Smart Park - Parking Management with Intelligent License Plate Recognition

Nithish Kumar Saravanan
1226 120 885

Problem Statement

Urban parking is often **inconvenient, congested, and inefficient**. Users struggle to find parking spaces, make payments, and have a smooth experience. Manual ticketing and payment systems are slow and error-prone, lacking technological efficiency.

What I Am Trying to Solve:

License Plate Recognition: Eliminate the need for physical tickets and enhance security.

Parking Space Allocation: Intelligently allocating available parking spaces to vehicles, ensuring efficient space utilization.

Hands-Free Payment (Future scope): Automatically charges users based on their parking duration.

System Components

License Plate Recognition (LPR) System: Identifying and recognizing the alphanumeric characters on the plates.

Camera System: To capture images of vehicles and their license plates as they enter the parking area.

Database: Stores information about registered users, their license plate data, parking space assignments, and payment details.

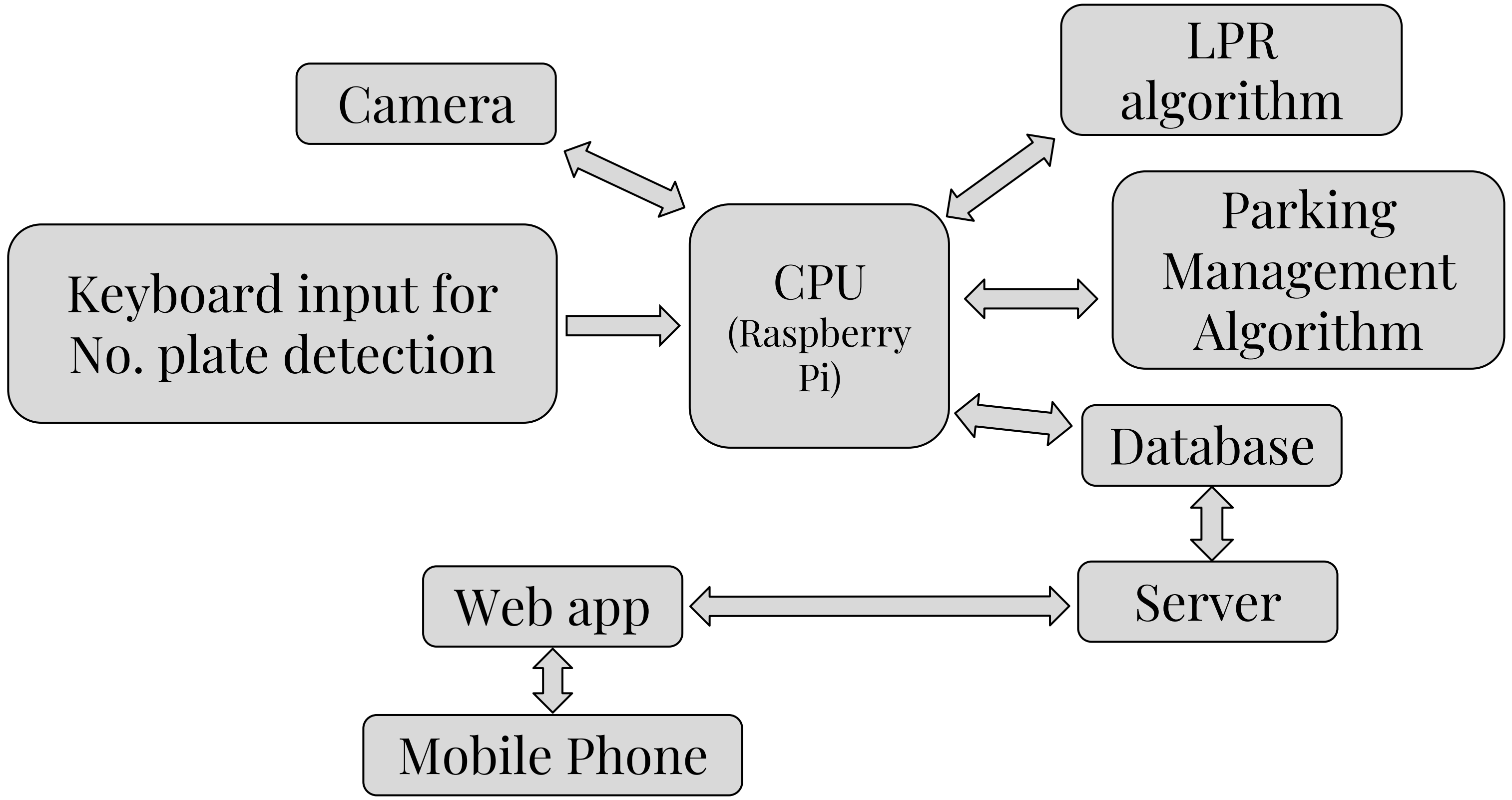
User Interface: Web application allows users to register their vehicles, view parking availability, make reservations.

Server: Hosts the web application

Parking Space Management: Optimizes parking space allocation.

Central Processing Unit: Processes the data received from cameras, manages the recognition of license plates, allocates parking spaces.

Interactions between Components



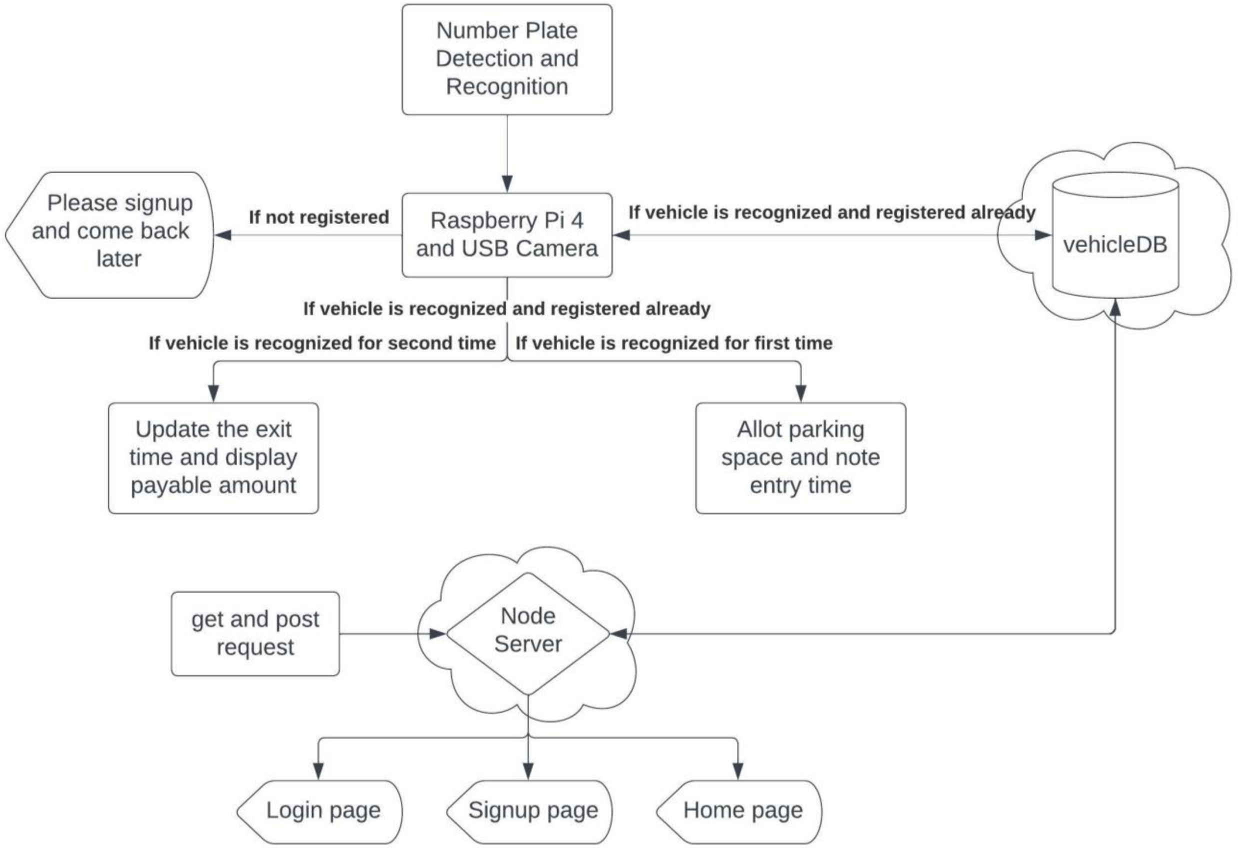
Tradeoffs in the Design

- This project only focuses on **Entry** and **Exit management** System. Only one camera setup will be used for both the systems.
- Parking Space Management algorithms allocates **random space** based on the free space data in database. In future, either sensors will be used or Computer vision algorithms will be used to detect free space.
- Presence of card credentials of the user are **only checked** and **payment APIs** are **not used** for transaction.

Tech Stacks

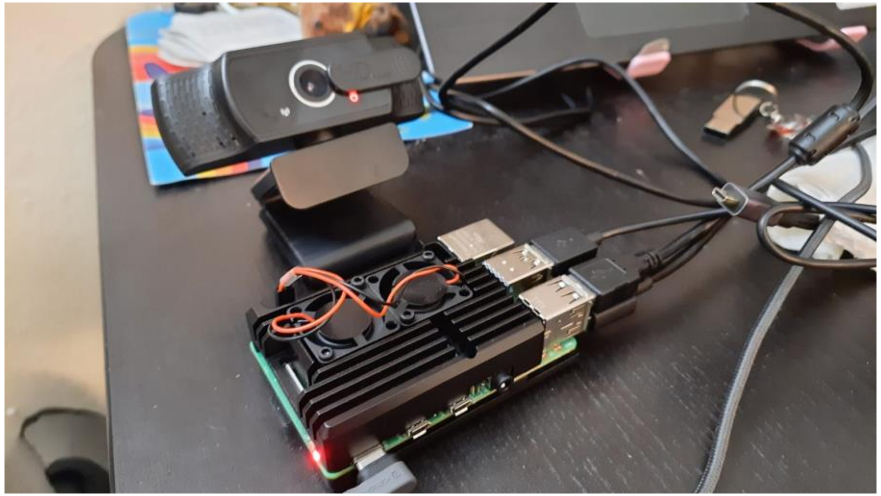
- **Web application:** HTML, CSS, JavaScript, EJS
- **Database Management System:** MongoDB Atlas
- **Server side script:** Node.js
- **License Plate Recognition (LPR)algorithm:** Haar cascade classifier, EasyOCR
- **Atlas CRUD operations:** Python, Node.js
- **Cloud Service:** AWS
- **LPR Deployment:** Raspberry Pi 4

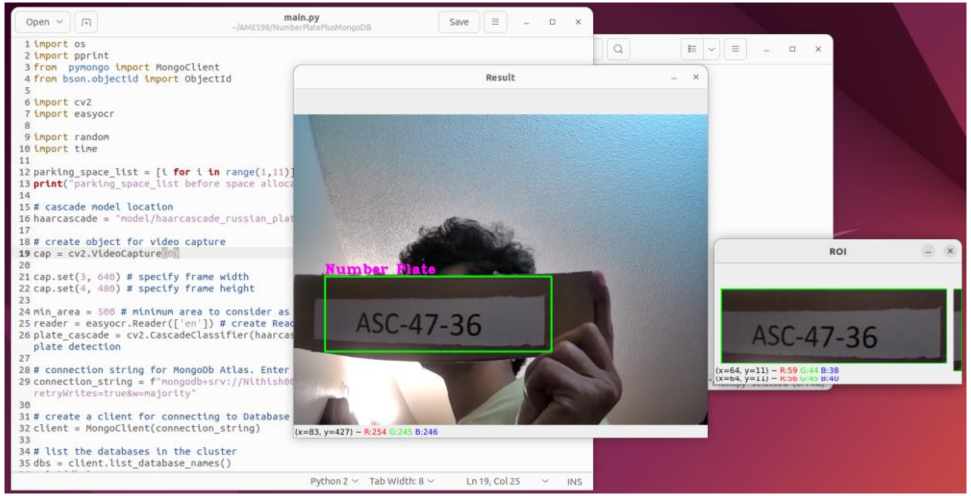
Process Flow


Number Plate Detection and Recognition
Please signup and come back later
If not registered
Raspberry Pi 4 and USB Camera
If vehicle is recognized and registered already
vehicleDB
If vehicle is recognized and registered already
If vehicle is recognized for second time
If vehicle is recognized for first time
Update the exit time and display payable amount
Allot parking space and note entry time
get and post request
Node Server
Login page
Signup page
Home page

License Plate Recognition

- Number Plate Detection – Haar cascade classifier
- Number Plate Recognition - EasyOCR

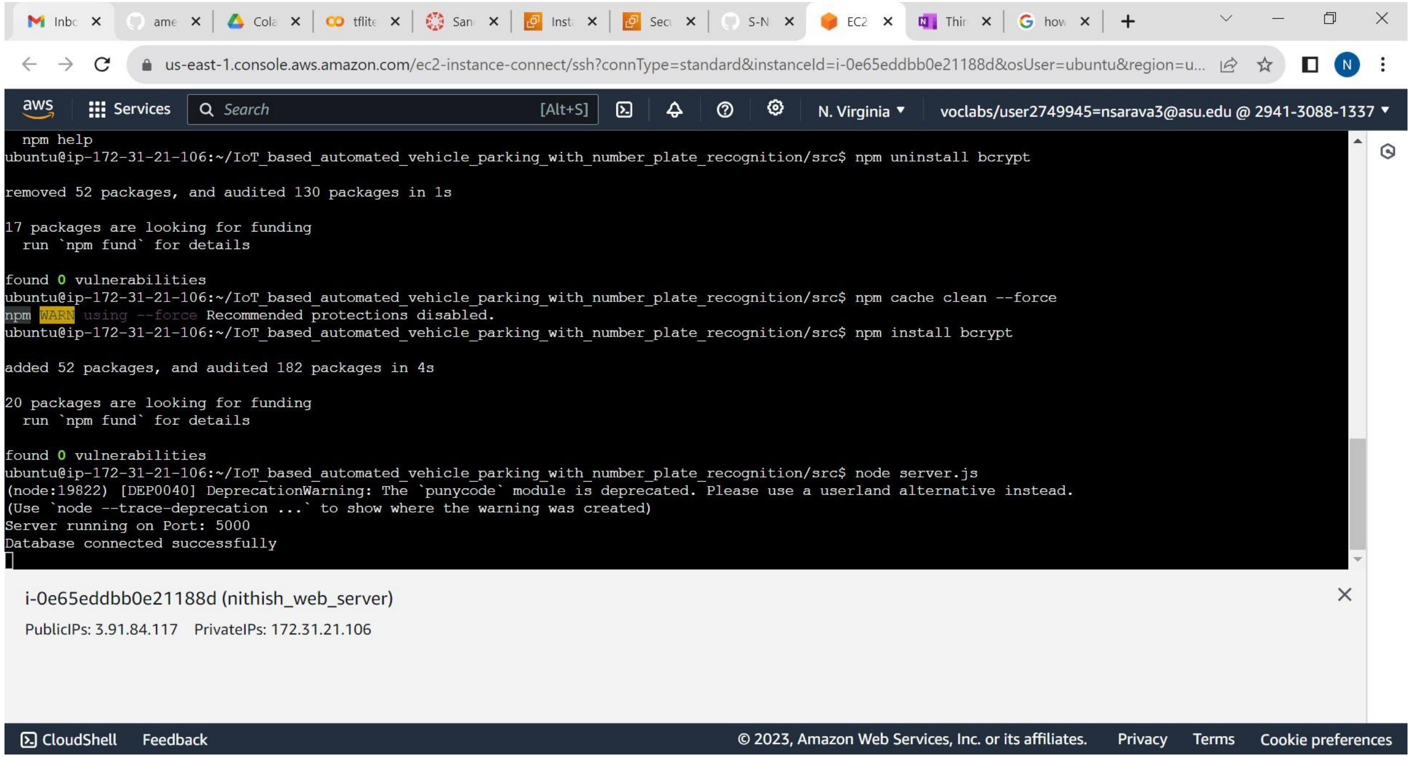

Raspberry Pi Setup



```python
import os
import pprint
from  pymongo import MongoClient
from bson.objectid import ObjectId

import cv2
import easyocr

import random
import time

parking_space_list = [i for i in range(1,11)]
print("parking_space_list before space alloca

# cascade model location
haarcascade = "model/haarcascade_russian_plat

# create object for video capture
cap = cv2.VideoCapture(0)

cap.set(3, 640) # specify frame width
cap.set(4, 480) # specify frame height

min_area = 500 # minimum area to consider as
reader = easyocr.Reader(['en']) # create Reac
plate_cascade = cv2.CascadeClassifier(haarcas
plate detection

# connection string for MongoDb Atlas. Enter
connection_string = f"mongodb+srv://Nithish06
retryWrites=true&w=majority"

# create a client for connecting to Database
client = MongoClient(connection_string)

# list the databases in the cluster
dbs = client.list_database_names()
```

Number Plate Detection and Recognition

Hosting Node Sever in AWS


Inbo
ame
Cola
tflite
San
Inst
Sec
S-N
EC2
Thir
how
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i-0e65eddbb0e21188d&osUser=ubuntu&region=u...
aws
Services
Search
[Alt+S]
N. Virginia
voclabs/user2749945=nsarava3@asu.edu @ 2941-3088-1337

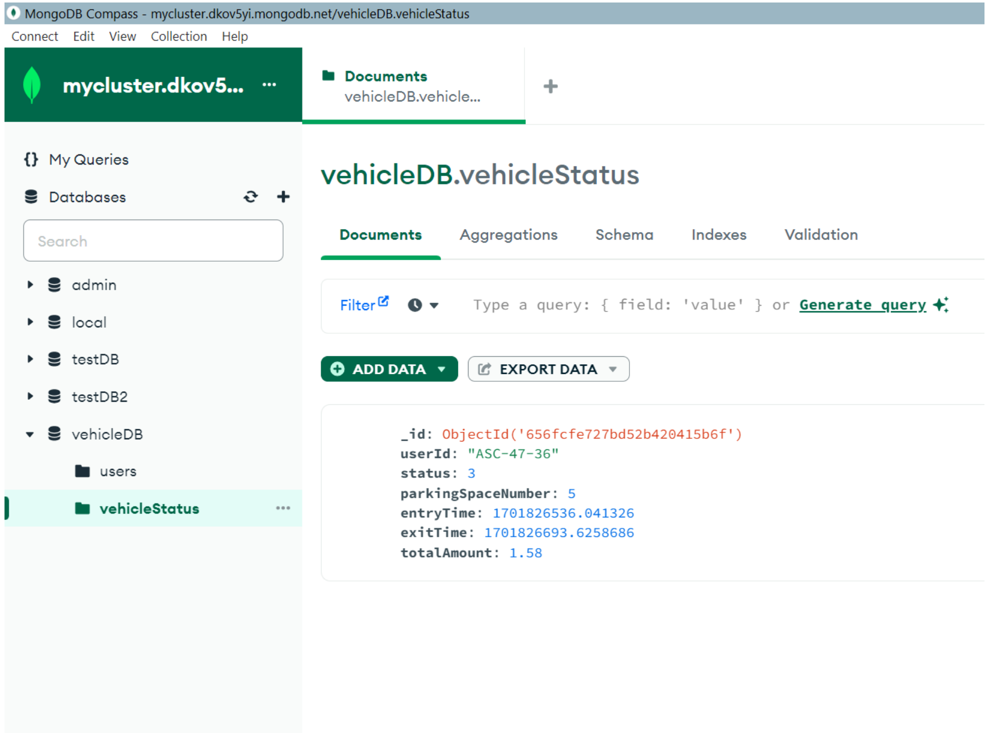
```
  npm help
ubuntu@ip-172-31-21-106:~/IoT_based_automated_vehicle_parking_with_number_plate_recognition/src$ npm uninstall bcrypt

removed 52 packages, and audited 130 packages in 1s

17 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
ubuntu@ip-172-31-21-106:~/IoT_based_automated_vehicle_parking_with_number_plate_recognition/src$ npm cache clean --force
npm WARN using --force Recommended protections disabled.
ubuntu@ip-172-31-21-106:~/IoT_based_automated_vehicle_parking_with_number_plate_recognition/src$ npm install bcrypt

added 52 packages, and audited 182 packages in 4s

20 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
ubuntu@ip-172-31-21-106:~/IoT_based_automated_vehicle_parking_with_number_plate_recognition/src$ node server.js
(node:19822) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
Server running on Port: 5000
Database connected successfully
```

i-0e65eddbb0e21188d (nithish_web_server)

PublicIPs: 3.91.84.117   PrivateIPs: 172.31.21.106

Database

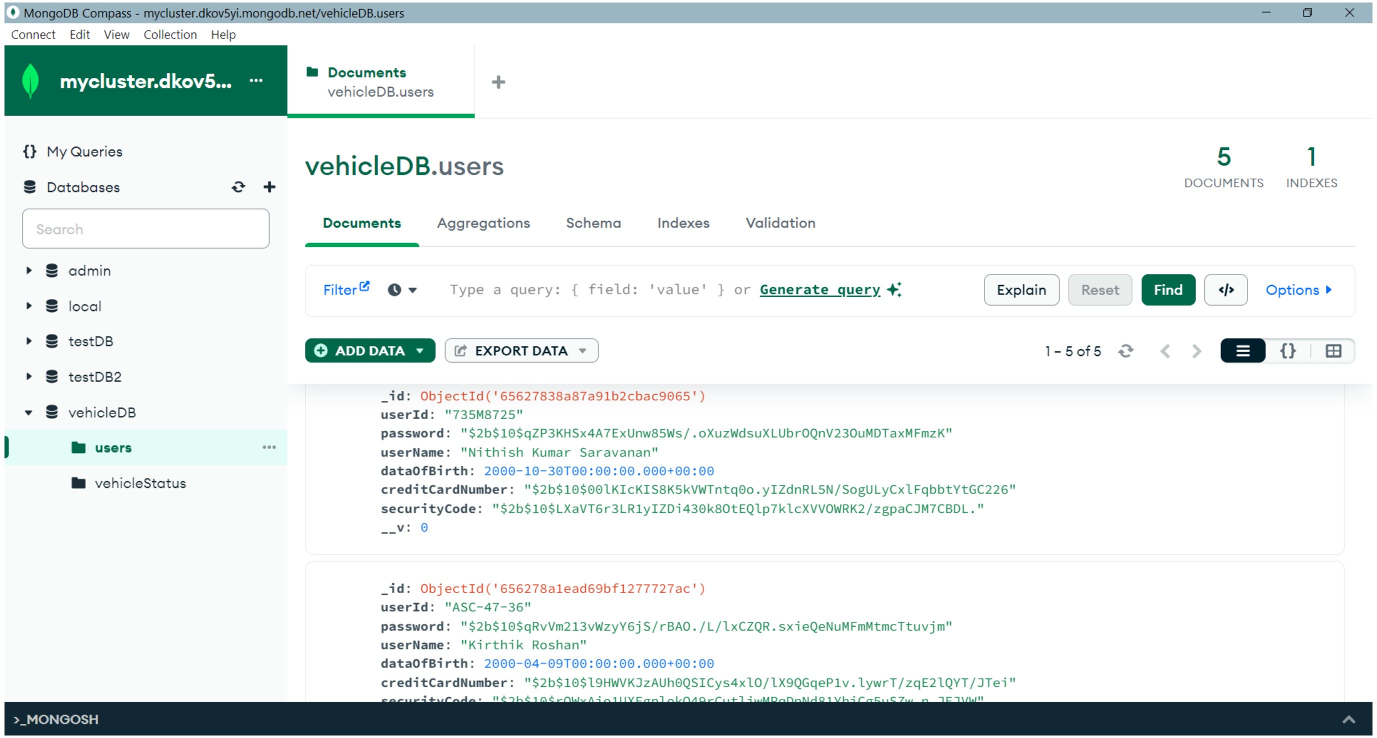
- Database - MongoDB Atlas
- Database Name – VehicleDB
- Collections – users, vehicleStatus
  - Users collection – Contains sign up details of the users.
  - vehicleStatus – Gets updated based on vehicle entry and exit.



```
_id: ObjectId('656fcfe727bd52b420415b6f')
userId: "ASC-47-36"
status: 3
parkingSpaceNumber: 5
entryTime: 1701826536.041326
exitTime: 1701826693.6258686
totalAmount: 1.58
```

vehicleStatus collection – updated after a vehicle exit

Database



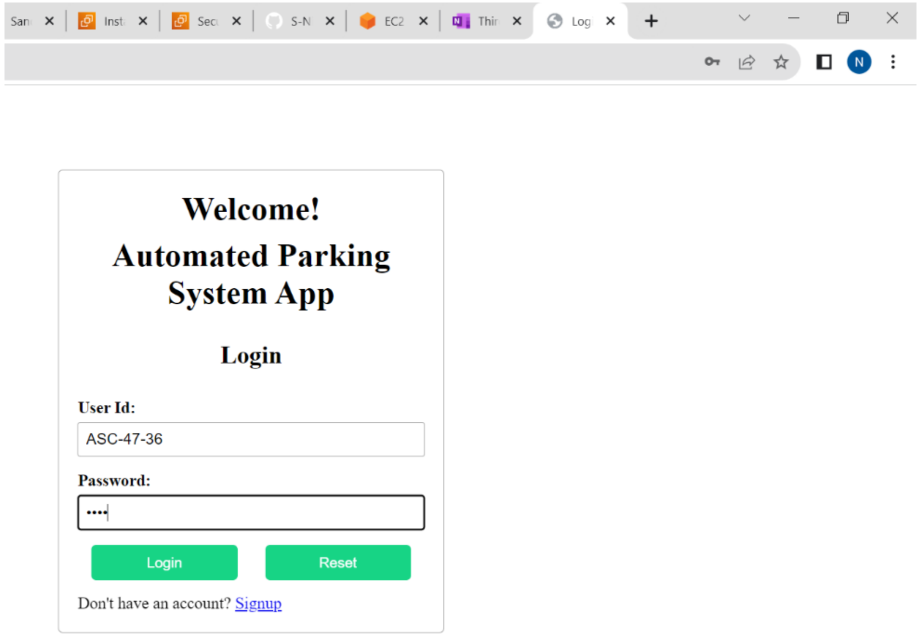
MongoDB Compass - mycluster.dkov5yi.mongodb.net/vehicleDB.users
Connect Edit View Collection Help
mycluster.dkov5...
Documents
vehicleDB.users
My Queries
Databases
Search
admin
local
testDB
testDB2
vehicleDB
users
vehicleStatus
vehicleDB.users
5 DOCUMENTS
1 INDEXES
Documents Aggregations Schema Indexes Validation
Filter
Type a query: { field: 'value' } or Generate query
Explain Reset Find Options
ADD DATA
EXPORT DATA
1 – 5 of 5
>_MONGOSH

```
_id: ObjectId('65627838a87a91b2cbac9065')
userId: "735M8725"
password: "$2b$10$qZP3KHSx4A7ExUnw85Ws/.oXuzWdsuXLUbrOQnV23OuMDTaxMFmzK"
userName: "Nithish Kumar Saravanan"
dataOfBirth: 2000-10-30T00:00:00.000+00:00
creditCardNumber: "$2b$10$00lKIcKIS8K5kVWTntq0o.yIZdnRL5N/SogULyCxlFqbbtYtGC226"
securityCode: "$2b$10$LXaVT6r3LR1yIZDi430k8OtEQlp7klcXVVOWRK2/zgpaCJM7CBDL."
__v: 0
```

```
_id: ObjectId('656278a1ead69bf1277727ac')
userId: "ASC-47-36"
password: "$2b$10$qRvVm213vWzyY6jS/rBAO./L/lxCZQR.sxieQeNuMFmMtmcTtuvjm"
userName: "Kirthik Roshan"
dataOfBirth: 2000-04-09T00:00:00.000+00:00
creditCardNumber: "$2b$10$l9HWVKJzAUh0QSICys4xlO/lX9QGqeP1v.lywrT/zqE2lQYT/JTei"
```

users collection – updated after a users sign up

User Interface


Welcome!

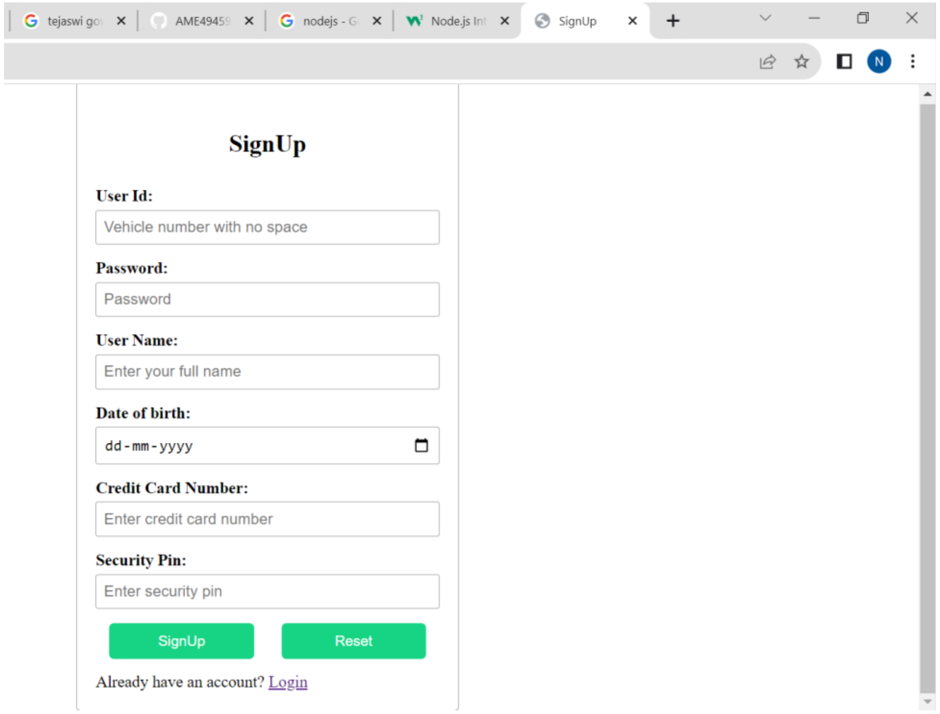
Automated Parking System App

Login

User Id: ASC-47-36

Password: ••••

Login Page


SignUp

User Id: Vehicle number with no space

Password: Password

User Name: Enter your full name

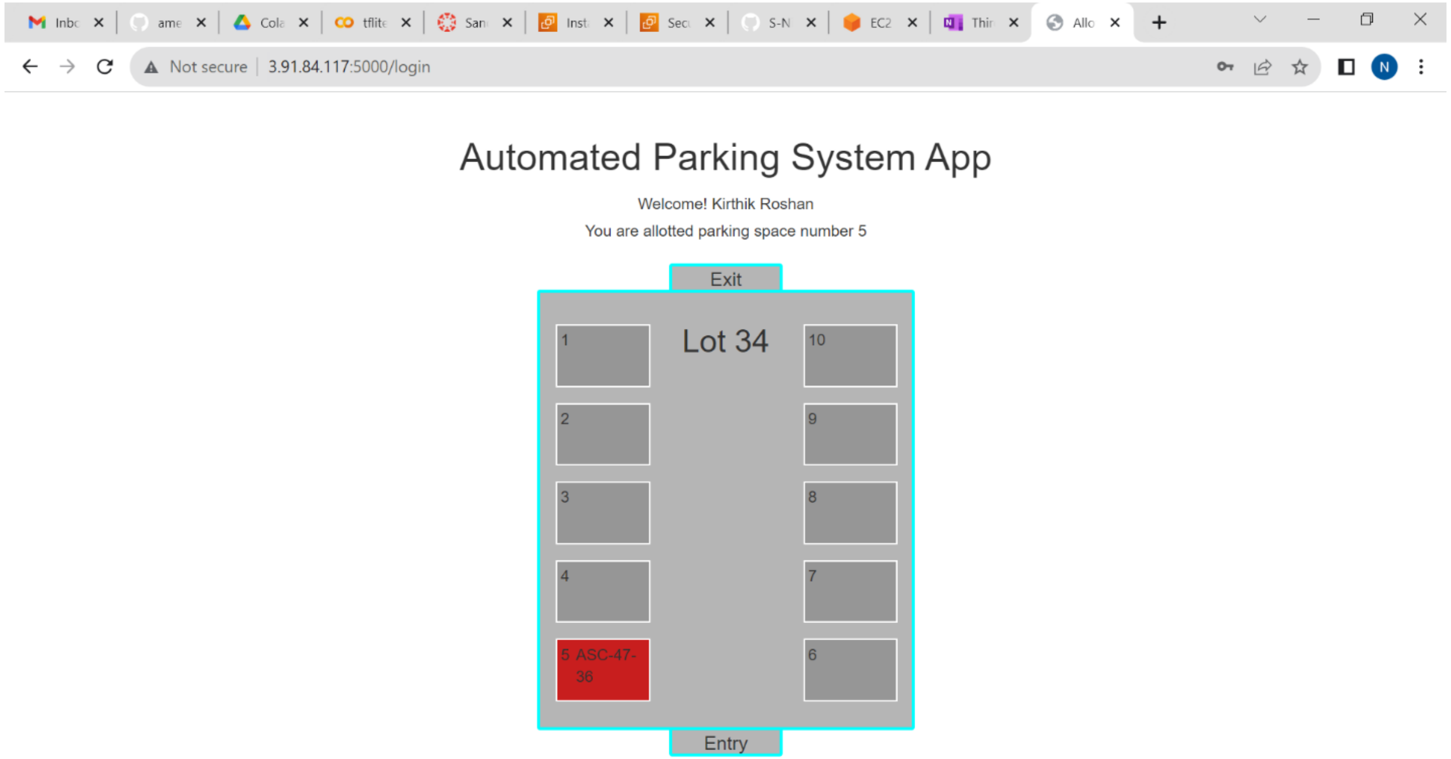
Date of birth: dd-mm-yyyy

Credit Card Number: Enter credit card number

Security Pin: Enter security pin

Signup Page

User Interface


Not secure | 3.91.84.117:5000/login

Automated Parking System App

Welcome! Kirthik Roshan

You are allotted parking space number 5

Home Page for Displaying Parking Lot Information

Design Gap

1. The User Interface is not responsive for different devices.
2. Right now, I am using time module in python for finding the timestamp for entry and exit. This is not accurate as the DateTime module especially when run time of the program goes beyond one hour.
3. I planned to use an ultrasonic sensor to detect the presence of vehicle. But due to the unavailability, I am just using the key press of the button ‘r’ on the keyboard to start the detection and recognition algorithm.
4. I was planning to use a Servo motor to mimic the opening and closing of gate at entry and exit but due to limited time I have skipped that part.

Achieved Goals

1. Deployment of LPR algorithm in Raspberry Pi.
2. Implementation of detection and recognition of simple number plates.
3. Accessing Database using two different drivers – python, node.js.
4. Deployment of node server in AWS.
5. Development of UI for displaying parking lot information.

Future Works

1. Make the UI responsive.

2. Incorporate money transfer API.

3. Add an Ultrasonic sensor to the system for detecting the presence of a vehicle.

4. Implement free parking space detection algorithm using Machine Learning model.

Updated Timeline

Start date - 10/27/2023

Week 1 - Research on License Plate Recognition algorithm

Week 2- Implement LPR algorithm and basic CRUD operations in Atlas

Week 3, 4 - UI and User authentication System development

Week 5 - System Integration and testing

End date - 12/04/2023

References

- https://ejs.co/
- https://www.mongodb.com/languages/python
- https://www.mongodb.com/docs/drivers/node/current/quick-start/connect-to-mongodb/
- https://www.w3schools.com/
- https://www.youtube.com/watch?v=O5kh3sTVSvA&list=WL&index=4
- https://youtu.be/ouaSi8v5CHQ?si=qqC_NH3DNHsixAoU
- https://github.com/Maleehak/Car-number-plate-recognition-using-OpenCV