind of comparing data on forms and the subtleties after that moved over to JSON-based arrangement to structures backend frameworks for further processing. Essentially, they comprises of segments designed to give a typical extension of center rationale embodied to be part of the layer below.

#### 2.3.3.2 Business Logical Layer

The solicitation developed for website page is passed to backend servers & control passed over to a separate strategy as indicated by the courses been given. The strategies contain the business rationale with respect to the login check, JWT token verification, DTI creation, Driver creation and different exercises.

#### 2.3.3.3 Access Layer

Received information is stored into PostgreSQL database by means of Django ORM.

### 2.3.4 Phase 4 - Implementation

This is executed on AWS EC2 system. The Database is kept up under RDS occasion utilizing PostgreSQL. Client Interacts by means of an internet Browser that sends HTTP solicitation to the server which answers back in the wake of handling the mentioned page. Along these lines, the framework is actualized as a Client server model. To get to the information, the server associated with the server which answers with mentioned Data Sets.

### 2.3.5 Phase 5 - Testing

DSP is made along with parallel testing forms. So as to build unwavering quality & to decrease expansive influence on system, any kind of testing isn't restricted to the way toward running of the system with the expectation of finding out some kind of bug (mistakes or different deformities). It’s centered around the assignment of interface alongside the exactness and solidness of server interaction. Programming testing is expressed as the way toward approving & confirming that a PC does:

- Is upto the expectation of minimum h/w and s/w
- Performs the required tasks as expected
- Can replicate the expected behaviour of our machines
- Works in expectancy with the client

# CHAPTER 3
# FUNCTIONALITY

3.1 DB Design based on logical structure

3.1.1 ERD

3.1.2 System’s Grid-based diagrams

3.2 I/O Designs

3.3 Exploring use cases

3.3.1 Sign Up

3.3.2 Sign In

3.3.3 Edit Profile

3.3.4 Manage Leads

3.3.5 Manage DTI

### 3.1 DB Design based on logical structure

A DB design is the skeleton structure that addresses the savvy point of view all in all dbs. It describes how the data is formed and how the data related to each other is accessed. It designs all of the confinements that are to be associated on the data.

### 3.1.1 ERD

It demonstrates the connections of substance sets put away in a db. A substance in this setting is a segment of information. As such, ER outlines show the legitimate structure of dbs.

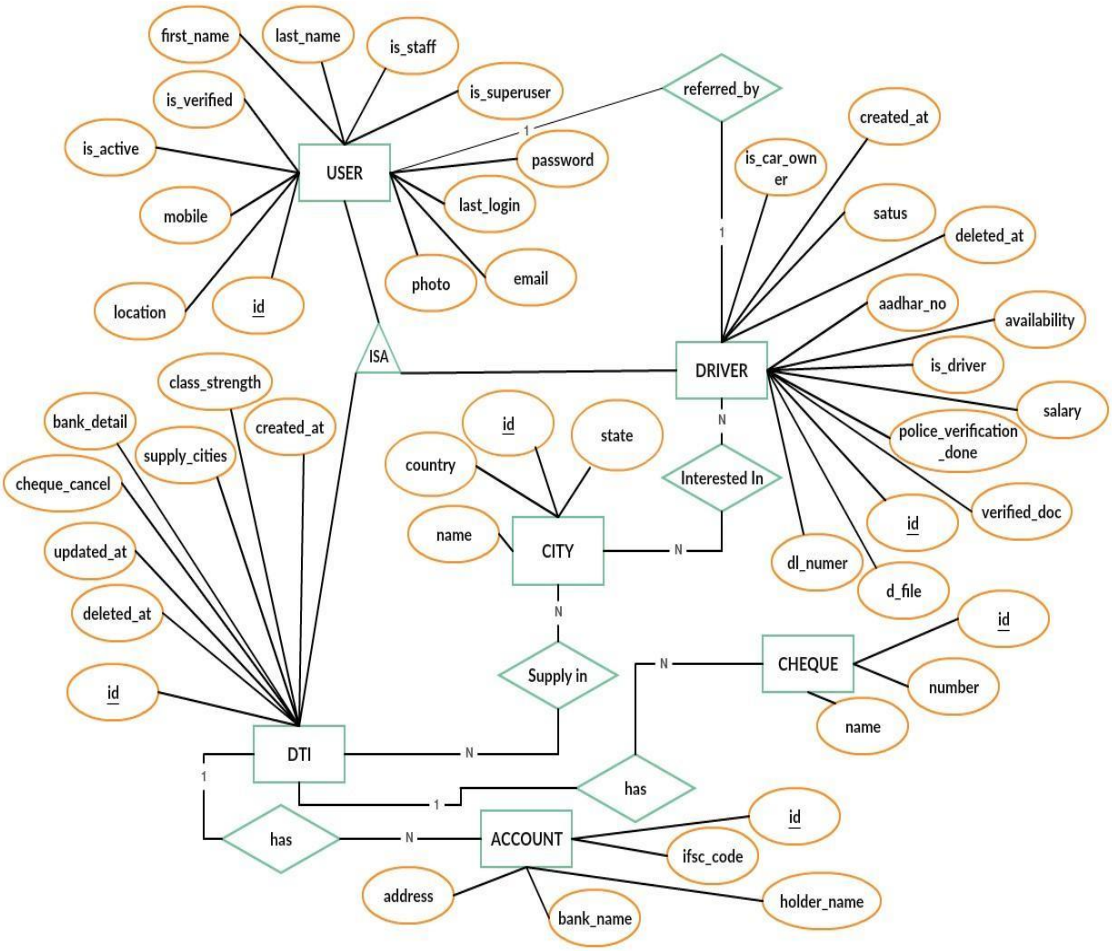


**Figure 3.1: E-R Diagram**

### 3.1.2 System's Grid-based diagrams

**1. User Table**

| Column | Type | Description |
|---|---|---|
| Id | Uuid | Primary key |
| email | varchar(50) | Electronic mail Id of user |
| password | varchar(32) | User's login password |
| first_name | varchar(30) | First name of user |
| last_name | varchar(30) | Last name of user |
| mobile | varchar(13) | Mobile number of user |
| location | varchar(40) | Hometown location of user |
| photo | varchar(256) | Profile image url of user |
| is_verified | Boolean | Determine if user is a verified user or not |
| is_active | Boolean | Determine if user's account is active or not |
| is_staff | Boolean | Determine if user is admin or not |
| is_superuser | Boolean | Determine if user is superuser or not |
| last_login | timestamp | Last login of user |

**Table 3.1: User Table**

**2. Driver Table**

| Column | Type | Description |
|---|---|---|
| Id | uuid | Primary key |
| user_id | uuid | One to One relation key to user table |
| referred_by | uuid | Foreign key to user table |

| | | |
|---|---|---|
| dl_file | varchar(256) | Url to driver license file |
| dl_number | varchar(20) | Driving licence number |
| verified_doc | varchar(256) | Url of verified document uploaded |
| police_verification_done | boolean | Holds a flag for document verification |
| Salary | decimal | Current salary if any |
| Availability | int | For how man hour driver is available |
| aadhar_no | varchar(16) | Aadhar number of driver |
| Status | varchar(12) | Status of user, accepted/rejected |
| is_car_owner | boolean | If user have a car or not |
| created_at | timestamp | When driver was created |
| deleted_at | timestamp | When driver was deleted |

**Table 3.2: Driver Table 3.**

**DTI Table**

| **Column** | **Type** | **Description** |
|---|---|---|
| Id | uuid | Primary key |
| user_id | uuid | One to One relation key to user table |
| supply_cities | text[] | Array containing id of cities |
| class_strength | int | Student graduating from DTI |
| created_at | timestamp | When DTI was created |
| deleted_at | timestamp | When DTI was deleted |

**Table 3.3: DTI Table**

**4. Account Table**

| Column | Type | Description |
|---|---|---|
| Id | uuid | Primary key |
| user_id | uuid | Foreign key to User table |
| Name | varchar(60) | Bank name |
| ifsc_code | varchar(20) | Bank IFSC code |
| Address | varchar(256) | Address of bank |

**Table 3.4: Account Table**

**5. Cheque Table**

| Column | Type | Description |
|---|---|---|
| Id | uuid | Primary key |
| user_id | uuid | Foreign key to User table |
| bank_name | varchar(60) | Bank name |
| Number | int | Cheque number |

**Table 3.5: Cheque Table**

## 3.2 Input Output Design

**1. Sign up Screen**

| | |
|---|---|
| **Purpose** | · This interface allows the User to Sign up to the system. |
| **Description of Each Field** | · Mobile Number: The user enters her/his number to register with the system.<br>· OTP: User enter the OTP received.<br>· Enter Secondary detail like name, email, etc.<br>· Sign Up Button: Register the user with primary and secondary details. |

| | |
|---|---|
| **Validation Checks** | · Number must not be registered in the database.<br>· Entered OTP should be correct. |

**Table 3.6: Sign up Screen**

**2. Login Screen**

| | |
|---|---|
| **Purpose** | · This interface allows the User to login to the system.<br>· Provides Token based authentication.<br>· Performs the authentication and authorization. |
| **Description of Each Field** | · Mobile Number: The user enters her/his number to log into the system.<br>· OTP: User enter the OTP received.<br>· Log In Button: Logs the user in or displays an error on mismatch of mobile number or OTP. |
| **Validation Checks** | · Number must be registered in the database.<br>· Entered OTP should be correct. |

**Table 3.7: Login Screen**

**3. DTI Dashboard**

| | |
|---|---|
| **Purpose** | · This page is reached when the DTI logs in successfully. |
| **Description of Each Field** | · Filter: DTI can filter Lead.<br>· Add: can add information about Lead. |

| | |
|---|---|
| | · Edit: can edit Lead information. |
| **Validation Checks** | · The mandatory fields should be filled. |

**Table 3.8: DTI Dashboard**

**4. Driver Dashboard**

| | |
|---|---|
| **Purpose** | · This page is reached when the driver logs in successfully. |
| **Description of Each Field** | · Driver can edit its Name.<br>· Driver can edit its address, hours of availability, interested location, etc. |
| **Validation Checks** | · The mandatory fields should be filled. |

**Table 3.9: Driver Dashboard**

**5. Admin Dashboard**

| | |
|---|---|
| **Purpose** | · This page is reached when the admin logs in successfully. |
| **Description of Each Field** | · Admin can manage users.<br>· Admin can add information about DTI, Driver and Lead.<br>· Admin can change status of any Entity. |
| **Validation Checks** | · The mandatory fields should be filled. |

**Table 3.10: Admin Dashboard**

## 3.3 Use Case Description

### 3.3.1 Login

| | |
|---|---|
| **Use Case 01** | **Login** |

| | |
|---|---|
| **Objective** | Helps us in considering the login process of the user login. |
| **Users** | Admin, DTI, Driver |
| **Pre Condition** | Should be registered on DSP. |
| **Post Condition** | The actor is provided with dashboard to perform operations. |
| **Basic Flow** | • The actor enters registered mobile number.<br>• Actor receive an OTP.<br>• Actor confirm received OTP<br>• The system displays appropriate dashboard. |
| **Alternative Flow** | · Actor entered not registered number.<br>· Actor will get an error message on the screen and will be prompt to signup accordingly. |

**Table 3.11: Use Case Description of Login**

### 3.3.2 Sign up

| | |
|---|---|
| **Use Case 02** | **View Service** |
| **Objective** | Helps us in considering the process of user's signup. |
| **Users** | DTI, Driver |
| **Pre Condition** | Should have opened the Sign up page and have a number that is not registered with DSP. |
| **Post Condition** | If the use case is successful, Actor will be registered with DSP |

| | |
|---|---|
| **Basic Flow** | • The actor fills the mobile number and if number is not already registered, he will be redirected to primary detail page to fill.<br>• Actor fills the secondary detail. |
| **Alternative Flow** | • Actor enter the mobile number already registered with DSP.<br>• Actor is shown an error message along with a prompt to login using that number. |

**Table 3.12: Use Case Description of Sign up**

### 3.3.3 Manage Lead

| **Use Case 03** | **View Project** |
|---|---|
| **Objective** | Helps us determine the process of adding, removing or editing profiles |
| **Users** | DTI, Admin |
| **Pre-Condition** | Should be logged into DSP dashboard. |
| **Post Condition** | If the use case is successful, then actor is able to add, remove or edit lead. |
| **Basic Flow** | • An add button is clicked to add a lead.<br>• Actor click on cross button to delete a lead.<br>• Actor click on edit button to edit a lead. |

| | |
|---|---|
| **Alternative Flow** | · If actor is not logged in, actor will be prompted to registered as DSP. |

**Table 3.13: Use Case Description of Manage Lead**

### 3.3.4 Manage DTI

| | |
|---|---|
| **Use Case 03** | **View Project** |
| **Objective** | Helps us determine the process of adding, removing or editing profiles |
| **Users** | Admin |
| **Pre Condition** | Should be logged into DSP dashboard. |
| **Post Condition** | If the use case is successful, then actor is able to add, remove or edit lead. |
| **Basic Flow** | • An add button is clicked to add a DTI.<br>• Actor click on cross button to delete a DTI.<br>• Actor click on edit button to edit a DTI. |
| **Alternative Flow** | · If actor is not logged in, actor will be redirected to sign up page. |

**Table 3.14: Use Case Description of Manage DTI**

# CHAPTER 4
# TESTING

4.1 Testing Activities

4.2 Unit Testing

- 4.2.1 Methodology Used
- 4.2.2 Tools Used
- 4.2.3 Test Cases

4.3 Integration Testing

- 4.3.1 Methodology Used
- 4.3.2 Tools Used
- 4.3.3 Test Cases

4.4 System Testing

- 4.4.1 Functional Testing
  - 4.4.1.1 Methodology Used
  - 4.4.1.2 Tools Used
  - 4.4.1.3 Test Cases
- 4.4.2 Non-Functional Testing
  - 4.4.2.1 Methodology Used
  - 4.4.2.2 Tools Used
  - 4.4.2.3 Test Cases

4.5 Test Reports and Debugging

4.6 Implementation

## 4.1 Testing Activities

- **Development team testing**

Entire group system was pursued where individuals with testing abilities were successfully inserted into the advancement group and the group was in charge of most of the testing. This methodology functions admirably.

- **End-of-lifecycle testing**

A significant piece of the discharge exertion was end-of-lifecycle testing where an autonomous test group approves that the framework was prepared to go into creation.

## 4.2 Unit Testing

### 4.2.1 Methodology Used

Unit testing was carried out by the developer environment only. Manual testing was done. The developers review their code to check whether their respective units under tests behave as expected.

### 4.2.2 Tools Used

Not applicable.

Manual testing was carried out in DSP.

### 4.2.3 Test Cases

| Test Id | Test Case Name | Test Case Description |
|---|---|---|
| 1. | Login only with correct credentials. | With registered mobile number and OTP, user should be easily logged in. |
| 2. | Error on Invalid mobile number. | When anon-registered mobile number is used to login, a proper error message must be displayed. |

| | | |
|---|---|---|
| 3. | EditLead/ Driver/ DTI | While retrieving information, new information should be updated in the database. |
| 4. | Add Lead/ Driver/ DTI | While retrieving information, new information should be added in the database. |
| 5. | Delete Lead/ Driver/ DTI | While retrieving information which has to be deleted, necessary changes should be reflected. |

**Table 4.1: Unit Test Cases**

## 4.3 Integration Testing

### 4.3.1 Methodology Used

Integration testing was carried out using bottom up approach where the lowest level components are tested first, and then used to facilitate the testing of higher level components. The process was repeated until the component at the top of the hierarchy is used.

### 4.3.2 Tools Used

Not applicable.

Manual testing was carried out in DSP

### 4.3.3 Test Cases

| **Test Id** | **Test Case Name** | **Test Case Description** |
|---|---|---|
| 1. | Manage Lead | Manage all the functions that can be performed with respect to Lead. |

| | | |
|---|---|---|
| 2. | Manage DTI | Manage all the functions that can be performed with respect to DTI. |
| 3. | Manage Driver | Manage all the functions that can be performed with respect to Driver. |
| 4. | Manage Admin | Manage all the functions that can be performed with respect to Admin. |

**Table 4.2: Integration Test Cases**

## 4.4 System Testing

### 4.4.1 Functional Testing

#### 4.4.1.1 Methodology Used

Under this the entire framework was tried under the advancement group.

Essentially, all functionalities according to prerequisites are tried here.

#### 4.4.1.2 Tools Used

None.

Testing was manually carried out in DSP

#### 4.4.1.3 Test Cases

| **Test Id** | **Test Case Name** | **Test Case Description** |
|---|---|---|
| 1. | SuperuserTest Case | They have high level access over the system |

| | | |
|---|---|---|
| 2. | Admin Test case | Admin should be able to edit and manage his profile along with DTI, Driver and Lead. |
| 3. | DTITest case | DTI should be able to edit and manage his profile along with Lead profile. |
| 4. | DriverTest case | Driver should be able to edit and manage his profile. |
| 5. | LeadTest case | Lead should be able to check his status of his application on DSP. |

**Table 4.3: Functional Test Cases**

### 4.4.2 Non-Functional Testing

#### 4.4.2.1 Methodology Used

Under this the whole system was tested under the development team.

Basically, all functionalities as per requirements are tested here.

#### 4.4.2.2 Tools Used

Not Applicable.

Manual testing was carried out in DSP

#### 4.4.2.3 Test Cases

| Test Id | Test Case Name | Test Case Description |
|---|---|---|
| 1. | Check for accuracy while displaying information | The update as Interface Layer should be as it was mapped to the database via Business Layer. |

| | | |
|---|---|---|
| 2. | System stability state | System's ability to recover and save last state. |

**Table 4.4: Non-Functional Test Cases**

## 4.5 Test Reports and Debugging

**· Unit Testing Report**

| Test Id | Test Case Name | Test Case Description | Test Case Input | Test Results | | Test Case Status |
|---|---|---|---|---|---|---|
| | | | | Expected | Actual | |
| 1. | Login only with correct credentials. | With registered mobile number and OTP, user should be easily logged in. | Registered mobile number | Should Login | Logged In | Pass |
| 2. | Error on | When an non | Not | Error | Error | Pass |
| | Invalid mobile number. | registered mobile number is used to login, a proper error message must be displayed. | registered mobile number | Should Appear | Raised | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 3. | EditLead/ Driver/ DTI | While retrieving information, new information should be updated in the database. | Valid Data | Records Updated | Records Updated | Pass |
| 4. | Add Lead/ Driver/ DTI | While retrieving information, new information should be added in the database. | Valid Data | Added and Records Updated | Added and Records Updated | Pass |
| 5 | Delete Lead/ Driver/ DTI | While retrieving information which has to be deleted, necessary changes should be reflected. | Valid Data | Deleted and Records Updated | Deleted and Records Updated | Pass |

**Table 4.5: Unit Testing Report · Integration Testing Report**

| **Test Id** | **Test Case Name** | **Test Case Description** | **Test Case Input** | **Test Results** | | **Test Case Status** |
|---|---|---|---|---|---|---|
| | | | | **Expected** | **Actual** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1. | Manage Lead | Manage all the functions that can be performed with respect to Lead. | Valid Data | Should Appear | Appears | Pass |
| 2. | Manage DTI | Manage all the functions that can be performed with respect to DTI. | Valid Data | Should Appear | Appears | Pass |
| 3. | Manage Driver | Manage all the functions that can be performed with respect to Driver. | Valid Data | Should Appear | Appears | Pass |
| 4. | Manage Admin | Manage all the functions that can be performed with respect to Admin. | Valid Data | Should Appear | Appears | Pass |

**Table 4.6: Integration Testing Report**

**System Testing Report · Functional Testing Report**

| **Test Id** | **Test Case Name** | **Test Case Description** | **Test Case Input** | **Test Results** | | **Test Case Status** |
|---|---|---|---|---|---|---|
| | | | | **Expected** | **Actual** | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 1. | Superuser Test Case | Superuser should have complete control over the system. | Admin Section Checked for superuser Role | All Admin Management functions available | Functions Working and Available | Pass |
| 2. | AdminTest Case | Admin should be able to edit and manage his profile along with DTI, Driver and Lead. | Detail to be updated. | Admin Should be Able to Access the Page to update detail. | Admin was able to successfully update details. | Pass |
| 3. | DTITest Case | DTI should be able to edit and manage his profile along with Driver and Lead. | Detail to be updated. | User Should be Able to Access the Page to update detail. | DTI was able to successfully update details. | Pass |
| 4. | DriverTest Case | Driver should be able to edit | Driver Input his detail to | All Driver related | Functions Worki | Pass |
| | | and manage his profile. | update. | function works as expected. | ng and Available | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 5. | LeadTest Case | Lead should be able to Check his application status. | Flow Checked With Backward Redirection | Screen Flow as per Requirements | Screen Flow as per Requirements | Pass |

**Table 4.7: Functional Testing Report**

· **Non-Functional Testing Report**

| Test Id | Test Case Name | Test Case Description | Test Case Input | Test Results | | Test Case Status |
|---|---|---|---|---|---|---|
| | | | | Expected | Actual | |
| 1. | Check for accuracy while displaying information | The update as Interface Layer should be as it was mapped to the database via Business Layer. | Update Performed on Various Entities via Admin Section | All Data Mapped Should be Exactly Same | All Data Perfectly Mapped | Pass |
| 2. | System stability state | System's ability to recover and save last state. | Check for System on Failure | On Failure redirects to proper error page. | On Failure redirects to proper error page. | Pass |

**Table 4.8: Non-Functional Testing Report**

## 4.6 Implementation

The tests were first actualized on individual units under Unit Testing. Every one of functionalities inputs were separated as Valid or Invalid Inputs and after that a blend of these was tried on the concerned Functional Modules. The consequences of these sources of info were depicted in experiment results region. Utilization of Stubs and Drivers any place important was finished.

Under System Testing the total usefulness of the framework according to the prerequisite was tried. Every single prerequisite was examined and its usage with the concerned usefulness was given.

# CHAPTER 5
# CONCLUSION AND REFERENCES

5.1 Conclusion

5.2 Drawbacks and constraints of the System

5.3 Plans for further improvements

5.4 References/Bibliography

## 5.1 Conclusion

A Driving Sourcing Panel has been developed where main interacting actors are Lead, Driver, DTI and Admin. DSP makes it easier for drivers to get a job with Uber and provide them a chance to be registered and drive for them whether if a driver have its own car or not.

- The DTI can now register on the platform to partner with Uber as a DTI.
- The driver can now register on the platform as a driver and find opportunities to earn on Uber.
- In case of Lead registrations, the drivers would be able to check their status using their phone number on OTP verification.

The admin can now do the following functions:

1) View & Download the list of entire Driver Leads uploaded by DTIs and Drivers
   a) Sort & Filter to view and download the leads based on DTI Name
   b) Sort & Filter to view and download the leads based on Uploaded Date
   c) Bulk select & delete the leads based on search filters if any
   d) Search for any lead based on phone number
   e) Add new leads / Edit the information of any lead

2) View the list of DTI's registered on the portal
   a).Search for any DTI's
   b).Activate/deactivate that DTI account.
   c).Reset password/phone number associated with that account

3) Update the status of the leads
   a).Update status of individual leads
   b).Update status of leads in bulk using csv
4) Admin will be able to manage the registration form as:

a).Add new field to form

b).Edit any field

c).Delete any field to form

5) Content Management by Admin: Static sections of the website should be manageable

- Info pages
- CSV sample for DTI's

### 5.2 Drawbacks and constraints of the System

Can automate the system where Driver Training Institute will get Flagged as favorite when Drivers coming from particular institute performs well.

### 5.3 Plans for further improvements

- The Notification Module can be integrated.
- The Flagging DTI in the system as favourite automatically.
- The Analytics is yet to be implemented for system.
- Further dynamic requirements can also be implemented in the system.

### 5.4 References

**Online**

[1]. https://docs.djangoproject.com/en/2.0/

[2]. https://docs.python.org/

[3]. http://www.django-rest-framework.org/

**Books**

[1]. Django: Web Development with Python - by Samuel Dauzon, AidasBendoraitis, Arun Ravindran

# CHAPTER 6 ANNEXURES

A-1 USE CASE Diagram

A-2 DFD Diagrams

A-3 Activity Diagram

A-4 ERD Diagram

A-5 Screenshots

## A-1 Use Case Diagram

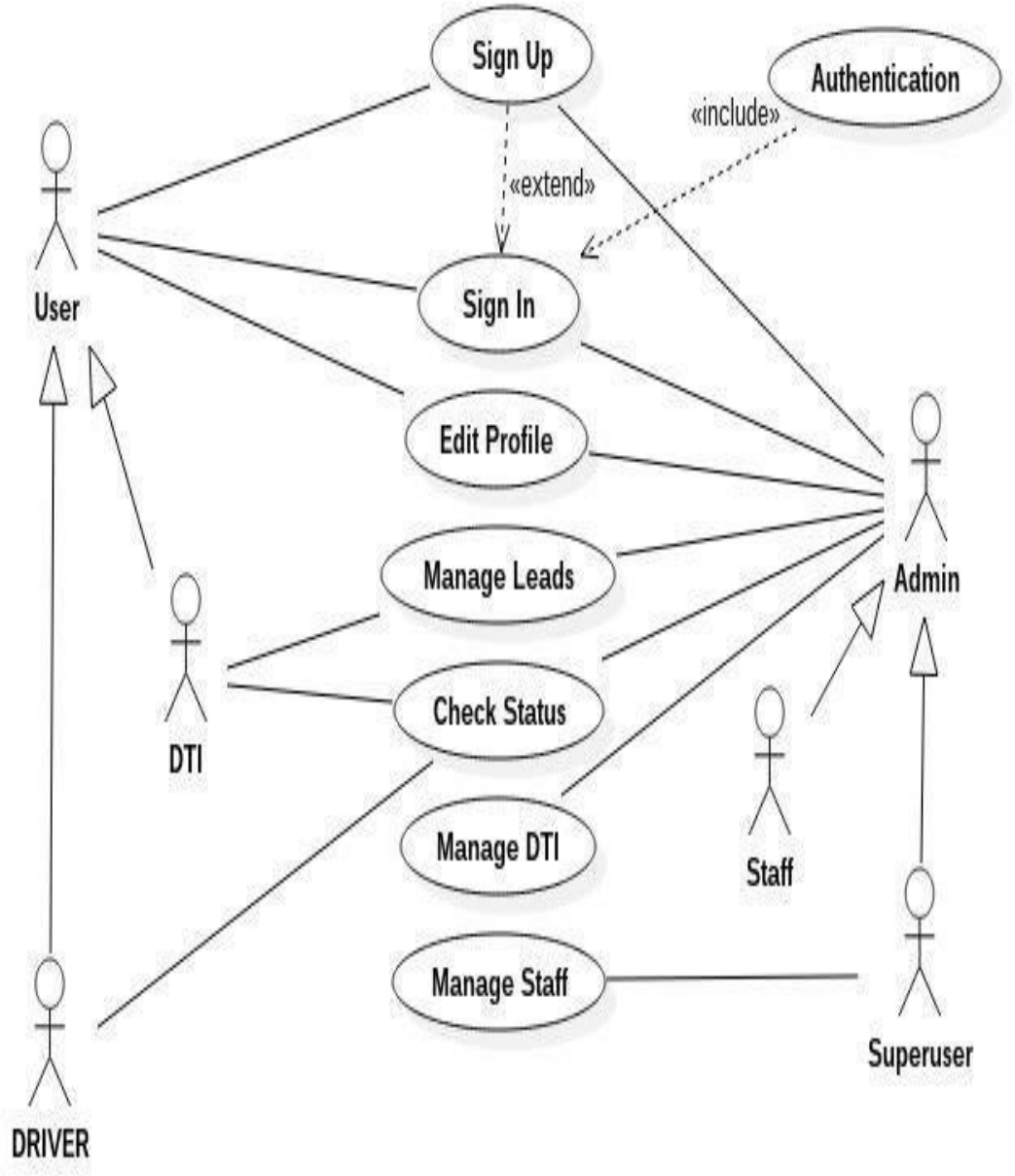


**Figure 6.1: Use Case Diagram for DSP**

## A-2 DFD Diagrams

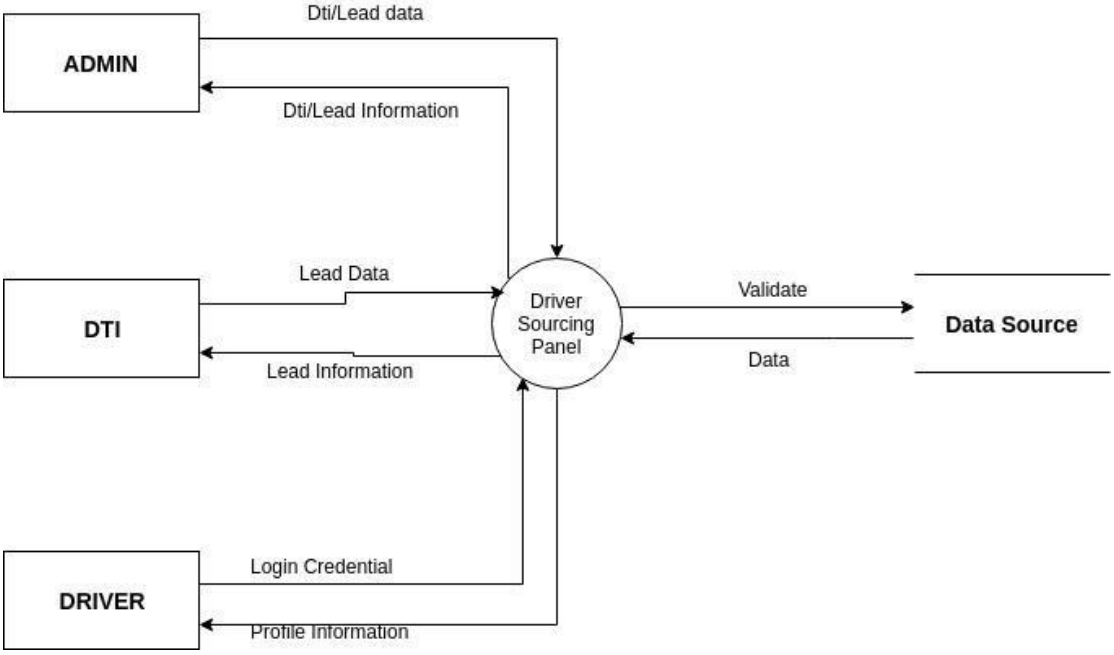


**Figure 6.2: Level 0 DFD for DSP**

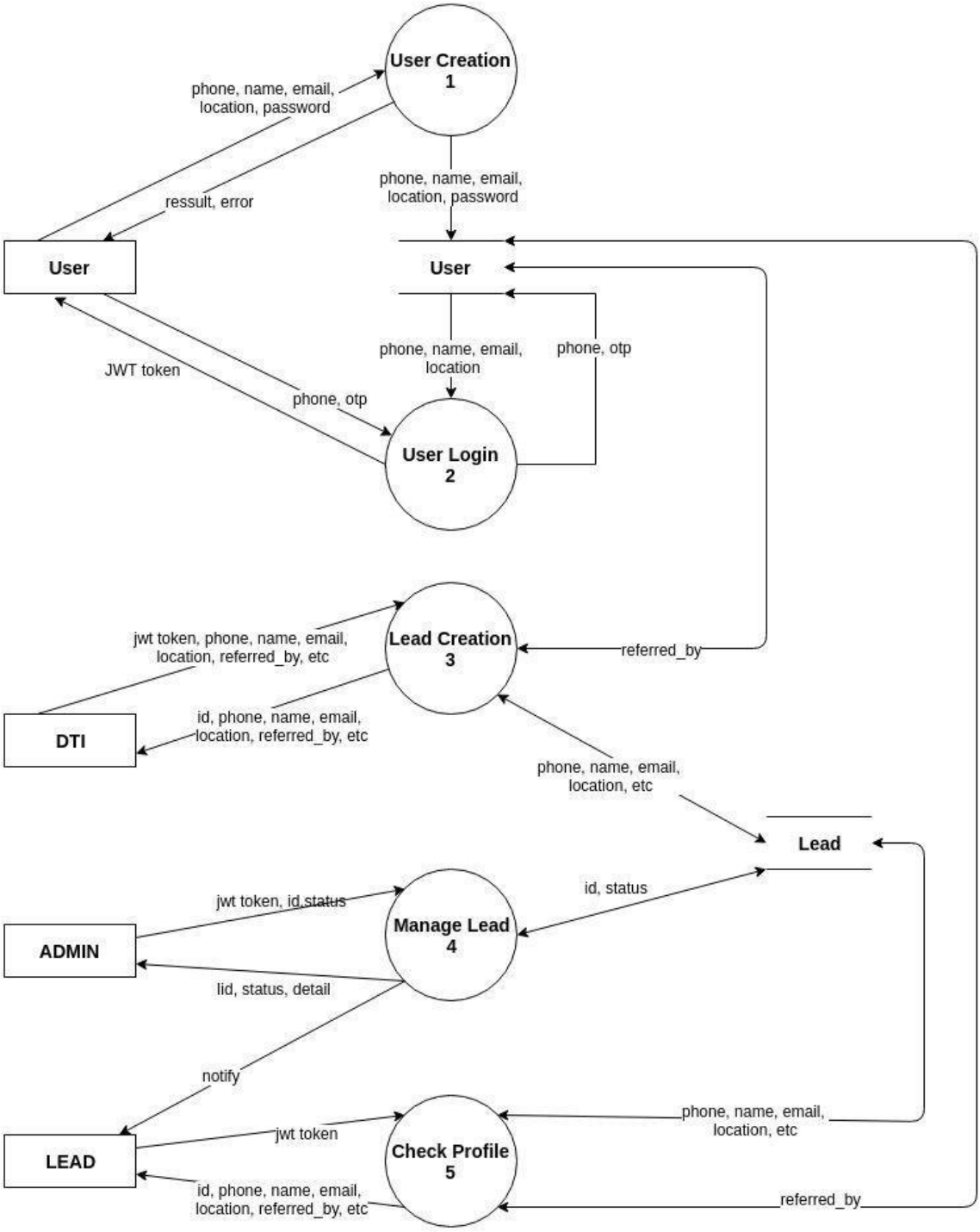


**Figure 6.3: Level 1 DFD for DSP**

## A-3 Activity Diagram

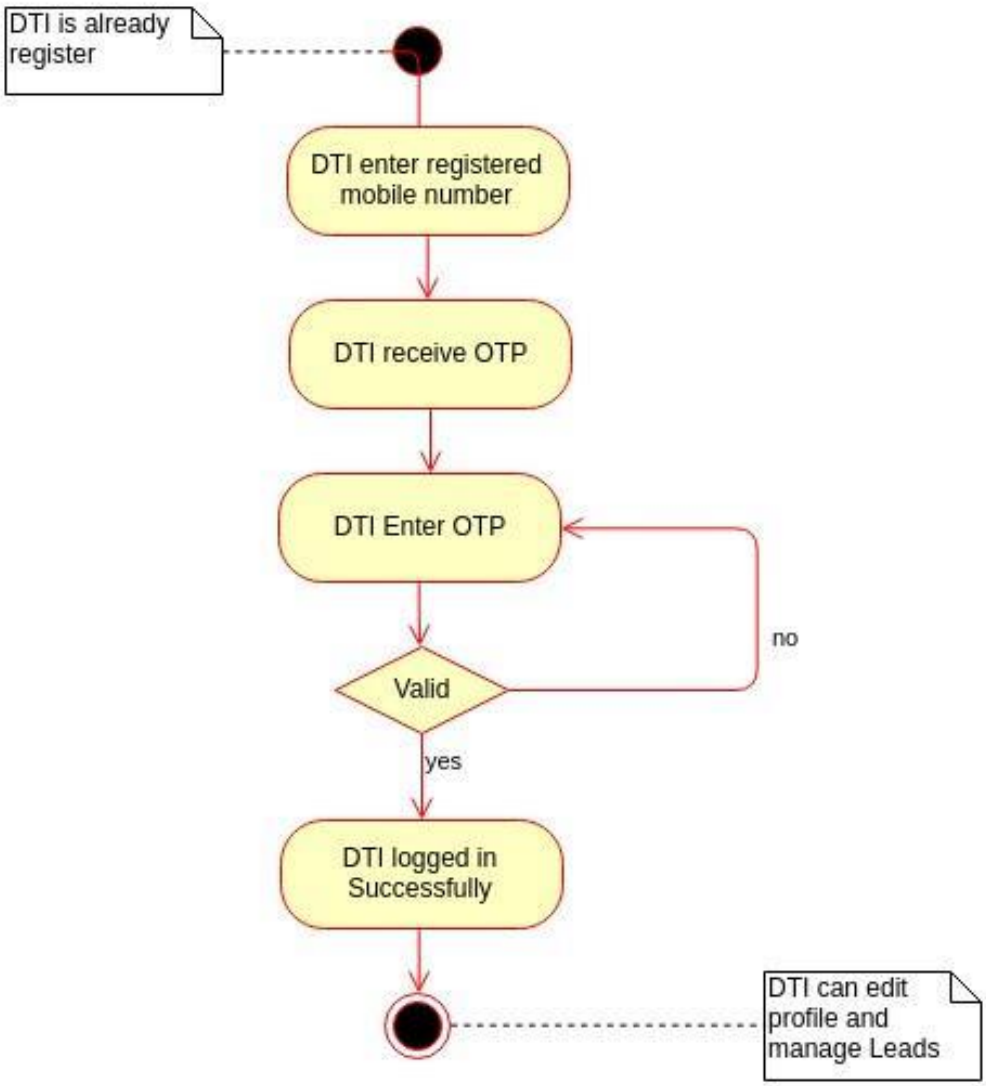


**Figure 6.4: Activity Diagram – DTI Login for DSP**

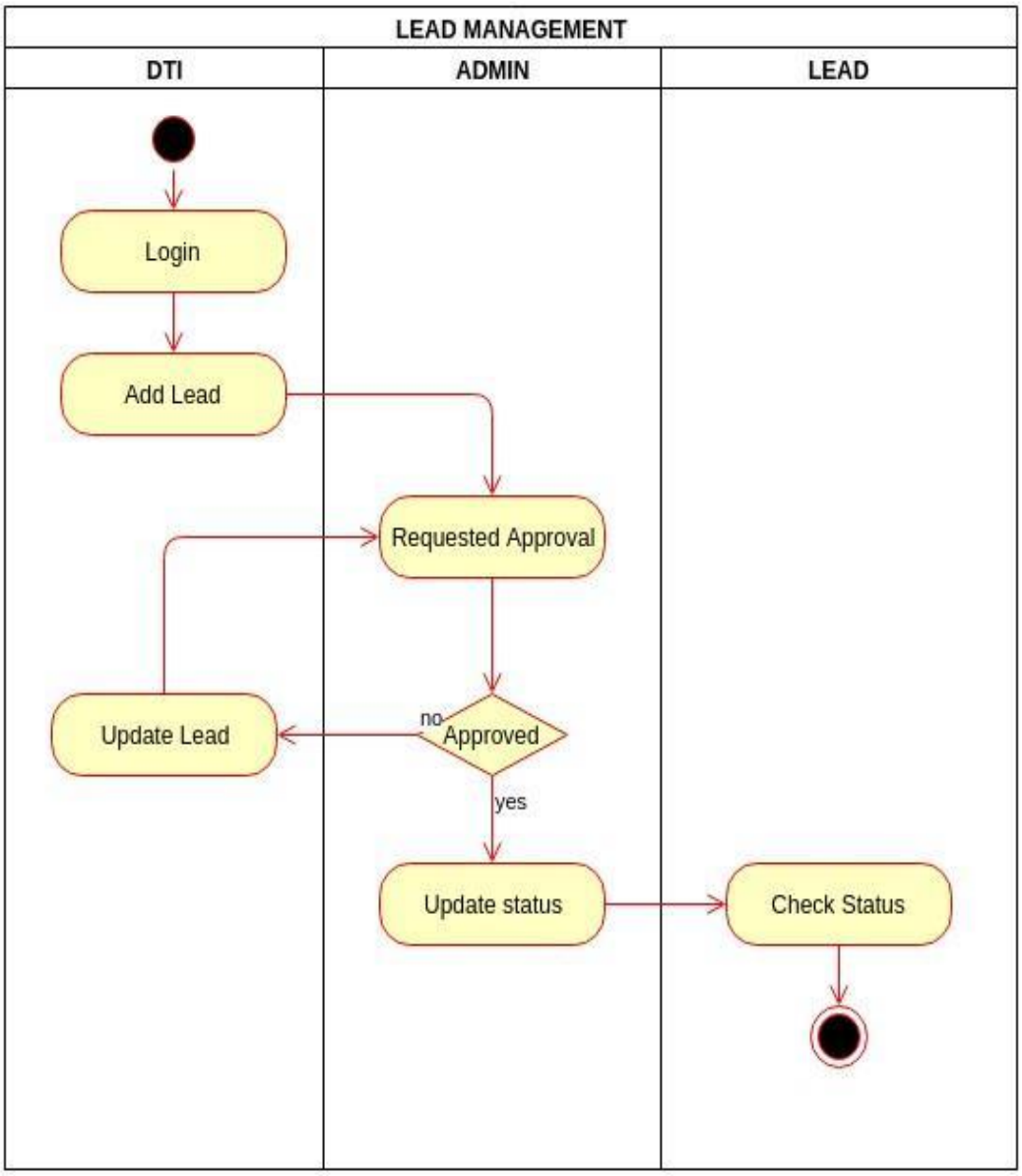


**Figure 6.5: Activity Diagram – Lead Management for DSP**

## A-4 ERD Diagram

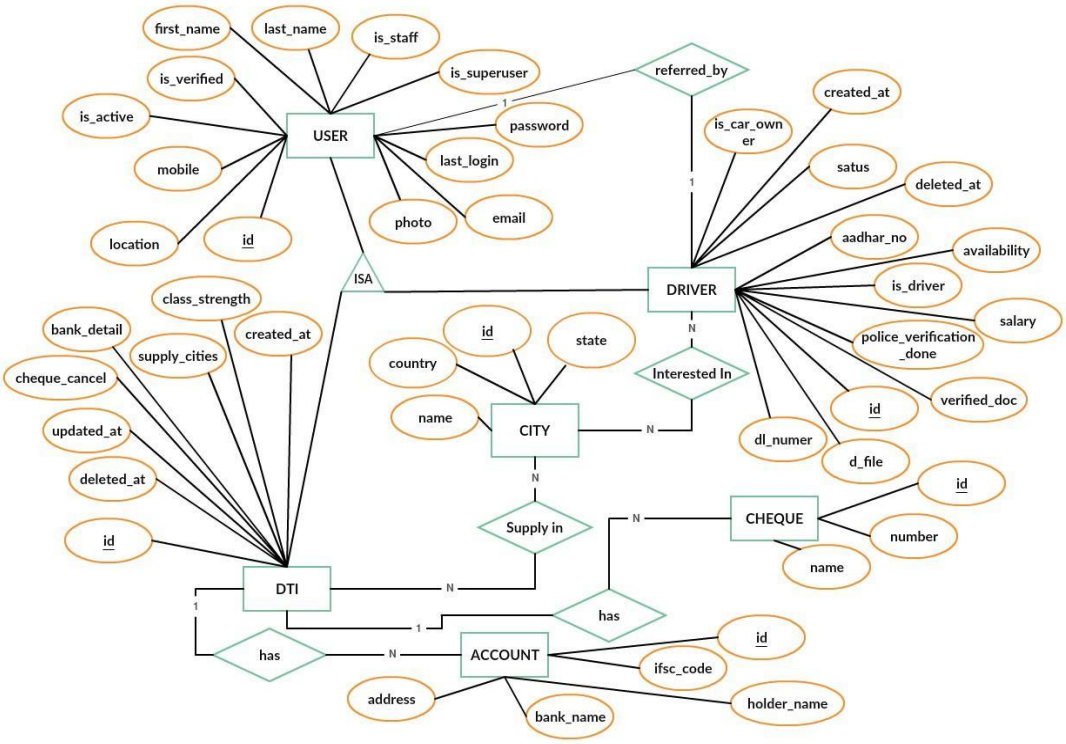


**Figure 6.6: ER Diagram for DSP**

## A-5 Screenshots

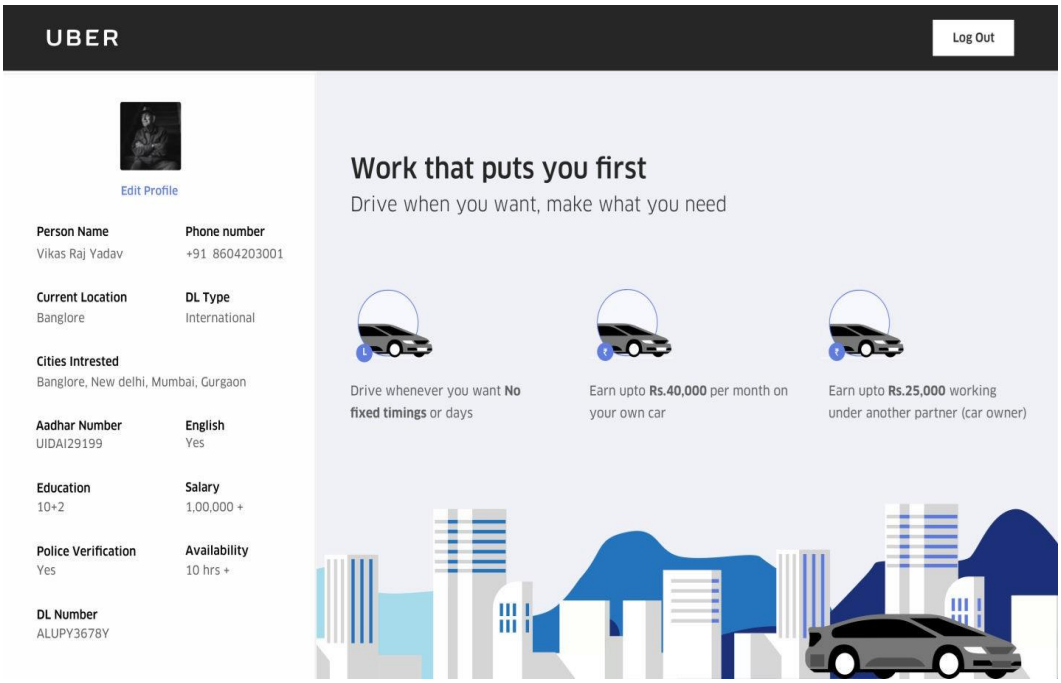

**Figure 6.7: Screen for Driver/Lead Profile**

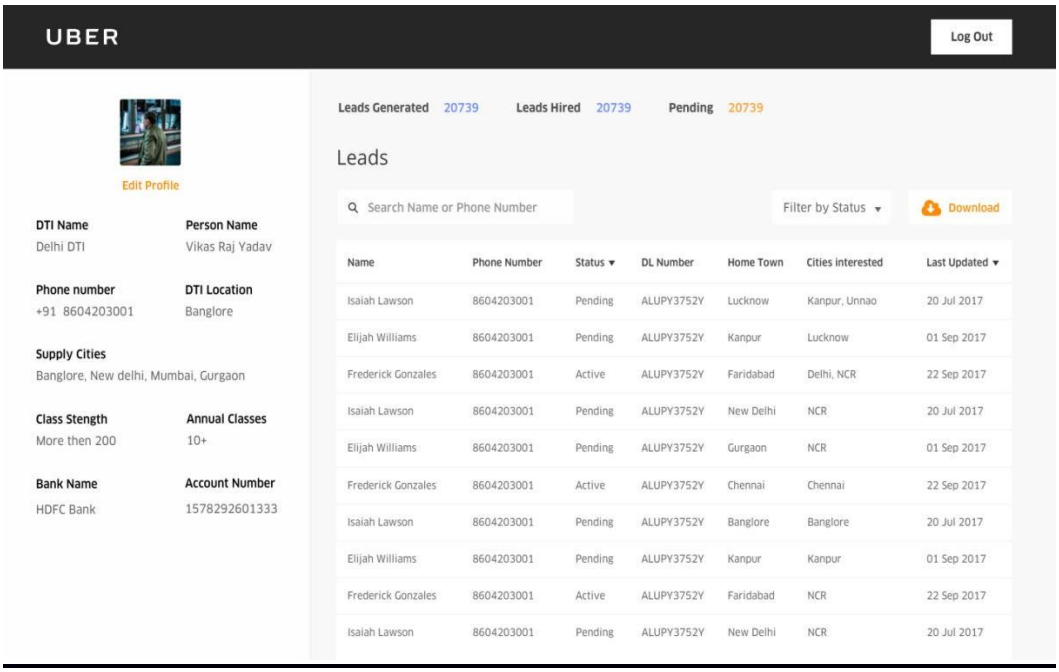


**Figure 6.8: Screen for DTI Dashboard**

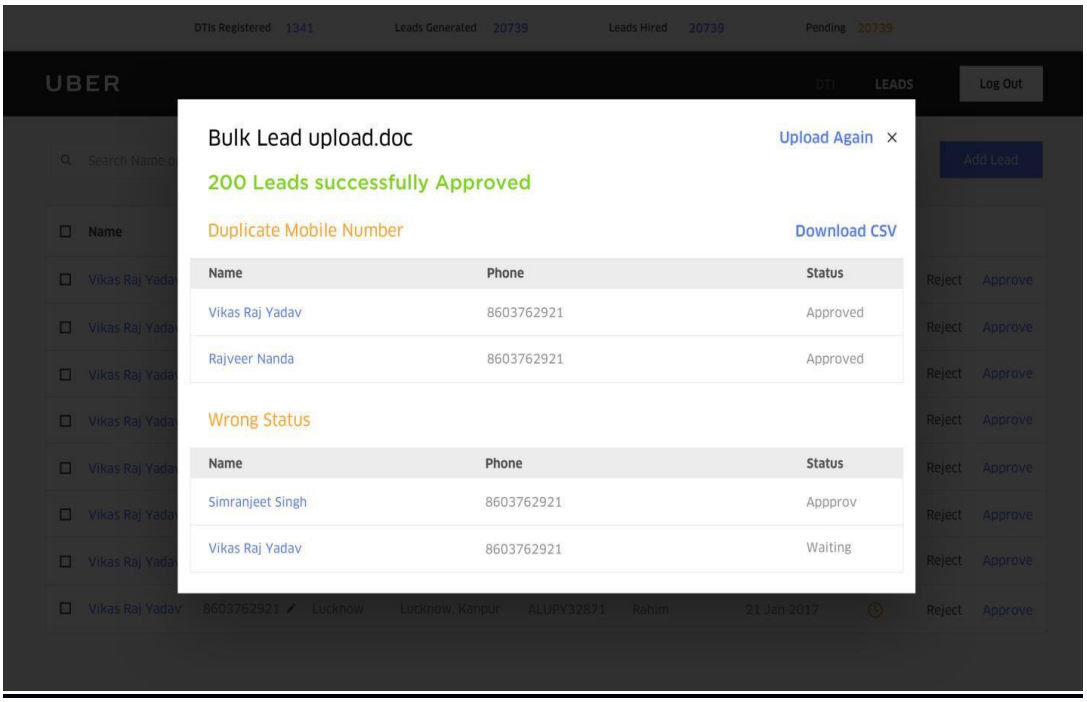

**Figure 6.9: Screen for Lead Upload result**

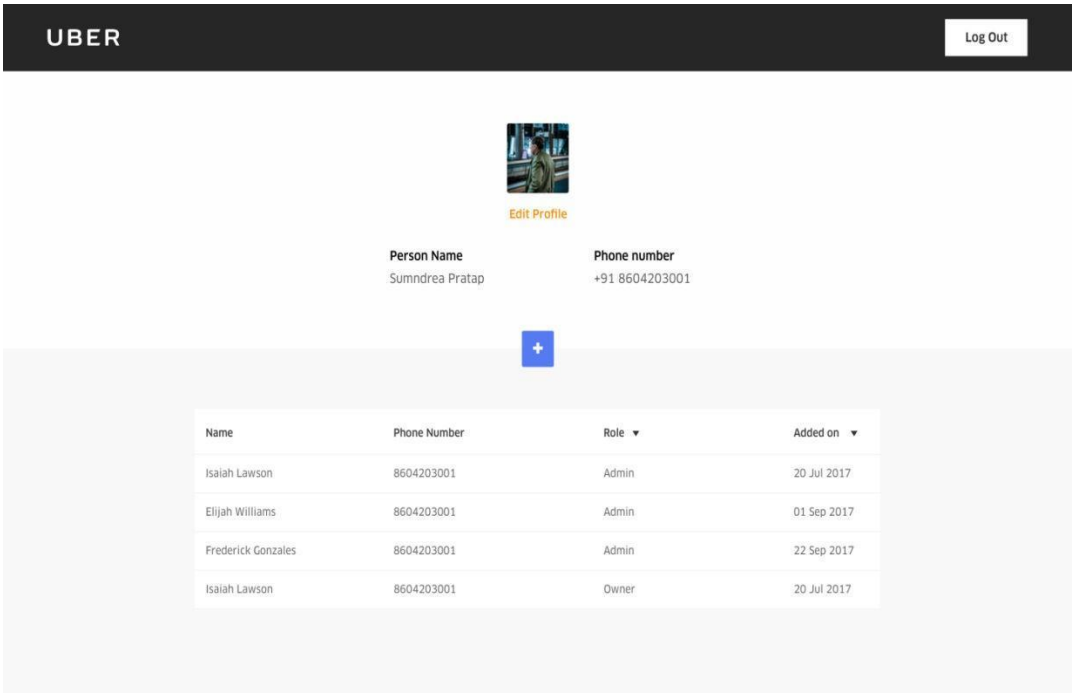


**Figure 6.10: Screen for Admin Profile**

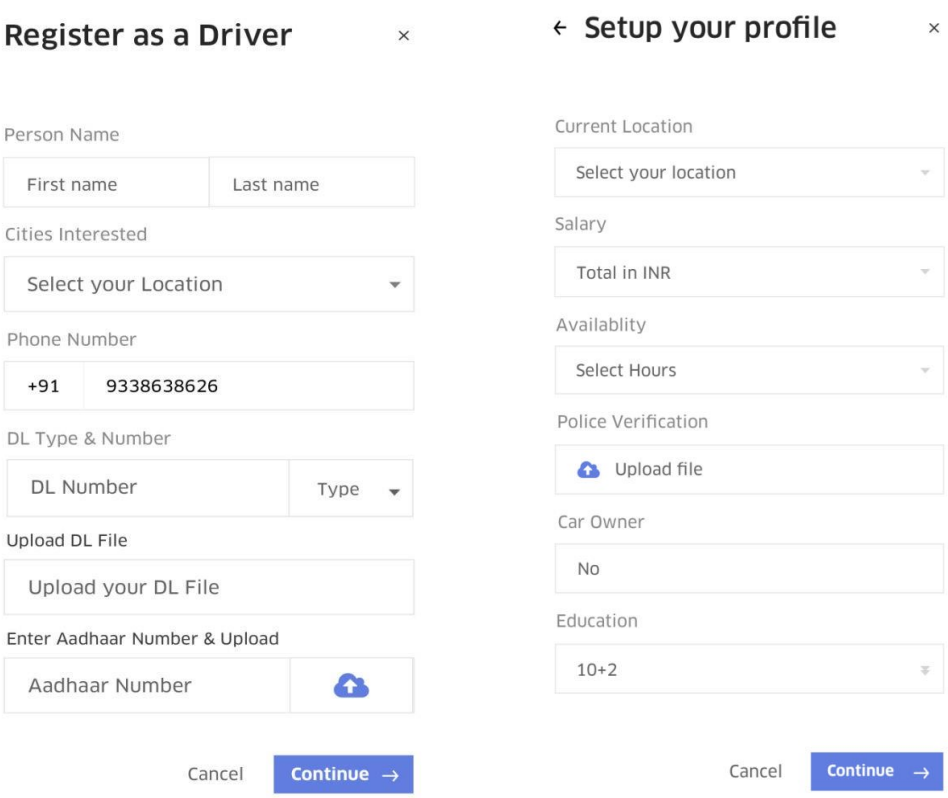


**Figure 6.11: Responsive Screen for Driver Sign up**