**DECENRALIZED VEHICLE BOOKING SERVICE**

**Major Project report submitted in partial fulfillment of the Requirements for the Award of   the Degree of**

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

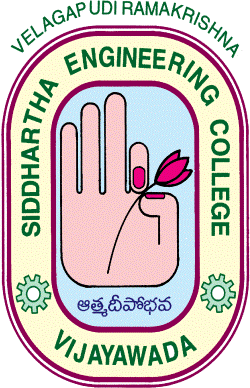
**By**

**P.HEMA   178W1A05A1**

**K.GAYATHRI PHANI KUMAR 178W1A0583**

**P.SRAVANI 188W5A0520**

**P.SATYENDRA 178W1A05A2**



**Under the Guidance of**

**Mr. Ch. Mukesh**

**Assistant Professor**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**V.R SIDDHARTHA ENGINEERING COLLEGE**

**Autonomous and Approved by AICTE**

**Affiliated to Jawaharlal Nehru Technological University, Kakinada**

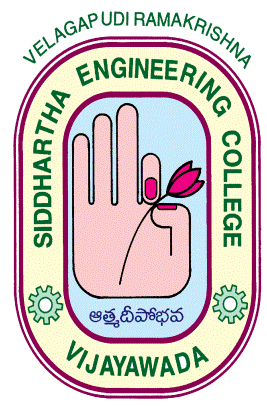
**Vijayawada 520007**

**2021**

**V.R.SIDDHARTHA ENGINEERING COLLEGE**

**(AUTONOMOUS)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**CERTIFICATE**

This is to certify that the EPICS Report entitled **“DECENTRALIZED VEHICLE BOOKING SERVICE ”** being **s**ubmitted by **P.HEMA (178W1A05A1), K.GAYATHRI PHANI KUMAR (178W1A0583), P.SRAVANI (188W5A0520), P.SATYENDRA (178W1A05A2)** in partial fulfillment for the award of the Degree of Bachelor of Technology in Computer Science and Engineering to the Jawaharlal Nehru Technological University, Kakinada is a record of bonafide work carried out under my guidance and supervision.

**Mr. Ch. Mukesh, M.S Dr. D Rajeswara Rao, M.Tech,Ph.D**

**Assistant Professor & Guide                                     Professor & HOD**

**DECLARATION**

We hereby declare that the EPICS project entitled **“DECENTRALIZED VEHICLE BOOKING SERVICE”** submitted for the B.Tech Degree is our original work and the dissertation has  not formed the basis for the award of any degree, associateship, fellowship or any other similar titles.

**Place: Vijayawada**

**Date:**

**P.HEMA   178W1A05A1**

**K.GAYATHRI PHANI KUMAR 178W1A0583**

**P.SRAVANI 188W5A0520**

**P.SATYENDRA 178W1A05A2**

**ACKNOWLEDGEMENT**

Behind every achievement lies an unfathomable sea of gratitude to those who activated it, without whom it would ever have come into existence. To them we lay the words of gratitude imprinted with us.

We would like to thank our respected Principal, **Dr. A.V. Ratna Prasad** and **Dr. D. Rajeswara Rao,** Head of the Department, Computer Science and Engineering for their support throughout our project.

It is our sincere obligation to thank our guide, **Mr. Ch. Mukesh,**Assistant Professor, Department of Computer Science and Engineering, for his timely valuable guidance and suggestions for this project.

We owe our acknowledgements to an equally long list of people who helped us in this Project. Finally, we wish to thank all the supporting staff who provided lab facilities for the completion of this Project.

**Place:** Vijayawada 

**P.HEMA   178W1A05A1**

**K.GAYATHRI PHANI KUMAR 178W1A0583**

**P.SRAVANI 188W5A0520**

**P.SATYENDRA 178W1A05A2**

# ABSTRACT

Online vehicle booking service became a growing need now-a-days as it offers an efficient and cheaper alternative. One can directly call a taxi to any location at any time they want without the need for a lengthy delay. There are various platforms implementing online vehicle booking service but the major issue with them is having a centralized authority which are pocketing huge profits. A fixed percentage of money paid by the user is taken by the centralized authority. These centralized systems also tend to misuse their user’s data and are also prone to several attacks. One of the best solutions to this problem is to use Blockchain. Our objective is to avoid this centralized authority by creating a decentralized application. Not just removing the central authority, Blockchain also provides several features like immutability, better transparency, enhanced security and traceability. Platforms like Ethereum and Hyperledger allows one to develop decentralized applications and deploy in the blockchain. Smart Contracts can be developed using Solidity to achieve decentralization. This maintains transparency, immutability and removes a central authority which may misuse its user's information**.**

**Keywords:** Decentralization, Ethereum, HyperLedger, Smart Contracts

Table of Contents

[1 INTRODUCTION 10](#_Toc75960873)

[1.1 BASIC CONCEPTS: 11](#_Toc75960874)

[1.1.1 Blockchain: 11](#_Toc75960875)

[1.1.2 Decentralized Applications (DApps): 11](#_Toc75960876)

[1.1.3 Smart Contract: 12](#_Toc75960877)

[1.1.4 Ethereum: 12](#_Toc75960878)

[1.1.5 Tokens: 12](#_Toc75960879)

[1.1 MOTIVATION: 13](#_Toc75960880)

[1.2 PROBLEM STATEMENT: 13](#_Toc75960881)

[1.3 SCOPE: 13](#_Toc75960882)

[1.4 OBJECTIVES: 13](#_Toc75960883)

[1.5 ADVANTAGES: 13](#_Toc75960884)

[1.6 DISADVANTAGES: 13](#_Toc75960885)

[2 LITERATURE SURVEY 14](#_Toc75960886)

[2.1 EtherRent: A Co-operative Car Rental Platform 14](#_Toc75960887)

[2.2 Decentralized Applications: The Blockchain-Empowered Software System 15](#_Toc75960888)

[2.3 Toward a Secure and Decentralized Blockchain-based Ride-Hailing Platform for Autonomous Vehicles 16](#_Toc75960889)

[2.4 Blockchain in Healthcare 17](#_Toc75960890)

[2.5 Blockchain for the Internet of Vehicles: A Decentralized IoT Solution for Vehicles Communication using Ethereum 18](#_Toc75960891)

[3 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS 19](#_Toc75960892)

[3.1 SOFWARE REQUIREMENTS: 19](#_Toc75960893)

[3.1.1 Remix ide: 19](#_Toc75960894)

[3.1.2 Visual Studio Code: 19](#_Toc75960895)

[3.1.3 MetaMask: 20](#_Toc75960896)

[4 SOFTWARE DESIGN 21](#_Toc75960897)

[4.1 SOFTWARE MODEL: 21](#_Toc75960898)

[4.2 PROCESS FLOW DIAGARAM: 22](#_Toc75960899)

[4.3 USE CASE DIAGRAM: 23](#_Toc75960900)

[4.4 SEQUENCE DIAGRAM: 24](#_Toc75960901)

[5 PROPOSED SYSTEM 25](#_Toc75960902)

[5.1 METHODOLOGY: 25](#_Toc75960903)

[6 IMPLEMENTATION 26](#_Toc75960904)

[6.1 ALGORITHM: 26](#_Toc75960905)

[7 CODING 27](#_Toc75960906)

[7.1 project.sol 27](#_Toc75960907)

[7.2 home.html: 28](#_Toc75960908)

[7.3 DriverSignUp.html: 28](#_Toc75960909)

[7.4 DocumentUpload.html: 28](#_Toc75960910)

[7.5 DriverSignIn.html: 28](#_Toc75960911)

[7.6 Cust\_SignUp.html: 28](#_Toc75960912)

[7.7 UserSignIn.html: 29](#_Toc75960913)

[7.8 Booked.html: 29](#_Toc75960914)

[7.9 CommonScript.js: 29](#_Toc75960915)

[8 RESULT 31](#_Toc75960916)

[9 CONCLUSION AND FUTURE WORK 36](#_Toc75960917)

[10 REFERENCES 37](#_Toc75960918)

**LIST OF FIGURES**

[Fig. 1.1: Blockchain 10](#_Toc75946984)

[Fig. 1.2: Decentralized Application 20](#_Toc75946988)

[Fig. 3.1: Remix ide 18](#_Toc75946985)

[Fig. 3.2: Visual Studio Code 18](#_Toc75946986)

[Fig. 3.3: MetaMask 19](#_Toc75946987)

[Fig. 3.4: Water fall model 20](#_Toc75946988)

[Fig. 3.5: Process Flow Diagram 2](#_Toc75946988)1

[Fig. 3.6: Use Case Diagram 2](#_Toc75946988)2

[Fig. 3.7: Sequence Diagram 2](#_Toc75946988)3

[Fig. 6.1: Home Page 26](#_Toc75946989)

[Fig. 6.2: Customer SignUp page 26](#_Toc75946990)

[Fig. 6.3: Driver SignUp page 26](#_Toc75946991)

[Fig. 6.4: License uploading page 27](#_Toc75946992)

[Fig. 6.5: Customer home page 27](#_Toc75946993)

[Fig. 6.6: Status page 1 28](#_Toc75946994)

[Fig. 6.7: Driver home page 28](#_Toc75946995)

[Fig. 6.8: Status page 2 29](#_Toc75946996)

[Fig. 6.9: Drop Taking page 29](#_Toc75946997)

[Fig. 6.10: Status page 3 30](#_Toc75946998)

# INTRODUCTION

Online Vehicle Booking Services provide their users to hire a vehicle and help them to reach their destination. Now-a-days these services are available in all cities, towns and some villages. These services help the users from wasting their time looking for a taxi or auto hire to reach their destination. One can directly select their source and destination and the registered driver who accepts the drop can take the customer to their destination. Not only customers but also vehicle drivers get benefit from these services. Drivers need no wait for the customers or have to ask every person on their way whether they require a drop or not. The drivers can find all their customers from the application.

Though driver and customer get benefit from the service there is a third party, which is making money from driver and customer deals. That is nothing but the centralized owner of application. They charge certain percentage from every deal happened using their website. According to a survey, it is told that only 60% of the amount paid by customer using online vehicle booking service reaches the driver, and from the rest of 40% only 9-12% is taken as tax and the rest goes to the centralized owner. As per a trusted source, a top centralized cab booking service has a market value of 72 billion dollars as of May 2018 in U.S.A .It also stated that, if the trend of using cab booking service continues like this the total market value will soon reach 200 billion dollars. Our main motive is to make driver and customer to get maximum profit by avoiding the third party. One of the best ways to achieve this is by using blockchain, where there is no third party existence and control.

Block chain networks like Ethereum and Hyperledger provides its users to develop Decentralized web and mobile applications and allow those applications to be accessed either by any person or by set of people depending on the nature of the application. Public blockchain networks like Ethereum allows one to develop both public and private applications where as Hyperledger allow one to develop private applications which are accessible to only a set of users. Applications developed using blockchain are immutable, more secure, traceable, and free form a centralized authority which make money from our data and transactions.

## BASIC CONCEPTS:

### 1.1.1 Blockchain:

Blockchain is a decentralized, distributed ledger which is used to store immutable transactions. Blockchain removes a centralized authority by bringing in a consensus mechanisms like proof of work, proof of stake etc., to make the network more reliable. Every transaction done in blockchain must be accepted using consensus mechanism, if not accepted the transaction is made invalid. Blockchain is able to overcome issues like double spending, which is the major problem of digital currency. Blockchain is gaining its popularity from a few years and now it is being used all most all sectors.

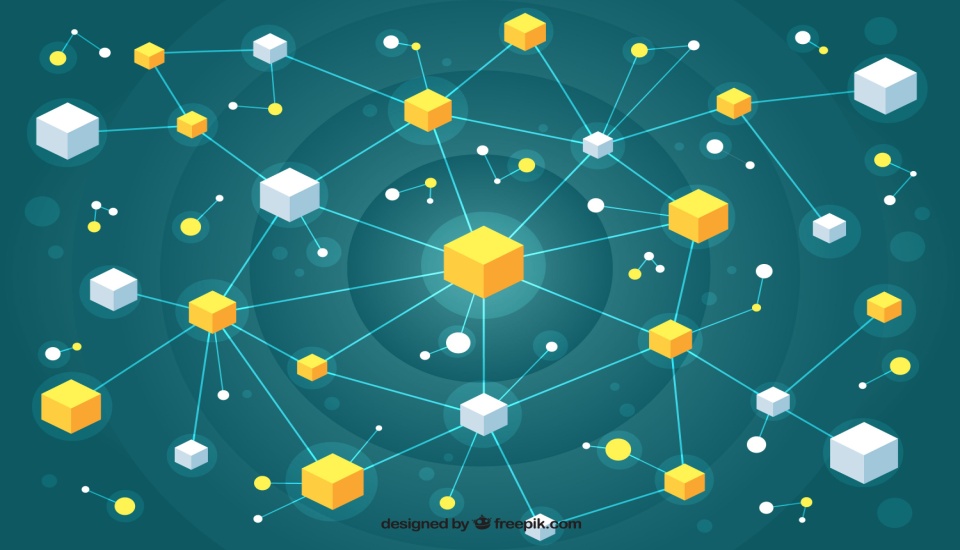


Fig. 1.1: Blockchain

### 1.1.2 Decentralized Applications (DApps):

Decentralized applications or DApps are the applications or programs that are developed and exist on top of a blockchain network. These applications can be developed using the guidelines and language provided by individual platforms. Blockchain networks like Ethereum and Hyperledger allows one to develop DApps on top of their network using solidity, go, python etc.

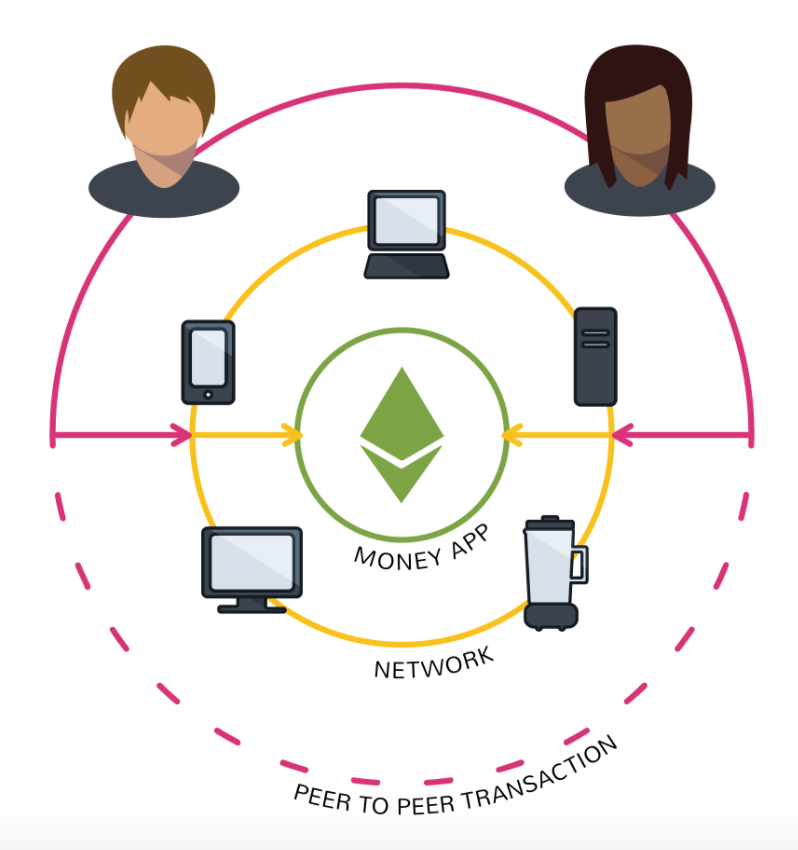


Fig. 1.2 Decentralized Application

### 1.1.3 Smart Contract:

A smart contract is a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of code. The code and the agreements contained therein exist across a distributed, decentralized [blockchain](https://www.investopedia.com/terms/b/blockchain.asp) network. The code controls the execution, and transactions are traceable and irreversible. Smart contracts permit trusted transactions and agreements to be carried out among disparate, anonymous parties without the need for a central authority, legal system, or external enforcement mechanism.

### 1.1.4 Ethereum:

Ethereum is a decentralized, open-source blockchain with smart contract functionality. Ether is the native cryptocurrency of the platform. After Bitcoin, it is the second-largest cryptocurrency by market capitalization. As a blockchain network, Ethereum is a decentralized public ledger for verifying and recording transactions. The network's users can create, publish, monetize, and use applications on the platform, and use its Ether cryptocurrency as payment. Insiders call the decentralized applications on the network Dapps.

### 1.1.5 Tokens:

Cryptographic tokens represent a set of rules, encoded in a smart contract – the token contract. Every token belongs to a blockchain address. These tokens are accessible with a dedicated wallet so ware that communicates with the blockchain and manages the public-private key pair related to the blockchain address. Only the person who has the private key for that address can access the respective tokens. This person can, therefore, be regarded as the owner or custodian of that token.

## MOTIVATION:

The need for decentralization is increasing day by day. The centralized systems are cashing their user’s data. Decentralization solves several problems of traditional centralized systems such as single point of failure, misuse of user’s data, commission from users etc.  Blockchain is an ideal technology to solve these problems and for implementation in trustless environments. As there is no central power no one is there to misuse the user’s information and hence solves the problem.

## PROBLEM STATEMENT:

To develop a Decentralized Vehicle Booking Service application which removes the centralized authority and provides a transparent, secure and immutable platform to book vehicle services.

## SCOPE:

Our proposed system can be used by normal people as well as vehicle drivers. Normal people can book vehicle service and vehicle drivers can accept the drops booked. Money transfer can be directly done between customers and drivers either as cash or crypto currency.

## OBJECTIVES:

* Build a decentralized vehicle booking service which removes centralized authority.
* Develop a model which can ensure transparency, immutability and traceability.
* Ensure trust by using IPFS to store vehicle driver’s identity (license).

## ADVANTAGES:

* No third party or centralized system
* Traceability of every action
* More secure

## DISADVANTAGES:

* Slower than centralized systems
* Gas must be paid for every transaction

# LITERATURE SURVEY

## EtherRent: A Co-operative Car Rental Platform

This paper was written by Akhilesh Nair, Nelson Chacko, Rihab Kasim and Ani Sunny. This paper gives a detailed explanation on working of decentralized online car rental service. If a customer wants to rent a car, he/she can choose a car from all the available cars by looking at the features of the cars. The features of the cars can be obtained using sensors incorporated in the cars. This paper also explains how the drawbacks of centralized system can be overcame using decentralized systems. Even after the usage of vehicle by customer, the owner can get the condition of car from the sensors incorporated. This information is updated in the blockchain and incase of any damage certain penalty has to be paid by the customer. Third party involved in the centralized systems is removed in this platform and the profit goes directly to the owner of the car.

**Advantages:**

1. As there is no involvement of third party entire profit goes to users.
2. All actions are easily traceable since blockchain is being used here.

**Disadvantages:**

1. Sensors being vulnerable to attacks it may lack in security.
2. Data during transit may be vulnerable.

## Decentralized Applications: The Blockchain-Empowered Software System

This paper was written by WEI CAI, ZEHUA WANG, JASON B, ZHEN HONG, CHEN FENG AND VICTOR C. M. LEUNG 2. This paper was published in IEEE Access, 2018. This paper presents the development of blockchain systems to highlight the importance of decentralized applications and the future value of blockchain. In this paper, all the key elements of the blockchain systems were clearly discussed. This paper provides the information regarding the evolution of decentralized ledgers which led to classic blockchain systems. It also provides various characteristics of blockchain platform. This paper also compares different block chains and their implementations. This paper provides the details about the things that we should consider when selecting a blockchain implementation. Recent developments in blockchain systems were also summarized clearly.

**Advantages:**

1. Detailed Explanation on blockchain, decentralized applications, smart contracts, software systems.
2. History and future of blockchain and decentralized applications were discussed clearly.
3. This paper has presented application scenario of decentralization applications.

**Disadvantages:**

1. Payment channels are not much supportive for decentralized applications in future.

## Toward a Secure and Decentralized Blockchain-based Ride-Hailing Platform for Autonomous Vehicles

This paper was written by Ryan Shivers, Mohammad Ashiqur Rahman, and Hossain Shahriar. This paper presents a structure for developing a decentralized ride-hailing application. Hyperledger fabric blockchain was used as a platform to implement the ride-hailing application. Ride-hailing services help to fill the vacant seats in vehicle with the persons who are travelling to same place as the driver. In this paper, autonomous vehicles are used in this ride-hailing application as they collect all the information from sensors such as cameras, lasers and electromagnetic field detectors. This paper also provides a detailed information about blockchain, Hyperledger fabric and information security This paper clearly explains the chain code protocols and transactions security with their architecture. This paper describes good evaluation methods such as chain code analysis, constant rate network traffic, poison distribution based network traffic and Organization Restructuring Traffic.

**Advantages:**

1. Privacy protection for both riders and drivers.
2. It performs well even under large network load.

**Disadvantages:**

1. Latency is high in delayed transactions.
2. No peer to peer public key infrastructure

## Blockchain in Healthcare

This paper was written by Sourav Kumar, Utkarsh Kumar, Nalin Sanjay Singh and Mohamed Fathimal. This paper gives a detailed explanation on working of Blockchain in healthcare. It is mainly concerned on healthcare in handling the sharing of medical data electronically and medical records to patients, doctors and other staffs members. On entering the platform the user has to decide whether he/she wants to register as doctor or patient or staff by entering their name and selecting their role. Medical records to be verified by the doctor the patient first needs to provide permission to the doctor. Doctor can get patient records from their access list. After analyzing the health records of patient doctor can prescribe necessary medication and adds current medical report to the network. Patient can get their medical reports and can revoke permission at any time.

**Advantages:**

1. Provides data security and availability.
2. Doctors can get reliable records of their patients.

**Disadvantages:**

1. Data redundancy is high since same chunk of information is stored in multiple nodes.

## Blockchain for the Internet of Vehicles: A Decentralized IoT Solution for Vehicles Communication using Ethereum

This paper was written by Rateb Jabbar , Mohamed Kharbeche , Khalifa Al-Khalifa , Moez Krichen and Kamel Barkaoui . This paper was published in MPDI, 2020. We know that these days IOT is used in every sector. Internet of Vehicles (IOV) is one of the applications of IOT. This IOV shows an interaction between vehicles and their infrastructure. But IOV have certain limitations like lack of privacy, centralization etc. In this paper we can see how blockchain can address the problems caused by IOV .Blockchain technology has a ability to improve the transport systems by making them secure and autonomous. In this way, the resources of transport systems such as crowd sourcing technology can be used more effectively. A working model of the smart contract on the Testnet of Ethereum has been established. This study provides many solutions like availability, integrity, and security to test whether the Blockchain is an efficient and secure mechanism for IoV communications or not. This study is useful to solve the problems that are caused in IOV by using blockchain technology.

**Advantages:**

1. Detailed Explanation on blockchain and its applications.
2. Blockchain is an efficient and secure mechanism for IoV communications.

**Disadvantages:**

1. Since IoT being incorporated, there is always a chance of security breach at hardware side.

# FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

## SOFWARE REQUIREMENTS:

### 3.1.1 Remix ide:

Remix ide is an online smart contract developing environment for Decentralized applications. Remix provides one to deploy smart contracts in local remix environment, private blockchain network and in public blockchain network. It provides test ethers when deployed in local remix environment. It provides various plug-ins for ease in development, deployment and analysis.



Fig. 3.1: Remix ide

### 3.1.2 Visual Studio Code:

Visual Studio code is free code editing software used for developing entire application without using multiple editors. It was developed by Microsoft and contains wide range of extensions. One can add necessary extension from the Visual Studio market place to the software. Other features include auto code complete suggestion, syntax highlight are present in this software.

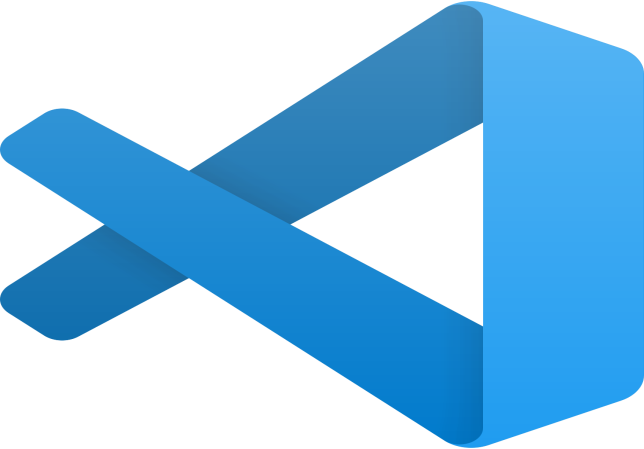


Fig. 3.2: Visual Studio Code

### 3.1.3 MetaMask:

MetaMask is a crypto currency wallet which is used to store Ethers. We can also interact with the Ethereum blockchain suing MetaMask. MetaMask comes as both mobile app and browser exension, which can then be used to interact with decentralized applications. MetaMask provides a facility to access different networks using same public address and even allow its users to work with private blockchain or local blockchain.

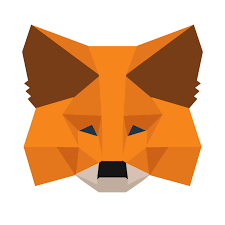


Fig. 3.3: MetaMask

# SOFTWARE DESIGN

## SOFTWARE MODEL:

The Water fall model is the most suitable development methodology to implement to this project. As all the requirements are known at the beginning of the project, water fall model is most flexible to implement this project. Below is the flow of water fall model.

Fig. 4.1: Water fall model

Requirement analysis phase involves collection of software requirements that are to be used to develop the model. System Design phase consists of designing the front end view of the application. Pages like Customer’s and driver’s login, signup, home pages etc. have to be designed in this phase. In implementation phase code required for Smart contract, frontend and integration is written using assigned languages. Frontend is developed using HTML and CSS, Smart contract is developed using solidity in Remix and Integration is done using Vanilla javascript. Testing phase involves testing the application using mocha and chai under different test cases. The model then can be deployed into the blockchain. Later in development phase any necessary changes are made for better working of the application.

## PROCESS FLOW DIAGARAM:

Process Flow diagram is a diagrammatic representation of sequence of logical steps of a program. Flowcharts use simple geometric shapes to depict processes and arrows to show relationships and process/data flow. Below is the process flow diagram of the project.

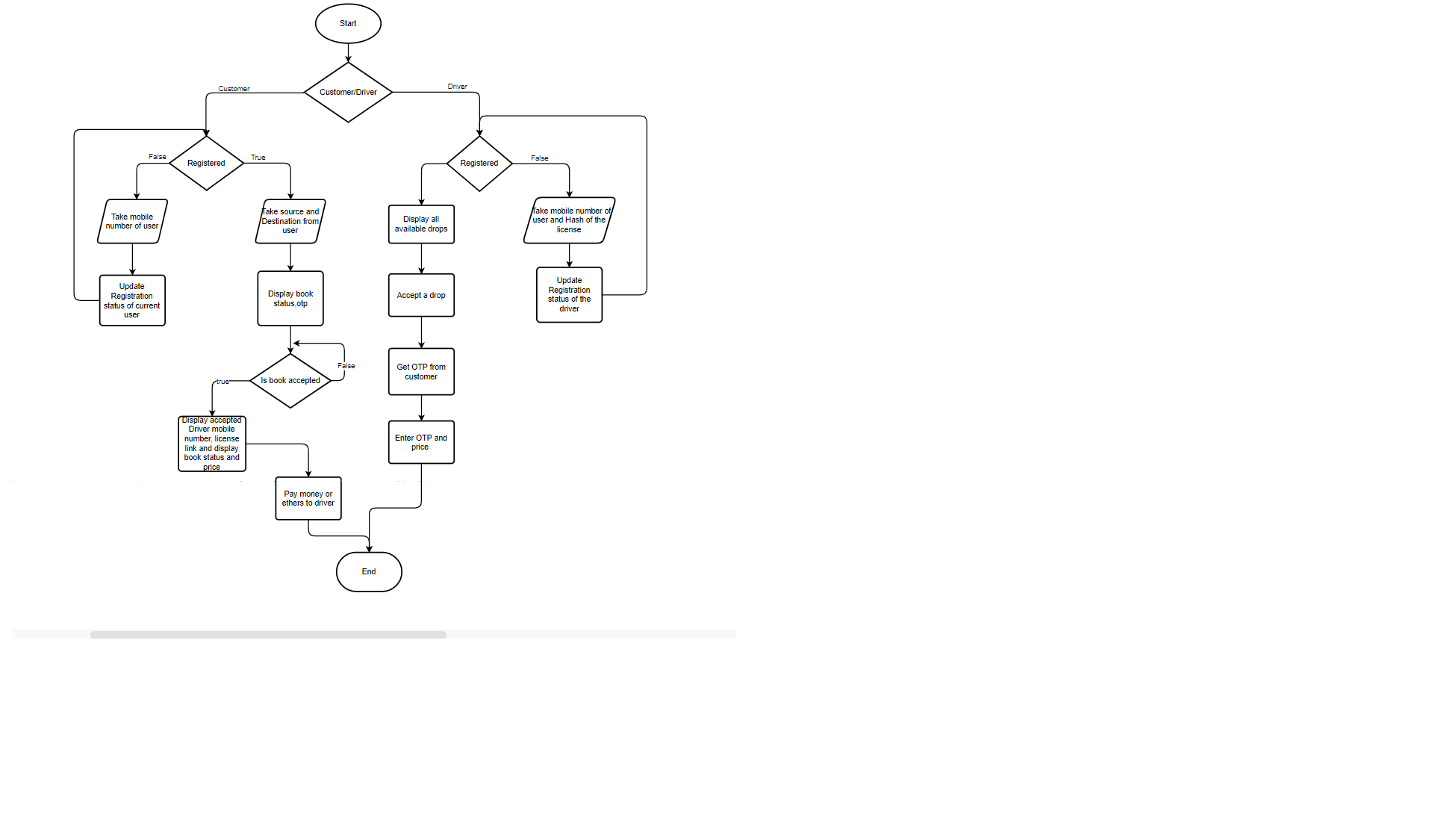
****

Fig. 4.2: Process Flow Diagram

## USE CASE DIAGRAM:

To model a system, the most important aspect is to capture the dynamic behavior. Dynamic behavior means the behavior of the system when it is running /operating. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Below is the use case diagram of this model.

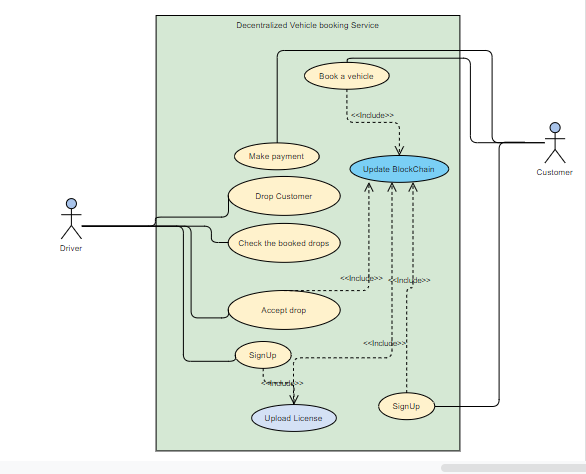
****

Fig. 4.3: Use Case Diagram

Actors of the above Use Case diagram are Driver and Customer. The major use cases including SignUp, booking a vehicle for drop, accept a drop, make payment and update blockchain are shown in the above diagram.

## SEQUENCE DIAGRAM:

Sequence diagram is the diagram in which main representation is of the sequence of messages flowing from one object to another. Main emphasis is on representing that how the messages/events are exchanged between objects and in what time-order. Below is the Sequence diagram of this model.

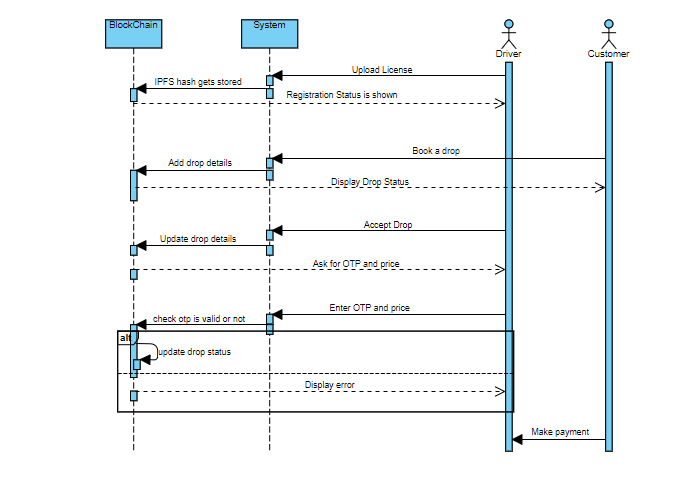
****

Fig. 4.4: Sequence Diagram

The actors in the above diagram are driver and customer. All the transactions and their corresponding results are shown in the above diagram.

# PROPOSED SYSTEM

## METHODOLOGY:

The proposed system involves creation of a smart contract, a front end, document uploading and an integration part.

**Smart contract:**

Smart contract can be developed in Remix IDE using solidity. The smart contract contains necessary code to perform required actions once an event is triggered. It is first tested using local remix blockchain and then deployed into Ethereum Ropstein Test network using Metamask. Once it is deployed the contract address is saved for future use.

**Front end:**

Front end of the application is created using HTML and the styling is done using CSS. It involves creation of HTML pages like Index page, SignUp pages of Customer and Driver, Homepages for Customer and Driver, Drop status pages and Document upload page.

**License Uploading:**

A page is created for the customer to upload their license. IPFS- InterPlanetary File System is used to upload and retrieve the document uploaded.

**Integration:**

Integration of front end with smart contract is done using vanilla javascript where every action is performed using promises.

On entering the platform the user has to decide whether he/she wants to register as driver or customer. If one wants to register as a customer one has to enter their mobile number else if one wants to register as driver one has to enter hash of their identity document uploaded in the IPFS network and their mobile number. If the driver didn’t uploaded their identity document he/she has to upload it and then enter the hash value. The customer can enter the source location and destination he/she wants to reach. An OTP is generated and shown to the customer.

All the drops entered by customers are displayed on home page of every driver. A driver can select a drop from the available drops. Once a drop is selected it cannot be selected by other driver or single driver cannot select more than one drop. Once a drop got accepted the driver details are updated at the customer side. The customer is provided with driver’s public address and license URL. Then driver can get OTP from customer and enter OTP along with price and on entering the drop status gets updated at customer side and customer has to pay charge to driver either as cash or ethers.

## ALGORITHM:

**At Driver’s side:**

Step 1: Check whether the user is registered as driver or not.

Step 2: If not, ask him/her to register as driver by uploading his/her identity proof such as license in blockchain.

Step 3: Display all available drops to the driver

Step 4: If driver accepts a drop update the details in blockchain and don’t allow him/her to take that drop until the current drop is completed.

Step 5: Get OTP from customer and enter OTP, price in the input fields given

**At Customer’s side:**

Step 1: Check whether the user is registered as customer or not.

Step 2: If not, ask him/her to register as customer by giving his mobile number.

Step 3: Take the source and destination if the customer needs a drop.

Step 4: Display the OTP of the drop and display the status of the drop (accepted driver id, link of the driver’s identity etc.) continuously.

# CODING

This chapter presents the functions or the code outline of the project and explains the corresponding functionality of each file.

## project.sol

The smart contract project.sol is developed using solidity in Remix IDE. This file contains necessary functions to perform actions when an event is triggered.

**Creation of structure to store blockchain data:**

struct person{

address add;

uint64 mobile;

bool reg;

uint8 lid;

mapping(uint8=>bool) allids;}

mapping(address=>person) allpersons;

**Function to get all unpicked drops:**

function all\_unpicked\_drops() public view returns(uint8[] memory ){

uint8[] memory allids=new uint8[](100);

uint8 z=0;

uint8 i=0;

for(i=1;i<=lastid;i++){

if(drops\_available[i].dropped==false){

if(z<100){

allids[z]=i;

z=z+1;}

else{

break;}}}

return allids;}

**Function to generate OTP of 4 digits:**

function generate\_otp\_for\_drop() public view returns (uint){

return uint(keccak256(abi.encodePacked(now,msg.sender,lastid)))%10000;

}

## home.html:

The HTML page home.html is used to display the sign in and signup options for driver and customer.

## DriverSignUp.html:

The HTML page DriverSignUp.html is used for driver to get registered into the network. The driver has to enter his number along with their license hash.

## DocumentUpload.html:

The HTML page DocumentUpload.html is used for driver to upload their license into IPFS.

**Creation of IPFS client object:**

const ipfsClient = require('ipfs-http-client')

const ipfs = new ipfsClient({ host: 'ipfs.infura.io', port: 5001, protocol: 'https' });

export default ipfs;

## DriverSignIn.html:

The HTML page DriverSignIn.html page displays all available drops if registered as driver. The driver first needs to accept a drop then need to get OTP from corresponding customer and enter it along with the price.

## Cust\_SignUp.html:

The HTML page Cust\_SignUp.html page is used for customer to get registered into network. The customer has to enter his/her mobile number to get registered.

## UserSignIn.html:

The HTML page UserSignIn.html is used for the customers to enter the source and destination they want to reach to place a drop request on to the network.

## Booked.html:

The HTML page Booked.html is used to show the status of their recent drop request at the customer side. It shows the drop id, drop accepted status, Ethereum public key and the license URL of the driver.

## CommonScript.js:

The javascript page CommonScript.js contains code to integrate the front end with the smart contract.

**Importing and loading web3 package:**

<script src="node\_modules/web3/dist/web3.min.js" type="text/jsx"></script>

async function loadWeb3() {

if (window.ethereum) {

window.web3 = new Web3(window.ethereum);

window.ethereum.enable();}}

**Importing sol file using ABI and contract address:**

await new window.web3.eth.Contract(ABI, contractAddress);

**Function to get current driver status:**

async function getAccStatus(){

const acc=await getCurrentAccount();

const ans=await window.contract.methods.is\_driver\_free(acc).call();

return ans;}

**Function to add a drop to blockchain:**

async function adddrop() {

var a=document.getElementById("source").value;

var b=document.getElementById("dest").value;

const account = await getCurrentAccount();

const drop\_id = await window.contract.methods.add\_drop(a,b).send({ from: account });

}

**Function to get all unpicked drops:**

async function getall\_undropped(){

const undropped= await window.contract.methods.all\_unpicked\_drops().call();

return undropped;

}

# TESTING

Testing helps to identify defects or errors that could possibly be made during the development phase. Without a well-thought testing effort, the project will undoubtedly fail overall and will impact the entire operational performance of the solution. With a poorly tested solution, the support and maintenance cost will escalate exponentially, and the reliability of the solution will be poor.

# RESULT

Whenever any user enters the network, he/she has to register either as a customer or as a driver. On entering the user can see a screen as below.



Fig. 8.1: Home Page

If the person is not registered as customer but still wants to access customer home page a message as shown below appears.

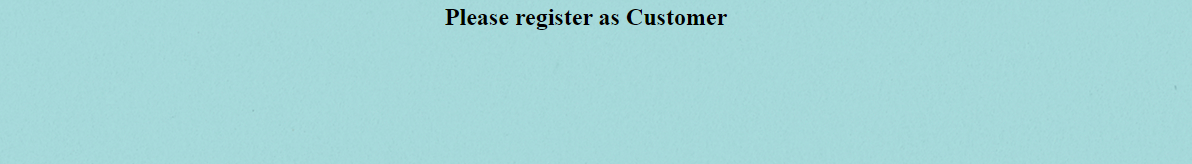


Fig. 8.2 Error message at customer’s side

If the person wants to enter the network as customer, he/she can choose corresponding option and enter the mobile number. The below image represents the customer’s signup page.

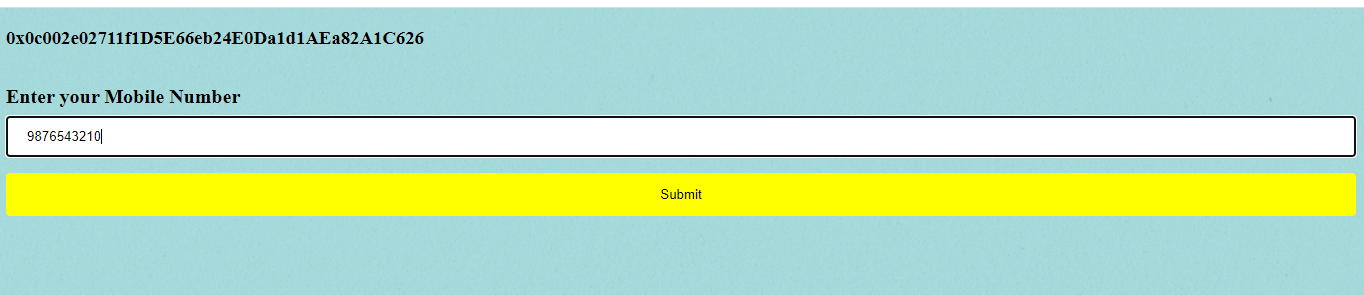
****

Fig. 8.3: Customer SignUp page

If the person is not registered as driver but still wants to access driver home page a message as shown below appears.

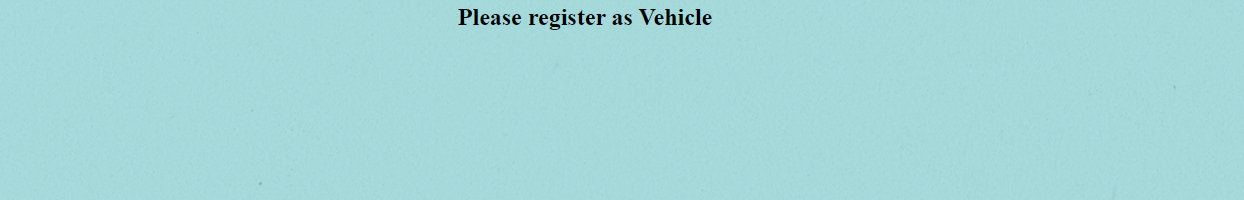


Fig. 8.4 Error message at driver’s side

If the person wants to enter network as driver, he/she can choose corresponding option and enter mobile number and hash value of their identity. The below image represents the driver’s signup page.



Fig. 8.4: Driver SignUp page

If the user never uploaded their license they can upload using the link provided. The below image represents the license uploading page.

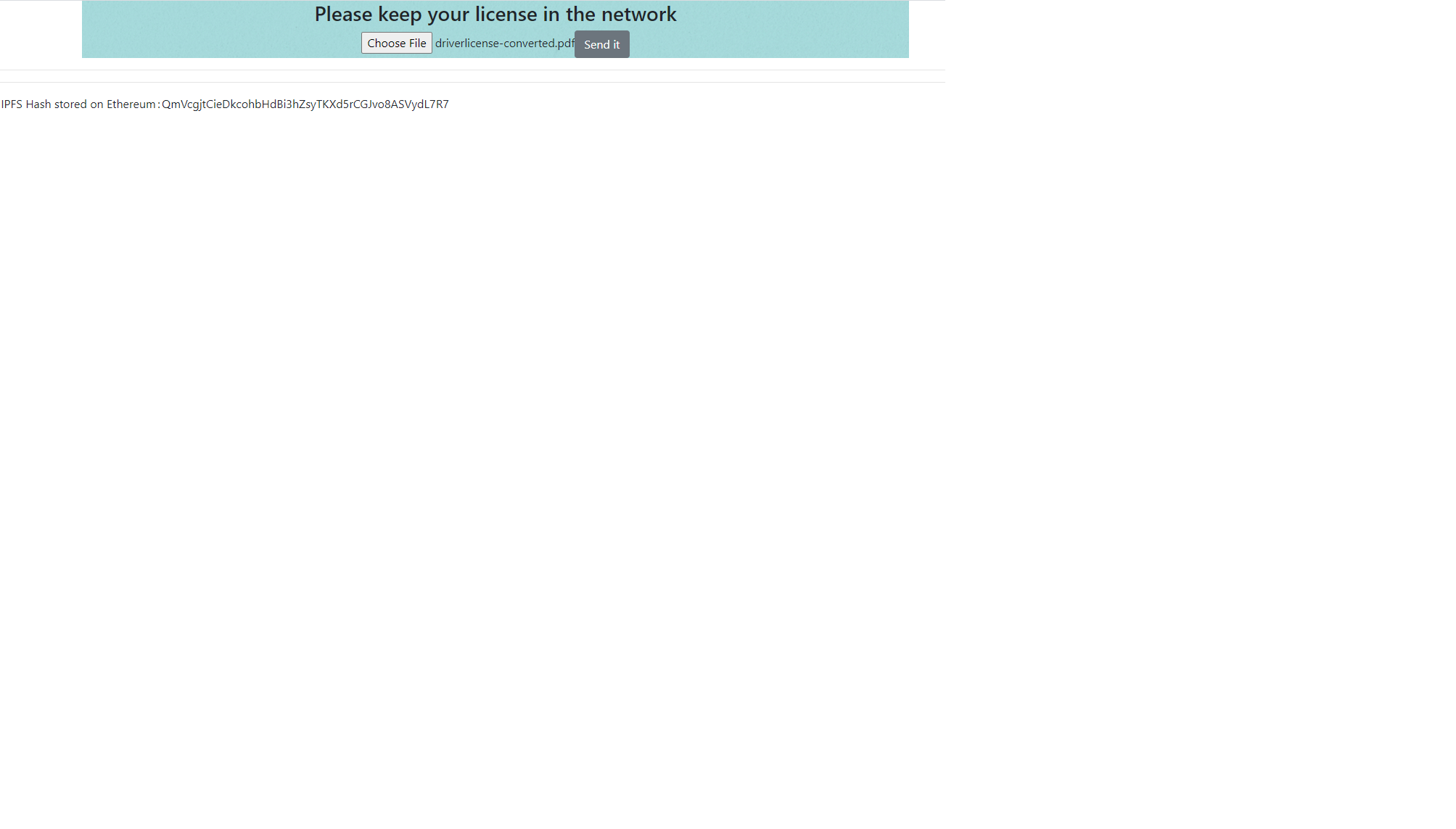
****

Fig. 8.5: License uploading page

The registered customer can book a drop by entering source and destination. The below image shows home page of customer.



Fig. 8.6: Customer home page

On clicking on submit it will be redirected to a status page, which shows the status of current drop. The below image shows the status page of the drop when no one accepted the drop.



Fig. 6.6: Status page 1

On driver’s side the screen containing all available drops are displayed he/she can select one drop from all the available drops and accept it. The home screen of the driver is shown below.



Fig. 6.7: Driver home page

Once the drive got accepted by someone the status changes and is shown in the figure below.

Fig. 6.8: Status page 2

The driver has to get OTP from customer and enter it along with the price. The below screen represents the drop taking page.

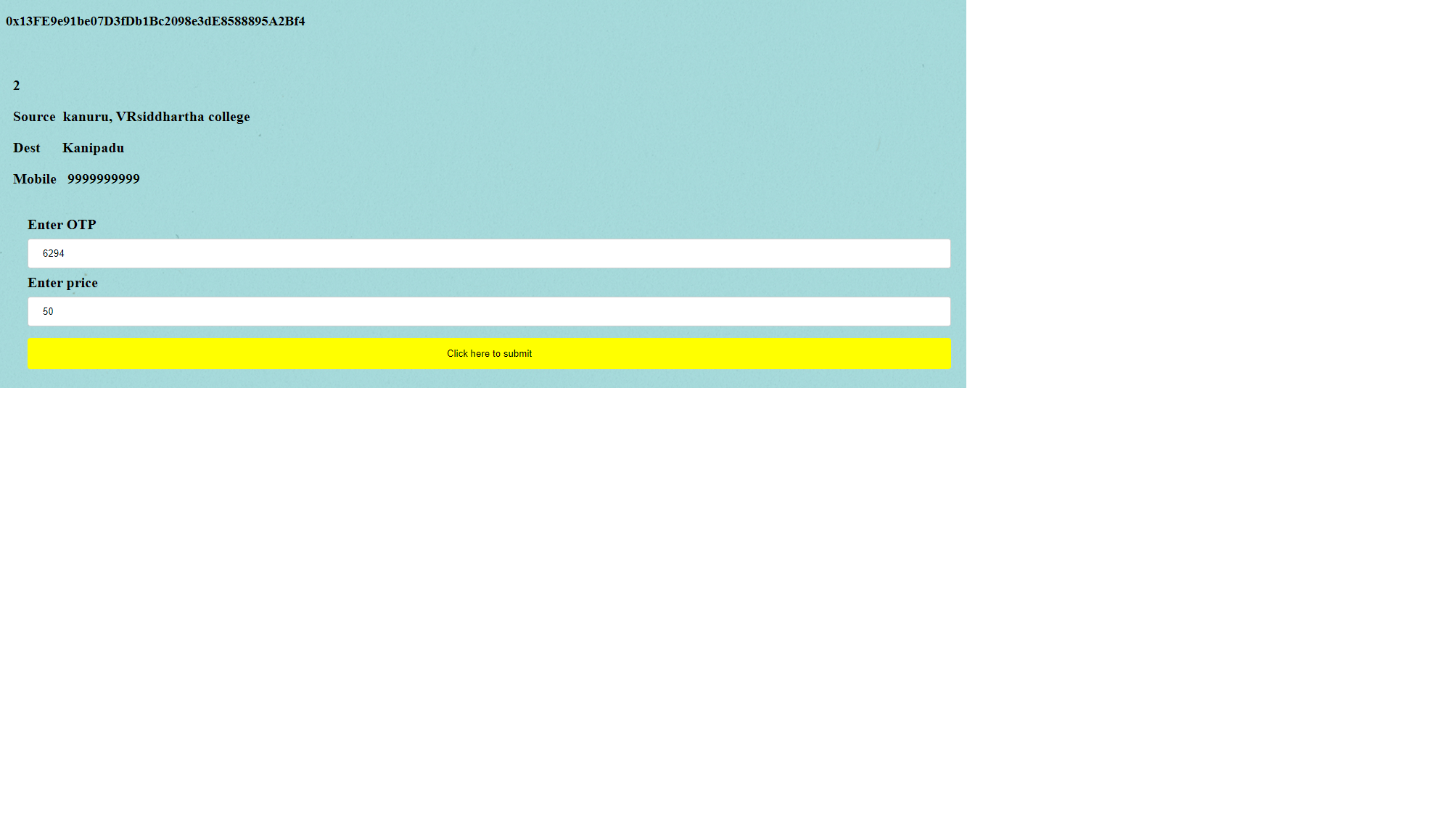


Fig. 6.9: Drop Taking page

Once the driver enters otp and price the drop gets completed and the status is updated at the customer’s side.



Fig. 6.10: Status page 3

# CONCLUSION AND FUTURE WORK

Through this project, we are aiming to overcome the major pitfalls of online vehicle booking services being in the hands of centralized authorities. Our decentralized platform avoids a third party getting profits from our transactions. Trustlessness, the major issue of decentralized platforms has overcame by uploading driver’s documents to block chain and allowing users to view the documents using theirs hash value. Traceability being one of the important features of decentralized applications will help us keep track of all the actions done in the network. The major issue is time consumption for every transaction. Decentralized applications take more time than that of centralized ones and that cannot be avoided.

This work can be improved by incorporating a bidding system to accept the drop instead of first come first serve sort of drop selection by the driver. The drivers can place their bid for a drop and the customer can choose appropriate bid from the available bids. This model can also be improved by incorporating automatic token or ether transfer from customer’s account to driver’s account after the drop is done. Drop accepted driver’s location can also be shared to customer to track the time taken for driver to reach customer.

# REFERENCES

1. Akhilesh Nair, Nelson Chacko, Rihab Kasim and Ani Sunny, “EtherRent: A Co-operative Car Rental Platform”, International Research Journal of Engineering and Technology (IRJET), volume 7, issue 4, April 2020.
2. T. Mitani and A. Otsuka, "Traceability in Permissioned Blockchain," 2019 IEEE International Conference on Blockchain, pp. 286-293, 2019.
3. Wang S, Ding W, Li J Yuan, Ouyang L and Wang F, “Decentralized autonomous organizations: concept, model, and applications”, IEEE Transactions on Computational Social Systems 6(5), pp. 870–878, 2019.
4. E. Nyaletey, R. M. Parizi, Q. Zhang and K. R. Choo, "BlockIPFS - Blockchain-Enabled Interplanetary File System for Forensic and Trusted Data Traceability," 2019 IEEE International Conference on Blockchain, pp. 18-25, 2019.
5. Roberto Tonelli, Andrea Pinna, Gavina Baralla and Simona Ibba, “Ethereum smart contracts as blockchain-oriented microservices”, IEEE 19th International Conference on Agile Software Development Companion, pp. 10-1145, 2018.
6. W. Cai, Z. Wang, J. B. Ernst, Z. Hong, C. Feng and V. C. M. Leung, "Decentralized Applications: The Blockchain-Empowered Software System," in IEEE Access, vol. 6, pp. 53019-53033, 2018, doi: 10.1109/ACCESS.2018.2870644.
7. Hofmann, F. Wurster, S. Ron and Bohmecke-Schwafert, “The immutability concept of blockchains and beneﬁts of early standardization”, IEEE ITU Kaleidoscope Challenges for a Data-Driven Society (ITU K), pp. 1-8, 2017.
8. Simone Porru, Andrea Pinna, Michele Marchesi, and Roberto Tonelli, “Blockchain-oriented software engineering: challenges and new directions”, IEEE 39th International Conference on Software Engineering Companion, pp. 169-171, 2017.
9. Davide Taibi, Valentina Lenarduzzi, and Claus Pahl, “Processes, Motivations, and Issues for Migrating to Microservices Architectures: An Empirical Investigation”, IEEE Cloud Computing 4, pp. 22-32, 2017.
10. Katiuscia Mannaro, Andrea Pinna, and Michele Marchesi, “Crypto-trading: Blockchain-oriented energy market”, IEEE AEIT International Annual Conference, pp. 1-5, 2017.

[11] Christidis K and Devetsikiotis M, “Blockchains and Smart Contracts for the Internet of Things”, IEEE Access 4, pp. 2292–2303, 2016.