Initial Backlog

1. USER STORIES

1.1. Booking & Reservation

1. **As a customer**, I want to enter my pickup and drop-off locations, so that I can book a ride easily.
2. **As a customer**, I want to see an estimated fare before booking, so that I know the cost in advance.
3. **As a customer**, I want to select the type of vehicle (economy, standard, premium), so that I can choose the best option for my needs.
4. **As a customer**, I want to schedule a ride in advance, so that I can plan my trips conveniently.
5. **As a customer**, I want to receive a booking confirmation, so that I know my ride is reserved.

1.2. User Accounts & Authentication

6. **As a customer**, I want to create an account, so that I can manage my bookings.
7. **As a customer**, I want to log in securely, so that my personal data is protected.
8. **As an admin**, I want to manage user accounts, so that I can handle customer issues.

1.3. Ride Management & Tracking

9. **As an admin**, I want to create and manage driver profiles, so that I can assign rides to available drivers.
10. **As an admin**, I want to manually assign a driver to a ride, so that I can control the allocation of trips.
11. **As an admin**, I want to update a ride’s status (pending, assigned, completed), so that both the driver and the customer are informed.
12. **As a customer**, I want to receive a notification when my ride is assigned, so that I know who my driver is.
13. **As a customer**, I want to view my ride history, so that I can keep track of my past bookings.

1.4. Dynamic Pricing & Demand Prediction

14. **As an admin**, I want to see real-time demand data, so that I can adjust pricing dynamically.
15. **As an admin**, I want the system to automatically adjust fares based on demand, so that revenue is optimized.
16. **As a customer**, I want to see price changes based on peak hours or high demand, so that I understand why prices vary.

1.5. Admin & Management Features

17. **As an admin**, I want to view all active and completed bookings, so that I can monitor platform activity.
18. **As an admin**, I want to generate reports on revenue, demand, and ride efficiency, so that I can analyze business performance.
19. **As an admin**, I want to adjust pricing rules manually, so that I have control over fare adjustments.

2. TECHNICAL REQUIREMENTS

2.1. System Architecture

- The system must be built using a **RESTful API** to allow seamless communication between the frontend and backend.
- The backend must be developed using **Python (FastAPI)** for handling business logic.
- The frontend must be developed using **React.js or Vue.js** for a dynamic user experience.
- The system must support **role-based access control (RBAC)** for users, drivers, and admins.
- The platform must integrate with **Google Maps API** (or an alternative) for route calculations.
- The system must include **a machine-learning model** for demand prediction and dynamic pricing.

2.2. Backend Requirements

- Must use **PostgreSQL or MySQL** as the database.
- Must implement **authentication & authorization** using OAuth or JWT tokens.
- Must provide API endpoints for:
  - **User management** (signup, login, role assignment).
  - **Ride booking** (create, update, cancel reservations).
  - **Fare estimation** (pricing logic, demand-based changes).
  - **Admin functionalities** (analytics, fare adjustments).
- Must include **unit tests and integration tests** for core functionalities.

2.3. Frontend Requirements

- The UI must be **responsive** and optimized for desktop & mobile.
- Must allow users to:
  - **Book rides** (select pickup/drop-off, vehicle type, estimated fare).
  - **View trip history** and manage bookings.
  - **Receive notifications** (email/SMS) for booking confirmations.

- Admin dashboard must allow:
  - **Viewing and managing bookings** in real time.
  - **Monitoring pricing changes** and adjusting fare rules.
  - **Viewing analytics** on demand and revenue.

2.4. Machine Learning & Dynamic Pricing Requirements

- Must collect and preprocess **synthetic demand data** for model training.
- Must train and validate a **predictive model** using **scikit-learn**.
- Must integrate **dynamic pricing logic** into the backend API.
- Must allow **manual override** of automatic fare adjustments by admins.
- The model must factor in:
  - **Time of day, location, weather, events, and historical demand patterns.**

2.5. Performance & Security Requirements

- The system must handle **at least 1,000 concurrent users** without performance degradation.
- Must implement **data encryption** for user information and payment details.
- Must use **rate limiting** to prevent API abuse.
- The database must have **regular backups** and disaster recovery mechanisms.

3. CODE SPIKES (Research & Exploration Tasks)

3.1. Dynamic Pricing & Demand Prediction

- **Find datasets** (real or synthetic) for training a demand prediction model.
- **Test different ML models** (Random Forest, XGBoost, Neural Networks) for predicting ride demand.
- **Research dynamic pricing strategies** (Uber-like surge pricing vs. rule-based fare adjustments).
- **Evaluate ML model integration methods** (direct API calls vs. batch processing).

3.2. Google Maps API & Route Optimization

- **Test Google Maps API integration** for route and fare estimation.
- **Explore alternatives** (Azure) if Google Maps does not work well.
- **Optimize API calls** to reduce costs and improve performance.

3.3. Authentication & Security

- **Research OAuth vs. JWT vs. Session-based authentication** for secure user logins.

- **Implement role-based access control (RBAC)** to restrict user/admin permissions.
- **Research encryption techniques** for secure data storage (e.g., passwords, payment info).

3.4. Backend Infrastructure & Deployment

- **Decide on cloud hosting** (AWS, Azure).
- **Research best practices for API scalability** (caching, load balancing).

4. Sprint 1 Prioritization (2 Weeks)

Goal: Set up the basic **backend & frontend structure** while researching the **ML model for demand prediction**.

4.1. Web App (60% Focus)

1. Backend Setup (API & Database)

- Set up a **FastAPI** project.
- Implement a **basic API** for **user registration & authentication**.
- Set up a **PostgreSQL/MySQL database** with tables for users & bookings.
- Create database schema

2. Frontend Setup

- Initialize a **React/Vue project**.
- Study frontend language
- Implement a basic page

4.2. ML Model (40% Focus)

4. Research & Collect Data for Demand Prediction

- Identify **potential datasets** for training.
- Test **data generation techniques** for synthetic demand data.

5. Explore ML Models for Demand Prediction

- Compare **Random Forest vs. XGBoost vs. Neural Networks**.
- Implement a **basic model prototype** to analyze demand trends.

4.3. Sprint Goal

✅ **Have a working backend & frontend where users can register & book rides.**
✅ **Have an initial prototype of the demand prediction model with test data.**