# Project Report: Data-Driven Cab Service Optimization for Maximizing Profitability

Student Name: Smital Kamdi
MBA Program, Leavey School of Business
ISBA 2403-MW: Database Management Systems
Professor Dr. Nihal K. Sahan

February 8, 2025

**Overview and Objectives**

The Cab Service Optimization System is a Flask-based web application integrated with a MySQL database. The primary goal of this project is to analyse driver performance, service demand, and revenue while providing features to:

- Search for drivers by name or zone.
- Filter bookings by selecting a start and end date.
- View the list of highest-paid drivers based on earnings.
- Analyse driver acceptance performance to determine service preferences.
- Provide a dashboard with insights into revenue, bookings, and acceptance rates.

Key Functionalities

1. Search for drivers (by name or location zone).
2. Filter bookings (by date range).
3. View highest-paid drivers.
4. Analyse driver acceptance rate.
5. Dashboard displaying revenue, total bookings, and driver performance.

**Database Design**

Schema Overview

The database consists of four primary tables along with views for data analysis.

| **Table Name** | **Description** |
|---|---|
| Drivers | Store details of drivers, including their availability and assigned zones. |
| Services | Defines different service types (Ride, Food Delivery, Grocery Delivery) and pricing. |
| Bookings | Stores ride history, fare details, driver ID, and customer rating. |
| Driver_Acceptance | Tracks how often drivers accept service requests for different services. |

**Entity Relationship Diagram**

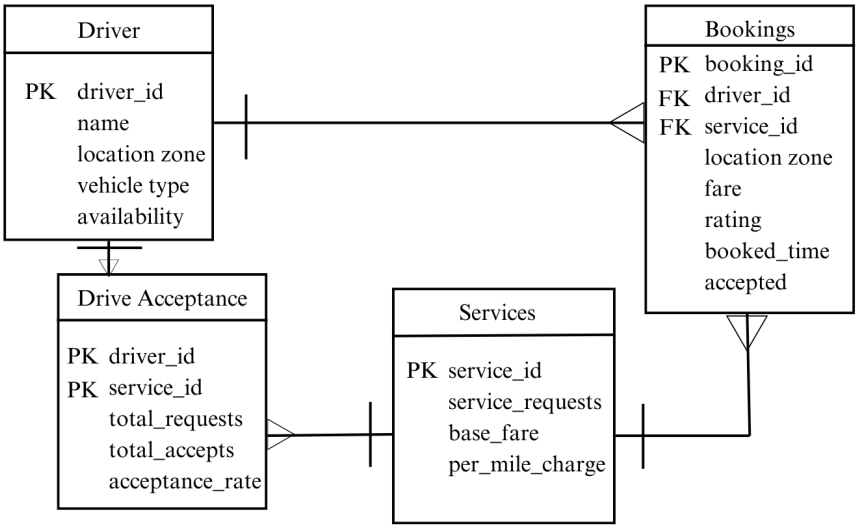



**Table and Attributes**

1. Drivers Table

Stores information about drivers, including their availability, location, and vehicle type.

| Column Name | Data Type | Description |
|---|---|---|
| driver_id | INT (PK) | Unique identifier for each driver |
| name | VARCHAR | Name of the driver |
| location_zone | VARCHAR | The zone where the driver operates (Zone 1-7) |
| vehicle_type | VARCHAR | Type of vehicle (Car, Bike, Van) |
| availability | BOOLEAN | Indicates if the driver is available for booking |

Relationships

- Primary Key: driver_id
- Referenced in: Bookings (via driver_id as FK), Driver_Acceptance (via driver_id as FK)

2. Services Table

Defines the types of services offered by the cab system and their pricing model.

| Column Name | Data Type | Description |
|---|---|---|
| service_id | INT (PK) | Unique identifier for each service |
| service_type | VARCHAR | Type of service (Ride, Food, Grocery) |
| base_fare | DECIMAL | Fixed starting fare |
| per_mile_charge | DECIMAL | Cost per mile |

Relationships

- Primary Key: service_id
- Referenced in: Bookings (via service_id as FK), Driver_Acceptance (via service_id as FK)

3. Bookings Table

Stores customer trip details, including driver assignments, fares, and customer ratings.

| Column Name | Data Type | Description |
|---|---|---|
| booking_id | INT (PK) | Unique identifier for each booking |
| driver_id | INT (FK) | Assigned driver ID |
| service_id | INT (FK) | Type of service booked |
| location_zone | VARCHAR | The zone where booking occurred |
| fare | DECIMAL | Total cost of the trip |
| rating | INT | Customer rating (1-5) |
| booking_time | TIMESTAMP | Timestamp of the booking |
| accepted | BOOLEAN | Indicates if the driver accepted the service |

Relationships

- Primary Key: booking_id
- Foreign Keys: driver_id → Drivers(driver_id), service_id → Services(service_id)

4. Driver_Acceptance Table

Tracks driver performance metrics related to accepting service requests.

| Column Name | Data Type | Description |
|---|---|---|
| driver_id | INT (FK) | Assigned driver ID |
| service_id | INT (FK) | Type of service booked |
| total_requests | INT | Total service requests received |
| total_accepts | INT | Total accepted requests |
| acceptance_rate | DECIMAL | (total_accepts / total_requests) * 100 |

Relationships

- Primary Key: Composite Key (driver_id, service_id)
- Foreign Keys: driver_id → Drivers(driver_id), service_id → Services(service_id)

**Application Implementation**

The Cab Service Optimization System is a Flask-based web application integrated with a MySQL database. The system enables data-driven decision-making by analyzing driver performance, service demand, and revenue while providing real-time insights. The backend is built using Python (Flask framework) and interacts with MySQL to retrieve and store data. The front end consists of HTML templates with dynamic data rendering.

System Architecture

The application follows a three-tier architecture:

1. Frontend (User Interface)
    - Built using HTML, CSS, and Flask templates.
    - Displays drivers, bookings, services, and analytics.
    - Provides search and filtering options.

2. Backend (Flask Web Application)
    - Manages routes and business logic.
    - Connects to MySQL database.
    - Handles user input processing.
3. Database (MySQL)
    - Stores drivers, services, bookings, and driver acceptance rates.
    - Provides structured queries and analytics.
    - Uses views, triggers, and relationships to ensure data consistency.

Functional Components and Database Interaction

Below is an outline of the main functionalities and how they interact with the database.

1. Search for Drivers (by Name or Location Zone)

Objective: Allow users to search for drivers using a keyword (name or location zone).

Implementation Steps:

- A search input field is available on the web interface.
- The Flask route queries the Drivers table using a LIKE clause.
- Matching drivers are displayed dynamically.

*SQL Query Used:*

SELECT * FROM Drivers WHERE name LIKE '%search_term%' OR location_zone LIKE '%search_term%';

2. Filter Bookings (by Date Range)

Objective: Allow users to filter bookings within a specified date range.

Implementation Steps:

- Users select start date and end date.
- A SQL query filters bookings based on booking_time.
- The filtered bookings are displayed in a table.

*SQL Query Used:*

SELECT * FROM Bookings

WHERE DATE(booking_time) BETWEEN 'start_date' AND 'end_date';

3. View Highest-Paid Drivers

Objective: Display drivers who have earned the most revenue from bookings.

Implementation Steps:

- The application retrieves total earnings per driver from the Bookings table.
- The top drivers are displayed on a ranking page.

*SQL Query Used (Using Aggregation):*

```
SELECT d.name, SUM(b.fare) AS total_earnings

FROM Bookings b

JOIN Drivers d ON b.driver_id = d.driver_id

GROUP BY d.driver_id

ORDER BY total_earnings DESC

LIMIT 10;
```

4. Analyze Driver Acceptance Rate

Objective: Evaluate how frequently drivers accept service requests.

Implementation Steps:

- The total requests and accepted requests per driver are stored in Driver_Acceptance.
- The acceptance rate is calculated using total_accepts / total_requests * 100.

*SQL Query Used:*

```
SELECT d.name, s.service_type,

    (da.total_accepts * 100 / da.total_requests) AS acceptance_rate

FROM Driver_Acceptance da

JOIN Drivers d ON da.driver_id = d.driver_id

JOIN Services s ON da.service_id = s.service_id

ORDER BY acceptance_rate DESC;
```

5. Dashboard Displaying Revenue, Bookings & Performance

Objective: Provide a centralized dashboard displaying key metrics.

Implementation Steps:

- SQL Aggregation Queries calculate total drivers, total bookings, and revenue.
- The dashboard displays grouped data by service type and location zone.

*SQL Query Used:*

```
SELECT b.location_zone, s.service_type,
    SUM(b.fare) AS revenue, COUNT(b.booking_id) AS total_bookings,
    AVG(da.acceptance_rate) AS avg_acceptance,
    COUNT(DISTINCT d.driver_id) AS total_drivers
FROM Bookings b
JOIN Services s ON b.service_id = s.service_id
JOIN Driver_Acceptance da ON b.driver_id = da.driver_id
JOIN Drivers d ON b.driver_id = d.driver_id
GROUP BY b.location_zone, s.service_type;
```

**Challenges and Solutions**

1. Challenge: Setting Up Flask with MySQL Integration

    Issue: Initially, setting up Flask to interact with MySQL using MySQL-connector-python was challenging due to connection errors and database authentication issues.

    Solution: Used explicit database connection parameters (host, user, password, database).

2. Challenge: Filtering Bookings by Date

Issue: Users needed to filter bookings between two dates, but querying timestamps caused incorrect results due to time zone differences.

Solution: Converted booking_time into DATE format before filtering. Adjusted Flask route to accept start_date and end_date.

3. Challenge: Grouping Data for Dashboard Analytics

Issue: The system required a dashboard to show total revenue, bookings, and driver performance, grouped by service type and location zone. The problem was SQL query complexity.

Solution:Used aggregation functions (SUM(), COUNT(), AVG()) to generate insights

4. Challenge: Identifying Highest-Paid Drivers Efficiently

Issue: The initial query to retrieve the highest-paid drivers using ORDER BY SUM(fare) DESC was slow with large datasets.

Solution: Created a MySQL View to precompute earnings. Queried the view instead of raw data, improving performance.

**Individual Contributions**

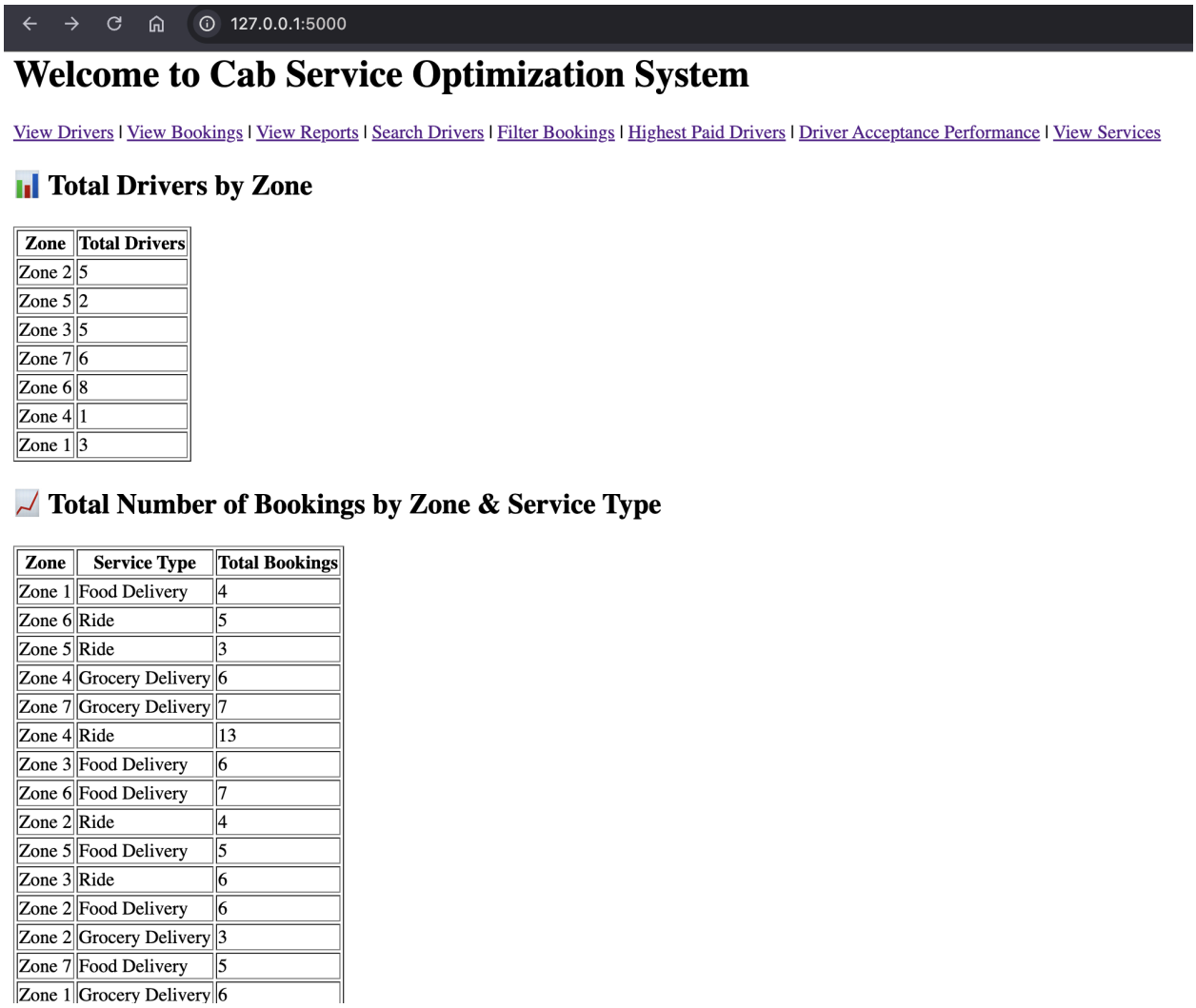
Since this was an individual project, the following contributions were handled by me.

| Task | Contribution |
|---|---|
| Database Design | Designed ERD, created MySQL schema, and defined tables & relationships. |
| Backend Development | Developed Flask API with routes for drivers, bookings, and analytics. |
| Frontend Development | Created HTML templates with dynamic data rendering. |
| SQL Query Optimization | Improved performance using views, indexing, and optimized queries. |

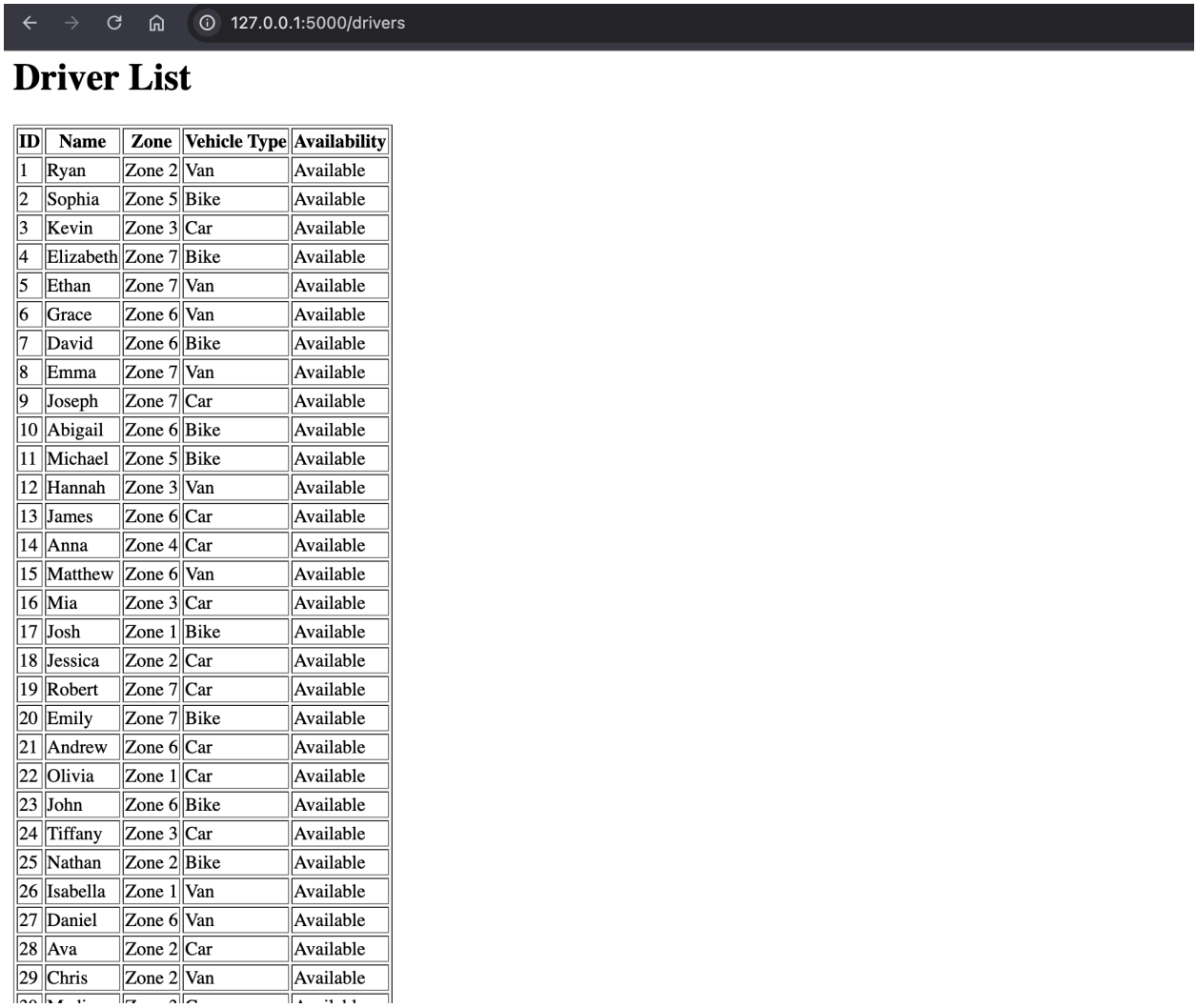
| | |
|---|---|
| Testing & Debugging | Conducted end-to-end testing, resolved bugs, and ensured data consistency. |
| Documentation | Wrote README.md with installation instructions and setup guide. |

### UI Screenshots

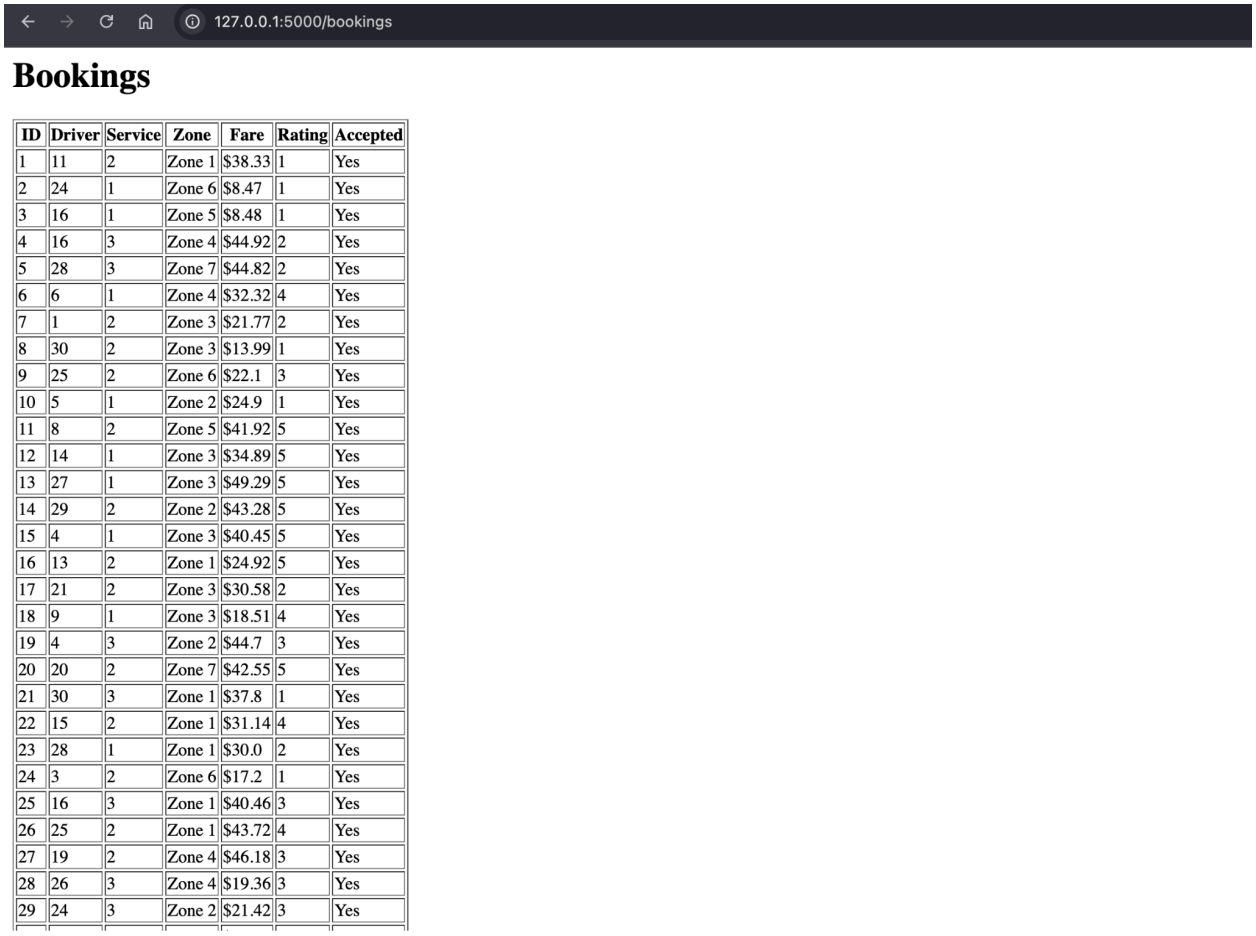
1. HomeScreen



2. Click on Drivers to check the details of the driver's table



3. Click on View Bookings for details




| ID | Driver | Service | Zone | Fare | Rating | Accepted |
|---|---|---|---|---|---|---|
| 1 | 11 | 2 | Zone 1 | $38.33 | 1 | Yes |
| 2 | 24 | 1 | Zone 6 | $8.47 | 1 | Yes |
| 3 | 16 | 1 | Zone 5 | $8.48 | 1 | Yes |
| 4 | 16 | 3 | Zone 4 | $44.92 | 2 | Yes |
| 5 | 28 | 3 | Zone 7 | $44.82 | 2 | Yes |
| 6 | 6 | 1 | Zone 4 | $32.32 | 4 | Yes |
| 7 | 1 | 2 | Zone 3 | $21.77 | 2 | Yes |
| 8 | 30 | 2 | Zone 3 | $13.99 | 1 | Yes |
| 9 | 25 | 2 | Zone 6 | $22.1 | 3 | Yes |
| 10 | 5 | 1 | Zone 2 | $24.9 | 1 | Yes |
| 11 | 8 | 2 | Zone 5 | $41.92 | 5 | Yes |
| 12 | 14 | 1 | Zone 3 | $34.89 | 5 | Yes |
| 13 | 27 | 1 | Zone 3 | $49.29 | 5 | Yes |
| 14 | 29 | 2 | Zone 2 | $43.28 | 5 | Yes |
| 15 | 4 | 1 | Zone 3 | $40.45 | 5 | Yes |
| 16 | 13 | 2 | Zone 1 | $24.92 | 5 | Yes |
| 17 | 21 | 2 | Zone 3 | $30.58 | 2 | Yes |
| 18 | 9 | 1 | Zone 3 | $18.51 | 4 | Yes |
| 19 | 4 | 3 | Zone 2 | $44.7 | 3 | Yes |
| 20 | 20 | 2 | Zone 7 | $42.55 | 5 | Yes |
| 21 | 30 | 3 | Zone 1 | $37.8 | 1 | Yes |
| 22 | 15 | 2 | Zone 1 | $31.14 | 4 | Yes |
| 23 | 28 | 1 | Zone 1 | $30.0 | 2 | Yes |
| 24 | 3 | 2 | Zone 6 | $17.2 | 1 | Yes |
| 25 | 16 | 3 | Zone 1 | $40.46 | 3 | Yes |
| 26 | 25 | 2 | Zone 1 | $43.72 | 4 | Yes |
| 27 | 19 | 2 | Zone 4 | $46.18 | 3 | Yes |
| 28 | 26 | 3 | Zone 4 | $19.36 | 3 | Yes |
| 29 | 24 | 3 | Zone 2 | $21.42 | 3 | Yes |

4. Click on View Report for Driver Performance and Service Demand


## Driver Performance

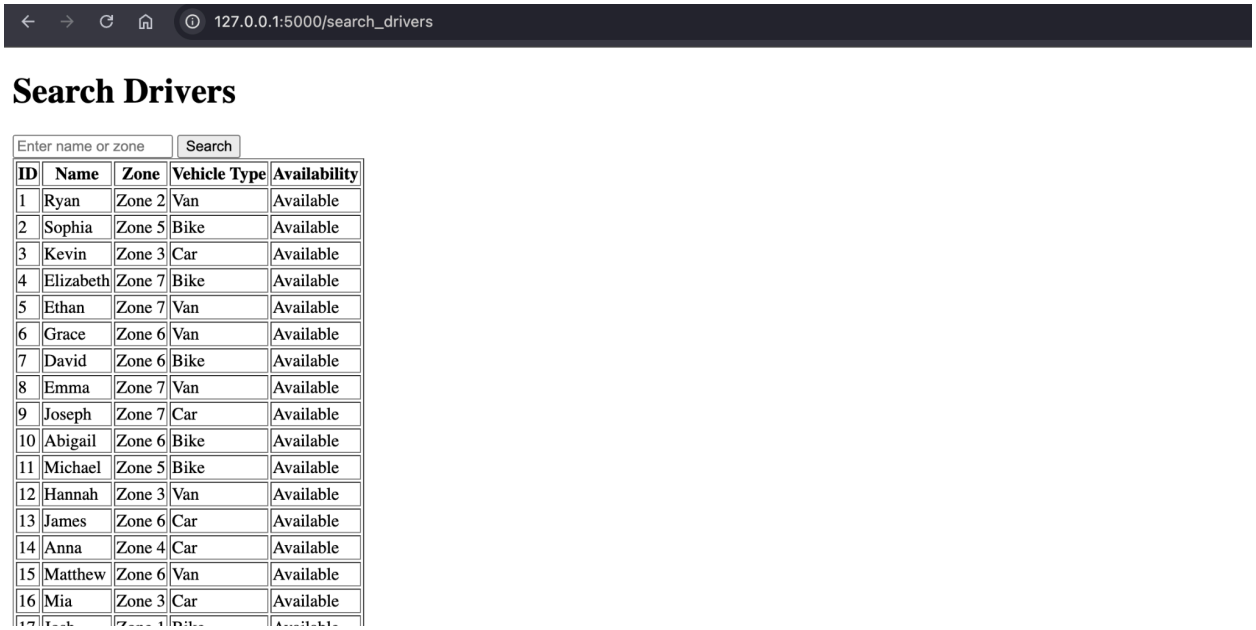
| ID | Name | Zone | Vehicle | Total Bookings | Avg Rating | Total Earnings |
|---|---|---|---|---|---|---|
| 1 | Ryan | Zone 2 | Van | 4 | 2.2500 | $140.0 |
| 2 | Sophia | Zone 5 | Bike | 1 | 1.0000 | $44.47 |
| 3 | Kevin | Zone 3 | Car | 3 | 2.6667 | $66.19 |
| 4 | Elizabeth | Zone 7 | Bike | 5 | 4.4000 | $149.81 |
| 5 | Ethan | Zone 7 | Van | 3 | 2.0000 | $71.84 |
| 6 | Grace | Zone 6 | Van | 3 | 4.6667 | $97.41 |
| 7 | David | Zone 6 | Bike | 2 | 4.0000 | $63.27 |
| 8 | Emma | Zone 7 | Van | 5 | 3.4000 | $157.33 |
| 9 | Joseph | Zone 7 | Car | 3 | 3.3333 | $69.07000000000001 |
| 10 | Abigail | Zone 6 | Bike | 5 | 3.2000 | $151.3 |
| 11 | Michael | Zone 5 | Bike | 5 | 1.8000 | $111.42999999999999 |
| 12 | Hannah | Zone 3 | Van | 0 | None | $None |
| 13 | James | Zone 6 | Car | 5 | 4.0000 | $138.32999999999998 |
| 14 | Anna | Zone 4 | Car | 1 | 5.0000 | $34.89 |
| 15 | Matthew | Zone 6 | Van | 3 | 3.0000 | $88.16 |
| 16 | Mia | Zone 3 | Car | 3 | 2.0000 | $93.86 |
| 17 | Josh | Zone 1 | Bike | 6 | 2.5000 | $162.24 |


| | | | | | | |
|---|---|---|---|---|---|---|
| 24 | Tiffany | Zone 3 | Car | 4 | 2.7500 | $53.16 |
| 25 | Nathan | Zone 2 | Bike | 3 | 3.0000 | $111.66999999999999 |
| 26 | Isabella | Zone 1 | Van | 4 | 3.5000 | $85.87 |
| 27 | Daniel | Zone 6 | Van | 3 | 4.3333 | $131.54999999999998 |
| 28 | Ava | Zone 2 | Car | 4 | 2.0000 | $167.49 |
| 29 | Chris | Zone 2 | Van | 4 | 3.7500 | $85.98 |
| 30 | Madison | Zone 3 | Car | 7 | 2.1429 | $170.53000000000003 |

## Service Demand

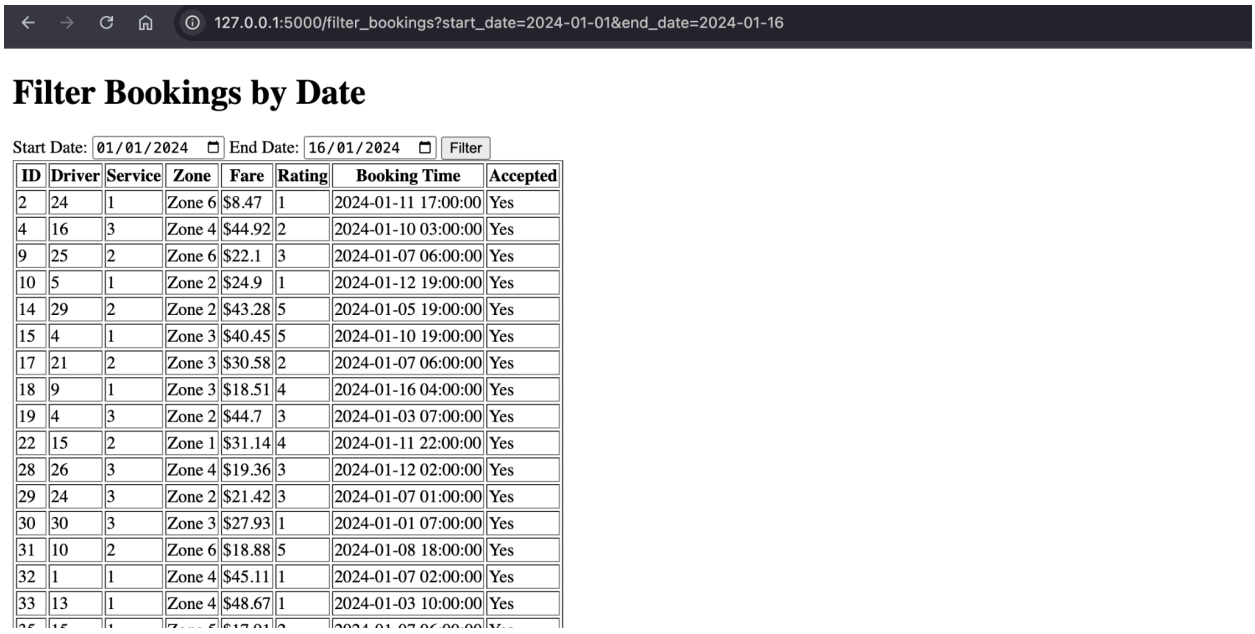
| Zone | Service | Total Bookings |
|---|---|---|
| Zone 1 | Food Delivery | 4 |
| Zone 6 | Ride | 5 |
| Zone 5 | Ride | 3 |
| Zone 4 | Grocery Delivery | 6 |
| Zone 7 | Grocery Delivery | 7 |
| Zone 4 | Ride | 13 |
| Zone 3 | Food Delivery | 6 |
| Zone 6 | Food Delivery | 7 |
| Zone 2 | Ride | 4 |
| Zone 5 | Food Delivery | 5 |
| Zone 3 | Ride | 6 |

5. Search for Driver’s details by name or zone




| ID | Name | Zone | Vehicle Type | Availability |
|---|---|---|---|---|
| 1 | Ryan | Zone 2 | Van | Available |
| 2 | Sophia | Zone 5 | Bike | Available |
| 3 | Kevin | Zone 3 | Car | Available |
| 4 | Elizabeth | Zone 7 | Bike | Available |
| 5 | Ethan | Zone 7 | Van | Available |
| 6 | Grace | Zone 6 | Van | Available |
| 7 | David | Zone 6 | Bike | Available |
| 8 | Emma | Zone 7 | Van | Available |
| 9 | Joseph | Zone 7 | Car | Available |
| 10 | Abigail | Zone 6 | Bike | Available |
| 11 | Michael | Zone 5 | Bike | Available |
| 12 | Hannah | Zone 3 | Van | Available |
| 13 | James | Zone 6 | Car | Available |
| 14 | Anna | Zone 4 | Car | Available |
| 15 | Matthew | Zone 6 | Van | Available |
| 16 | Mia | Zone 3 | Car | Available |

6. Select a start and end date to filter bookings by date

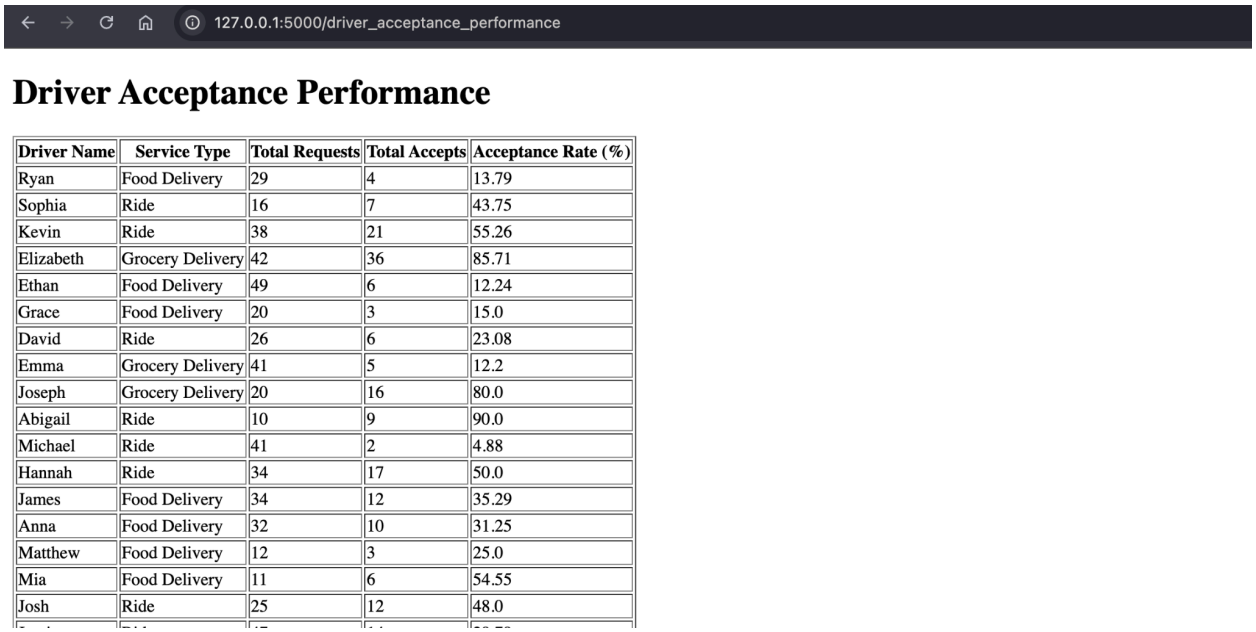



| ID | Driver | Service | Zone | Fare | Rating | Booking Time | Accepted |
|---|---|---|---|---|---|---|---|
| 2 | 24 | 1 | Zone 6 | $8.47 | 1 | 2024-01-11 17:00:00 | Yes |
| 4 | 16 | 3 | Zone 4 | $44.92 | 2 | 2024-01-10 03:00:00 | Yes |
| 9 | 25 | 2 | Zone 6 | $22.1 | 3 | 2024-01-07 06:00:00 | Yes |
| 10 | 5 | 1 | Zone 2 | $24.9 | 1 | 2024-01-12 19:00:00 | Yes |
| 14 | 29 | 2 | Zone 2 | $43.28 | 5 | 2024-01-05 19:00:00 | Yes |
| 15 | 4 | 1 | Zone 3 | $40.45 | 5 | 2024-01-10 19:00:00 | Yes |
| 17 | 21 | 2 | Zone 3 | $30.58 | 2 | 2024-01-07 06:00:00 | Yes |
| 18 | 9 | 1 | Zone 3 | $18.51 | 4 | 2024-01-16 04:00:00 | Yes |
| 19 | 4 | 3 | Zone 2 | $44.7 | 3 | 2024-01-03 07:00:00 | Yes |
| 22 | 15 | 2 | Zone 1 | $31.14 | 4 | 2024-01-11 22:00:00 | Yes |
| 28 | 26 | 3 | Zone 4 | $19.36 | 3 | 2024-01-12 02:00:00 | Yes |
| 29 | 24 | 3 | Zone 2 | $21.42 | 3 | 2024-01-07 01:00:00 | Yes |
| 30 | 30 | 3 | Zone 3 | $27.93 | 1 | 2024-01-01 07:00:00 | Yes |
| 31 | 10 | 2 | Zone 6 | $18.88 | 5 | 2024-01-08 18:00:00 | Yes |
| 32 | 1 | 1 | Zone 4 | $45.11 | 1 | 2024-01-07 02:00:00 | Yes |
| 33 | 13 | 1 | Zone 4 | $48.67 | 1 | 2024-01-03 10:00:00 | Yes |

7. Highest paid driver with the ID, their name and total eranings


| ID | Name | Total Earnings |
|---|---|---|
| 30 | Madison | $170.53 |
| 28 | Ava | $167.48999999999998 |
| 17 | Josh | $162.24 |
| 8 | Emma | $157.33 |
| 20 | Emily | $153.14 |
| 10 | Abigail | $151.3 |
| 4 | Elizabeth | $149.81 |
| 21 | Andrew | $143.67 |
| 1 | Ryan | $140.0 |
| 13 | James | $138.33 |
| 27 | Daniel | $131.55 |
| 25 | Nathan | $111.66999999999999 |
| 11 | Michael | $111.42999999999999 |
| 6 | Grace | $97.41 |
| 16 | Mia | $93.86000000000001 |
| 15 | Matthew | $88.16 |
| 29 | Chris | $85.98 |

8. Driver Acceptance performance with acceptance rate




| Driver Name | Service Type | Total Requests | Total Accepts | Acceptance Rate (%) |
|---|---|---|---|---|
| Ryan | Food Delivery | 29 | 4 | 13.79 |
| Sophia | Ride | 16 | 7 | 43.75 |
| Kevin | Ride | 38 | 21 | 55.26 |
| Elizabeth | Grocery Delivery | 42 | 36 | 85.71 |
| Ethan | Food Delivery | 49 | 6 | 12.24 |
| Grace | Food Delivery | 20 | 3 | 15.0 |
| David | Ride | 26 | 6 | 23.08 |
| Emma | Grocery Delivery | 41 | 5 | 12.2 |
| Joseph | Grocery Delivery | 20 | 16 | 80.0 |
| Abigail | Ride | 10 | 9 | 90.0 |
| Michael | Ride | 41 | 2 | 4.88 |
| Hannah | Ride | 34 | 17 | 50.0 |
| James | Food Delivery | 34 | 12 | 35.29 |
| Anna | Food Delivery | 32 | 10 | 31.25 |
| Matthew | Food Delivery | 12 | 3 | 25.0 |
| Mia | Food Delivery | 11 | 6 | 54.55 |
| Josh | Ride | 25 | 12 | 48.0 |

9. List of available services


# Available Services

| Service ID | Service Type | Base Fare | Per Mile Charge |
|---|---|---|---|
| 1 | Ride | $5.0 | $1.5 |
| 2 | Food Delivery | $3.0 | $1.0 |
| 3 | Grocery Delivery | $4.0 | $1.2 |

Back to Home