# RIDE HAILING APP

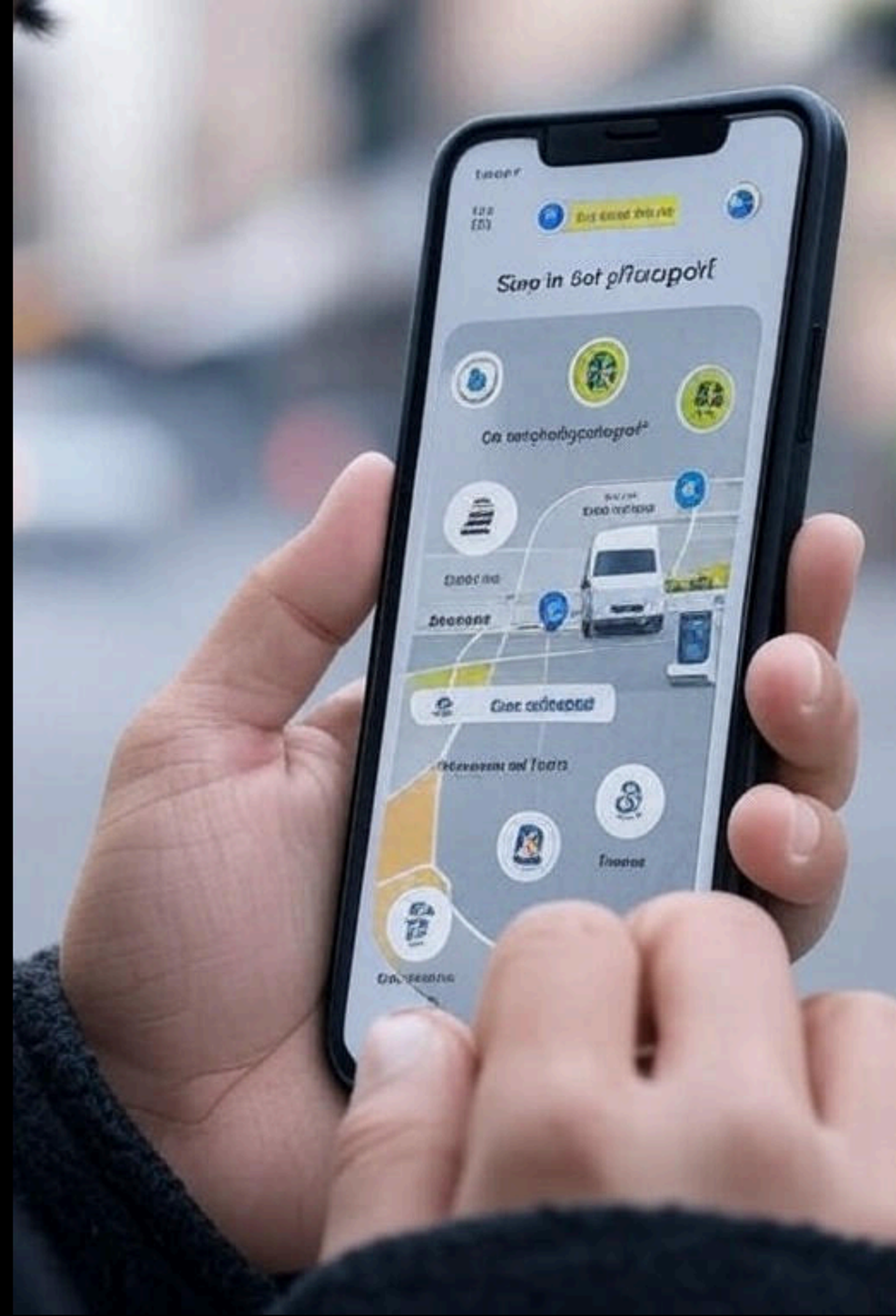
**STUDENT NUMBER:**

**PREPARED BY: ALFIYA ANJUM**

**BSC (HONS) SOFTWARE ENGINEERING**

# 🚗 WHAT IS THIS APP ABOUT?

- A full-stack mobile ride-booking application
- Built with React Native, Stripe, Google Maps, Clerk, and Zustand
- Users can:
  - Book rides
  - View fare estimates
  - Track rides
  - Pay securely online
  - View ride history



## WHY DID I BUILD THIS APP?

- In Bulgaria, many drivers don’t accept card payments
- Riders often need to carry cash or guess the fare in advance
- I wanted to solve this problem with a cashless, app-based ride solution
- Inspired by real-life commuting challenges

## WHAT PROBLEM DOES IT SOLVE?

- No more cash hassle – pay through the app using Stripe
- Fare transparency – know the price before booking
- Convenience – find rides, book, and pay in one place
- Track and manage rides easily with history and map integration

# 🎯 GOALS OF THE PROJECT

Learned to build a full-stack mobile app from scratch

Used real-world tools like:

- Stripe for payments
- Google Maps API for navigation and location
- Clerk for secure authentication

Build an app with:

- Modern UI/UX using Tailwind CSS
- Real-time features like live maps and autocomplete
- Showcased my skills as a mobile and full-stack developer

# ⚙ TECH STACK

| Layer | Tools / Technologies Used |
| --- | --- |
| Frontend | React Native, TailwindCSS (via NativeWind), Expo |
| Authentication | Clerk (Email/Password & Google OAuth) |
| Backend / Database | Serverless PostgreSQL (via Neon) |
| Maps & Location | Google Maps API, Google Places Autocomplete |
| Payments and AI chabot | Stripe and Grok |
| State Management | Zustand |
| Cross-Platform Support | Optimized for Android |

# 🔋 KEY FEATURES

## 🧭 ONBOARDING & AUTH

- Seamless user onboarding flow
- Email/Password authentication with verification
- Google OAuth login
- Secure user session management with Clerk
- Role-based authorization

## 🚗 RIDE BOOKING

- Search for rides by entering "From" and "To"
- See estimated fare & time before booking
- Confirm ride details before payment
- Pay securely using Stripe
- Ride is created only after successful payment

## 🗺️ MAPS & LOCATION

- Live location tracking on Google Maps
- Map markers for available rides
- Google Places Autocomplete for easy location search
- Select rides directly from the map

## 📱 USER EXPERIENCE

- Recent rides list for quick access
- Ride history with past bookings
- Profile management to update user info
- Responsive design for Android and iOS devices

## AUTHENTICATION (CLERK + GOOGLE OAUTH)

```tsx
const OAuth = () => {
  const { startOAuthFlow } = useOAuth({ strategy: "oauth_google" });

  const handleGoogleSignIn = async () => {
    const result = await googleOAuth(startOAuthFlow);

    if (result.code === "session_exists") {
      Alert.alert("Success", "Session exists. Redirecting to home screen.");
      router.replace("/(root)/(tabs)/home");
    }

    Alert.alert(result.success ? "Success" : "Error", result.message);
  };

  return (
    <View>
      <View className="flex flex-row justify-center items-center mt-4 gap-x-3">
        <View className="flex-1 h-[1px] bg-general-100" />
        <Text className="text-lg">Or</Text>
        <View className="flex-1 h-[1px] bg-general-100" />
      </View>

      <CustomButton
        title="Log In with Google"
        className="mt-5 w-full shadow-none"
        IconLeft={() => (
          <Image
            source={icons.google}
            resizeMode="contain"
            className="w-5 h-5 mx-2"
          />
        )}
        bgVariant="outline"
        textVariant="primary"
        onPress={handleGoogleSignIn}
      />
    </View>
  );
};

export default OAuth;
```

## AI INTEGRATION (GROK)

```tsx
const GROQ_API_URL = "https://api.groq.com/openai/v1/chat/completions";
const API_KEY = process.env.EXPO_PUBLIC_GROQ_API_KEY;

export const sendMessageToGroq = async (message: string) => {
  try {
    const response = await axios.post(
      GROQ_API_URL,
      {
        messages: [{ role: "user", content: message }],
        model: "gemma2-9b-it",        You, 1 second ago • Uncommitted changes
        temperature: 0.7,
        max_tokens: 1000,
      },
      {
        headers: {
          Authorization: `Bearer ${API_KEY}`,
          "Content-Type": "application/json",
        },
      }
    );

    return response.data.choices[0].message.content;
  } catch (error: any) {
    console.error("Error sending message to Groq:", error.response?.data || error
    throw error;
  }
};
```

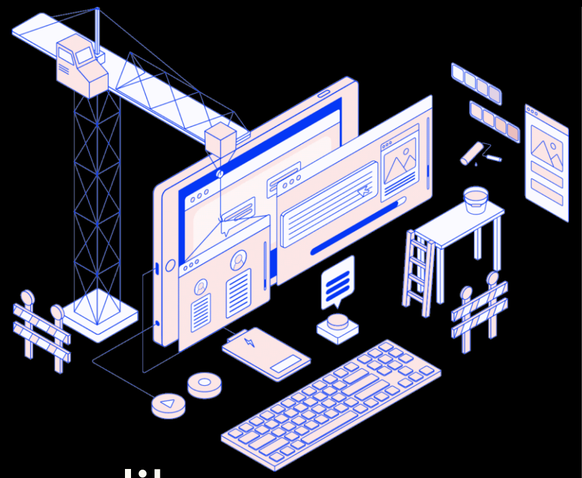
## 💳 STRIPE PAYMENT INTEGRATION

```
const Payment = ({
  fullName,
  email,
  amount,
  driverId,
  rideTime,
}: PaymentProps) => {
  const { initPaymentSheet, presentPaymentSheet } = useStripe();
  const {
    userAddress,
    userLongitude,
    userLatitude,
    destinationLatitude,
    destinationAddress,
    destinationLongitude,
  } = useLocationStore();

  const { userId } = useAuth();
  const [success, setSuccess] = useState<boolean>(false);

  const openPaymentSheet = async () => {
    const initialized = await initializePaymentSheet();
    if (!initialized) return;

    const { error } = await presentPaymentSheet();

    if (error) {
      Alert.alert(`Error code: ${error.code}`, error.message);
    } else {
      await fetchAPI("/(api)/ride/create", {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify({
          origin_address: userAddress,
          destination_address: destinationAddress,
          origin_latitude: userLatitude,
          origin_longitude: userLongitude,
          destination_latitude: destinationLatitude,
          destination_longitude: destinationLongitude,
          ride_time: rideTime.toFixed(0),
          fare_price: parseInt(amount) * 100,
          payment_status: "paid",
```

## MAPVIEWDIRECTIONS

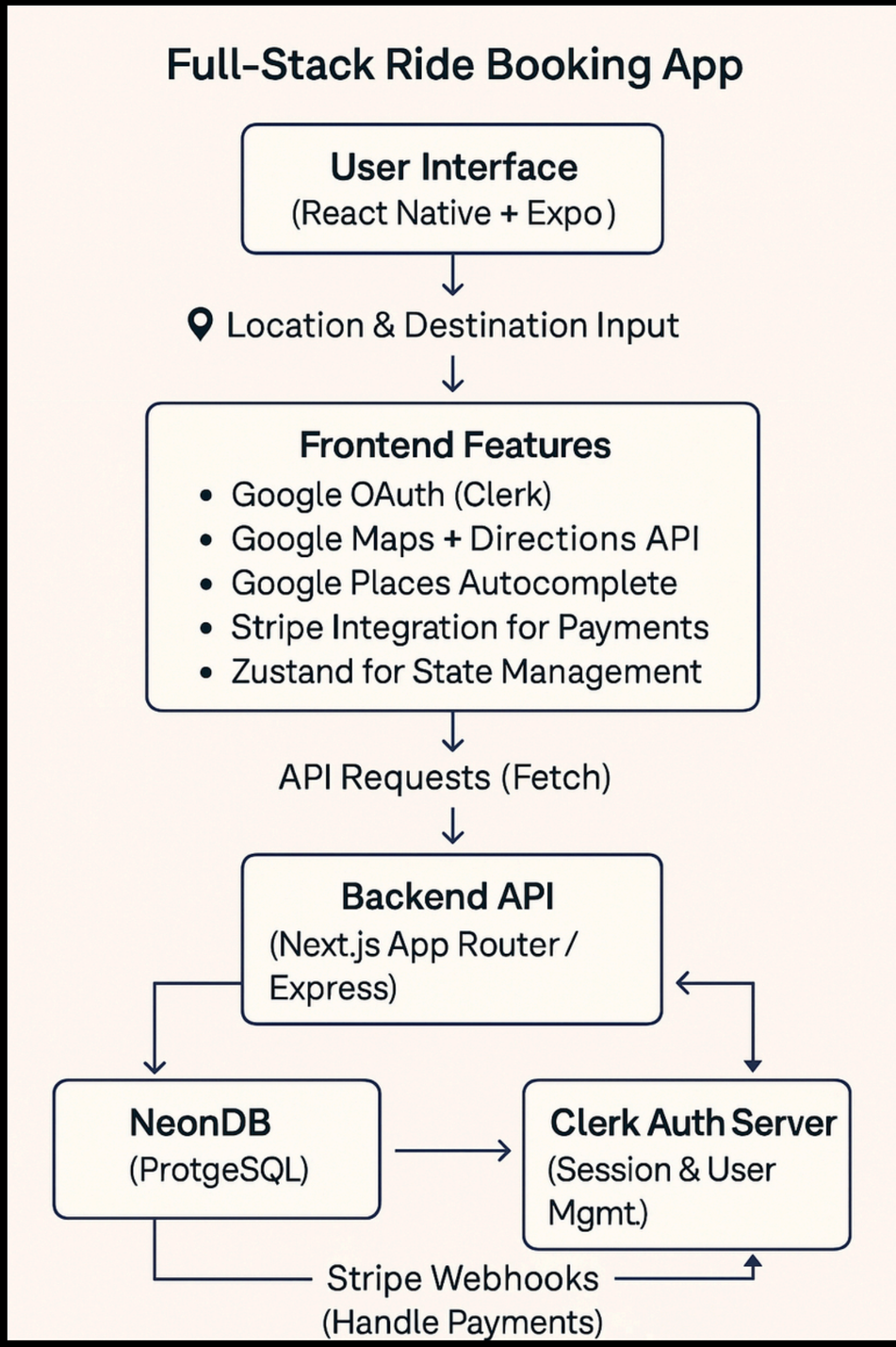
```
{destinationLatitude && destinationLongitude && (
          <>
            <Marker
              key="destination"
              coordinate={{
                latitude: destinationLatitude,
                longitude: destinationLongitude,
              }}
              title="Destination"
              image={icons.pin}
            />
            <MapViewDirections
              origin={{
                latitude: userLatitude!,
                longitude: userLongitude!,
              }}
              destination={{
                latitude: destinationLatitude,
                longitude: destinationLongitude,
              }}
              apikey={process.env.EXPO_PUBLIC_GOOGLE_API_KEY!}
              strokeColor="#0286FF"
              strokeWidth={2}
            />
          </>
        )}
</MapView>

        </View>
      );

  };

  export default Map;
```

# 🔧 App Architecture



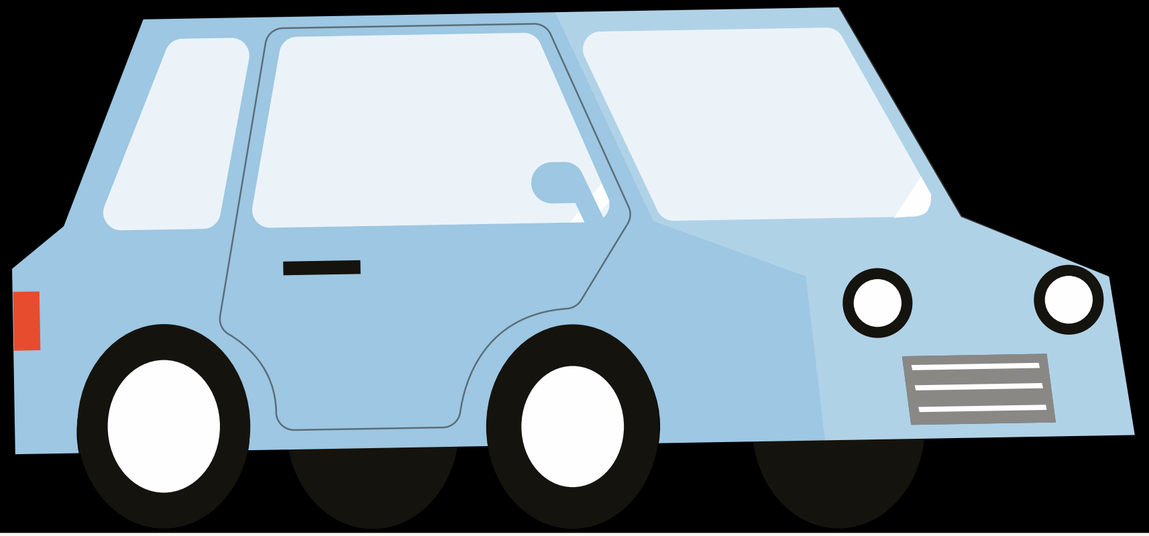
## 🔍 WHAT THIS SHOWS:

- Frontend is React Native with features like Google Maps, Autocomplete, Clerk login, etc.
- You use Zustand for simple global state like selected driver and ride details.
- Backend API handles saving rides, user creation, fetching drivers.
- NeonDB (PostgreSQL) stores user and ride data.
- Clerk manages login, sessions, and user info.
- Stripe handles payment processing and confirmations.



# 🧪 TESTING OVERVIEW

- Framework: Jest for both unit & integration testing
- Focus: Verified reliability of payment, sign-up, API, and UI components



## Key Tests:

Payment Flow

- Simulated end-to-end Stripe payment:
  → intent creation → sheet init → sheet present → ride creation
- (All mocked using Jest)

Sign-Up Flow

- Simulated user input & success confirmation
- API Service
- Mocked fetchAPI to test successful ride creation response
- Custom Components
- Verified rendering of buttons and static texts

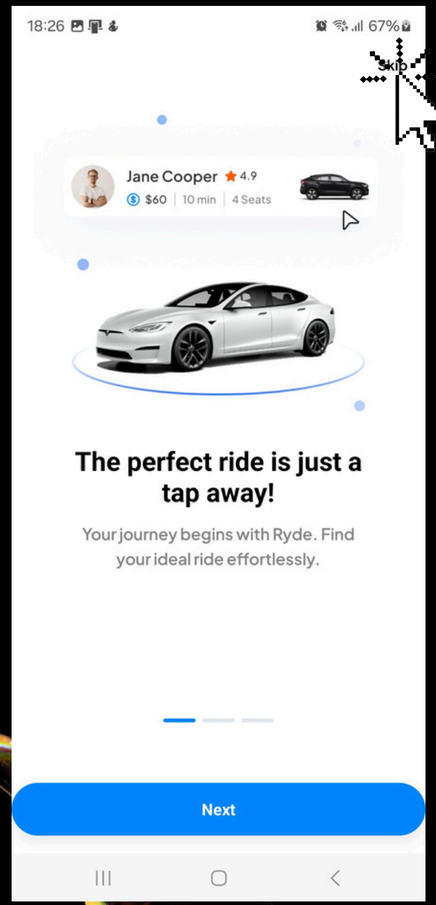
## Mocked Dependencies:

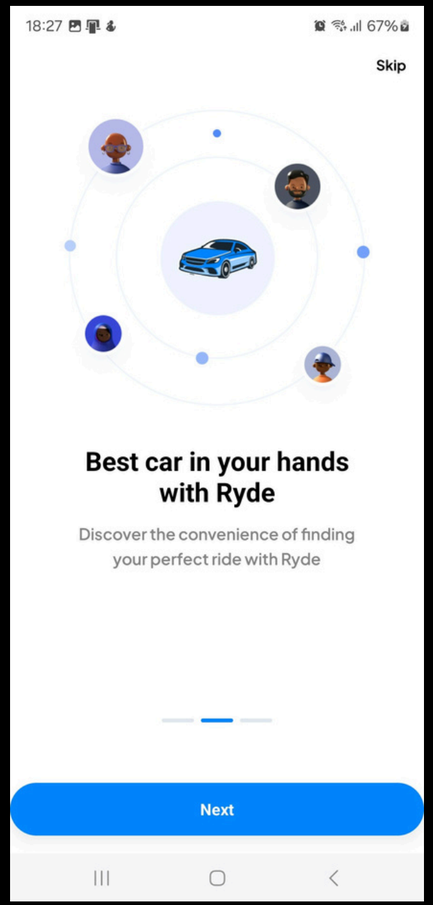
Stripe SDK, Clerk Auth, Modals, and Location Store

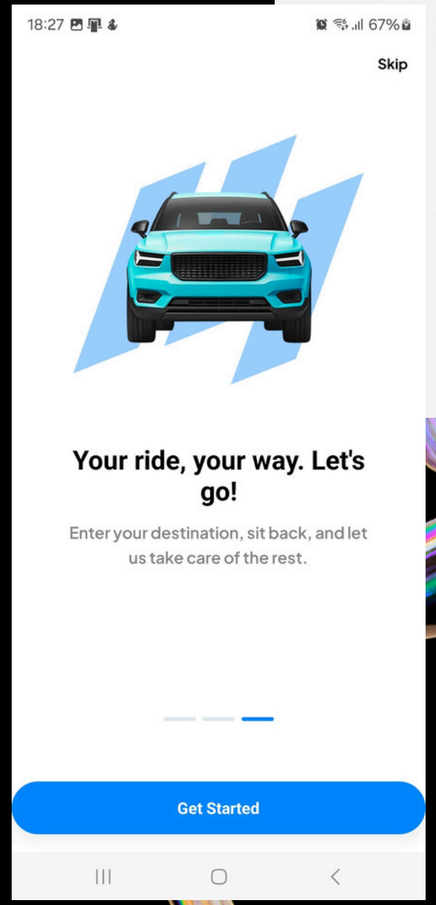
## 🎯 Purpose:

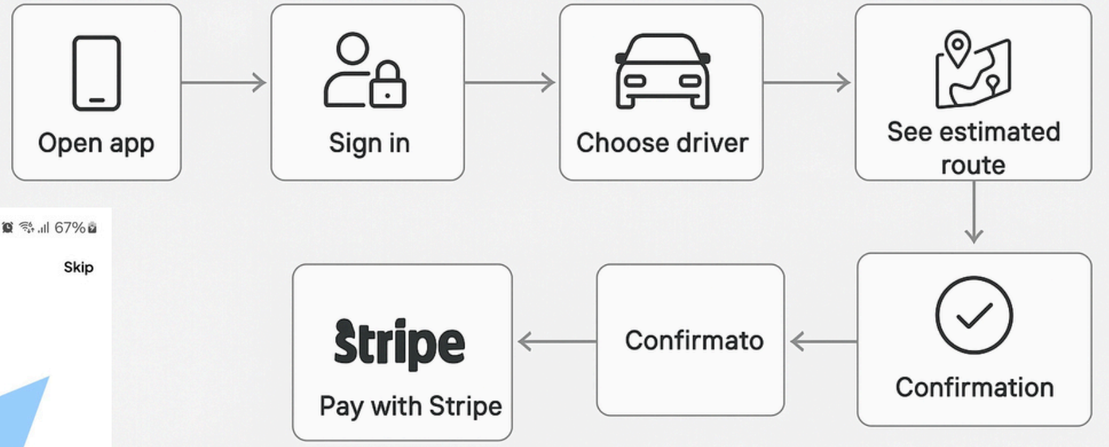
- Confirms critical features work as expected
- Boosts reliability & maintainability
- Prevents regressions during future updates

# 1. APP FLOW DIAGRAM / VISVALISATION

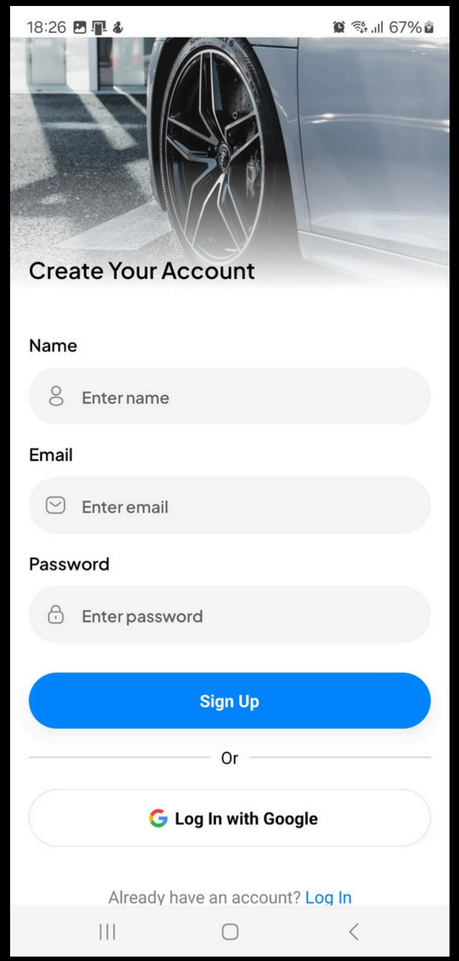


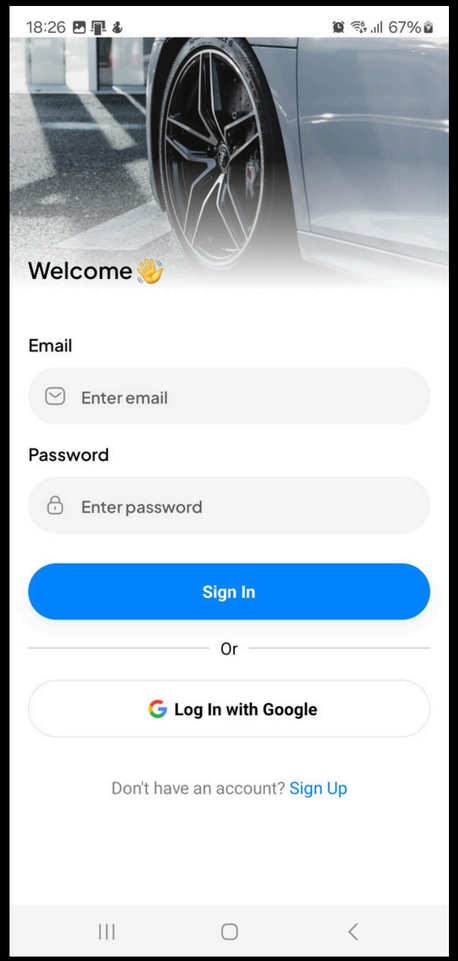



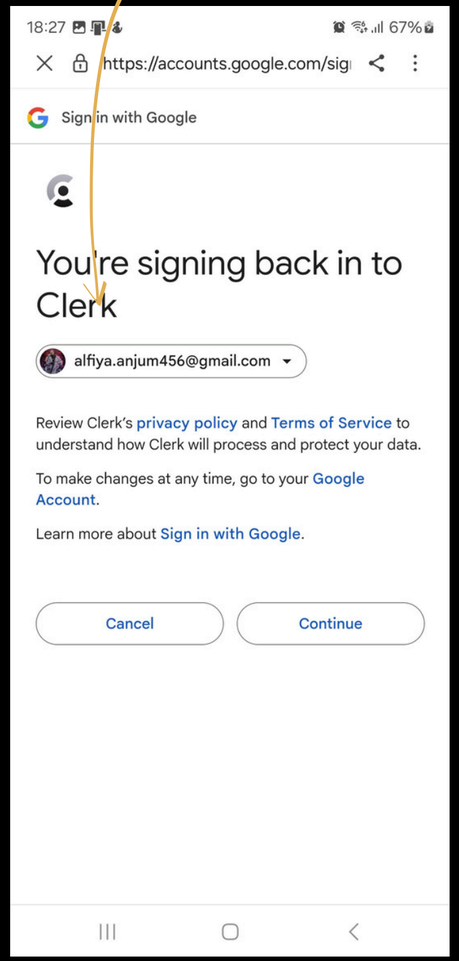



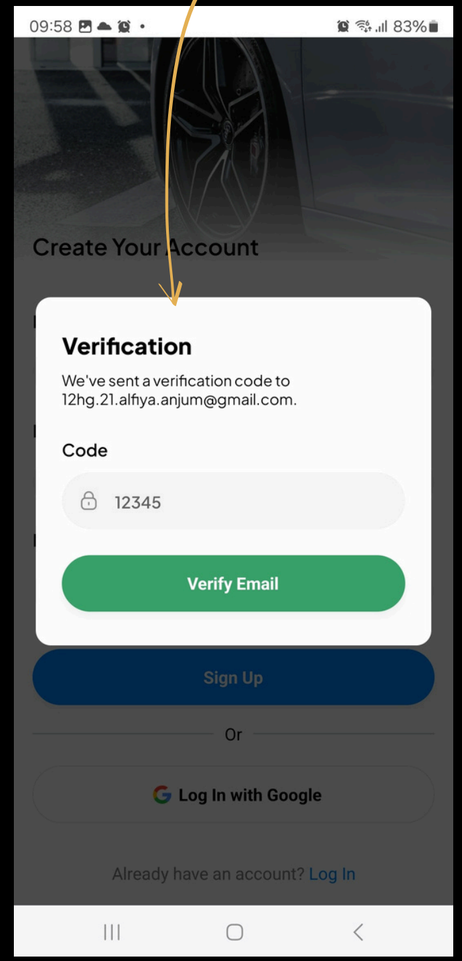



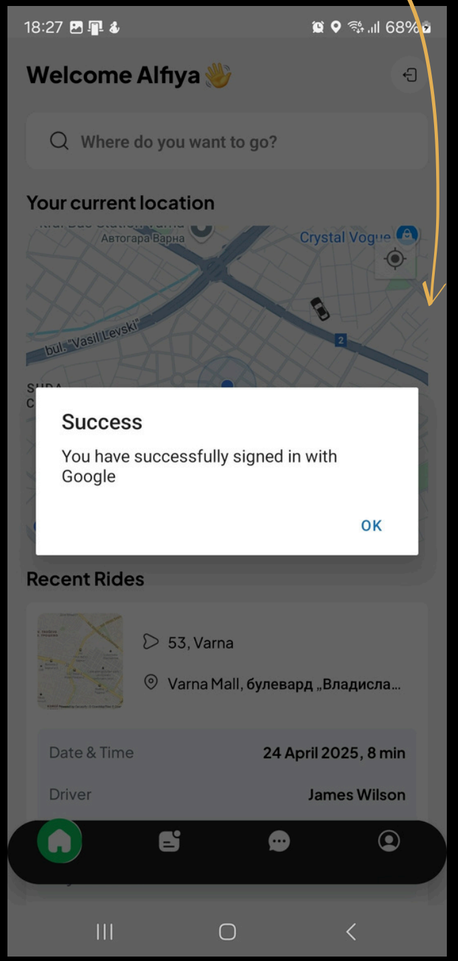

# SIGNUP/ LOGIN PAGES



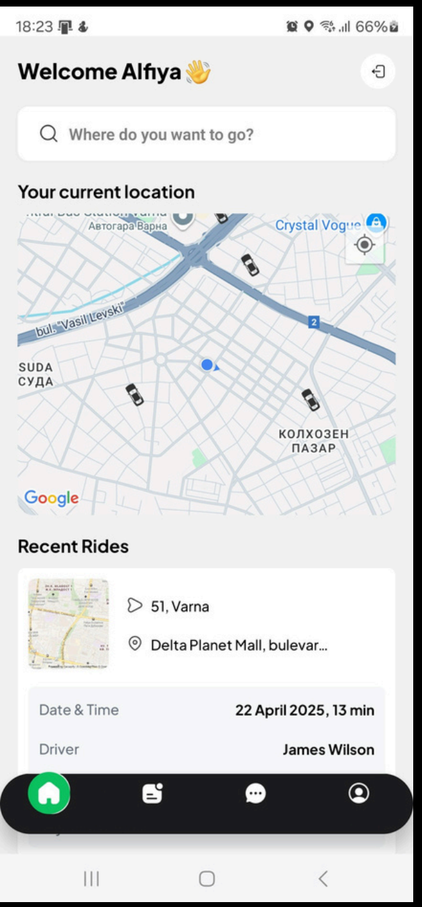



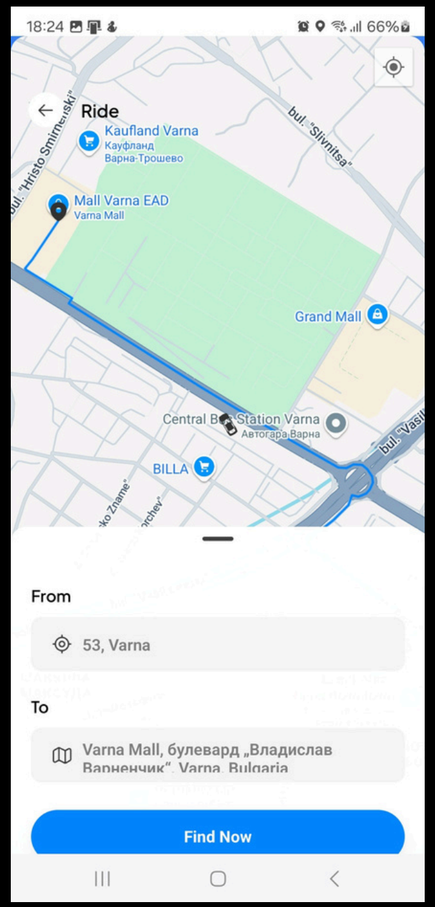



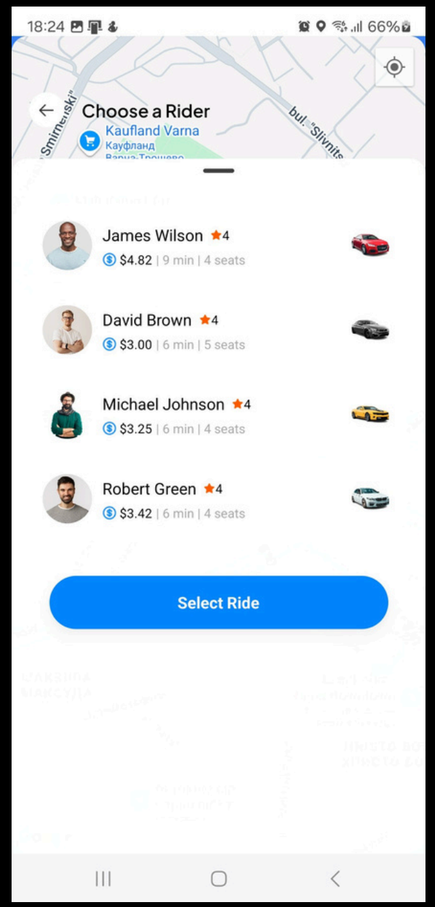



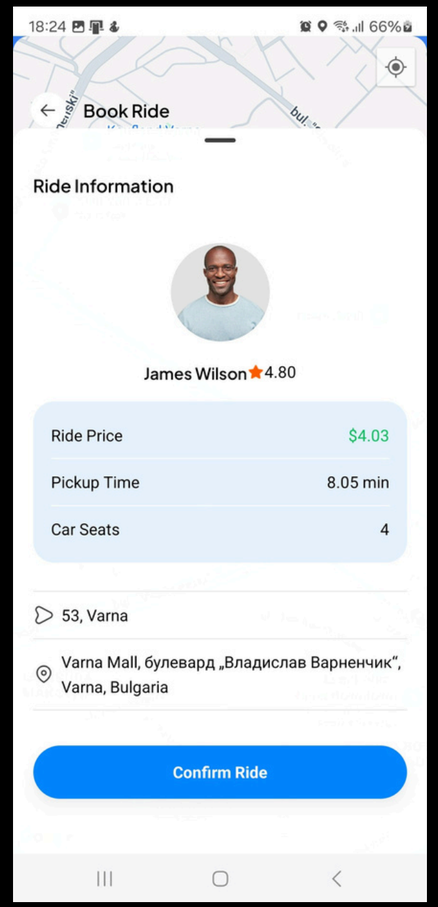



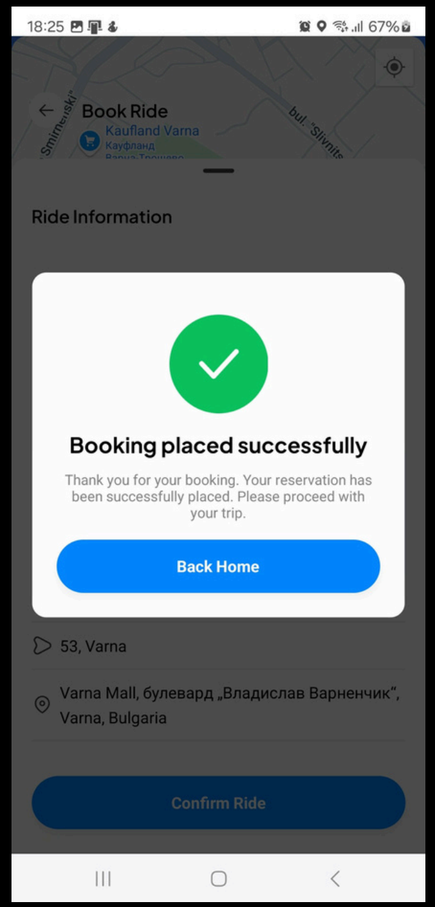
# MAIN SCREENS

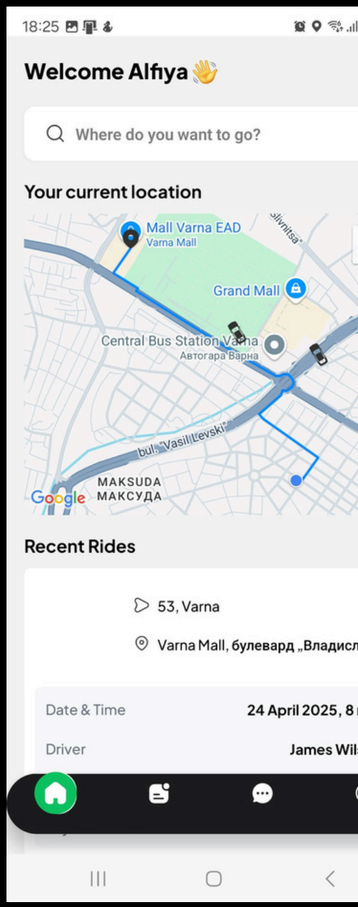


# All Rides

53, Varna

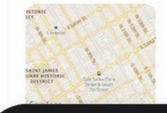
Varna Mall, булевард „Владисла...

| | |
|---|---|
| Date & Time | **24 April 2025, 8 min** |
| Driver | **James Wilson** |
| Car Seats | **4** |
| Payment Status | **Paid** |

51, Varna

Delta Planet Mall, bulevar...

| | |
|---|---|
| Date & Time | **22 April 2025, 13 min** |
| Driver | **James Wilson** |
| Car Seats | **4** |
| Payment Status | **Paid** |



San Francisco, CA, USA

Date & Time **21 April 2025, 59 min**



Ask something... SEND

**You:**

Where is varna. And must visit locations in varna for tourists

**AI:**

Varna is a city located on the **Black Sea coast of northeastern Bulgaria**.

It's the third-largest city in Bulgaria and a popular tourist destination known for its beautiful beaches, rich history, and vibrant culture.

Here are some must-visit locations in Varna for tourists:

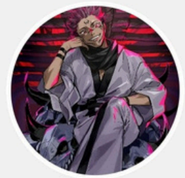
**Historical and Cultural Sites:**

* **The Roman Ruins:** Explore the well-preserved remains of the ancient Roman city of Varna, including the amphitheater, forum, and residential areas.
* **Varna Archaeological Museum:** Discover one of the largest and most important archaeological museums in Bulgaria, showcasing artifacts from the ancient Thracian, Greek, Roman, and Byzantine civilizations.
* **The Sea Casino:** This historic building, built in 1913, is a beautiful example of Art Nouveau architecture and offers stunning views of the Black Sea.
* **St. Nicholas Church:** Visit this iconic church, dating back to the 1st century AD, and admire its beautiful frescoes and mosaics.

**Beaches and Parks:**



# My profile



**First name**

Alfiya

**Last name**

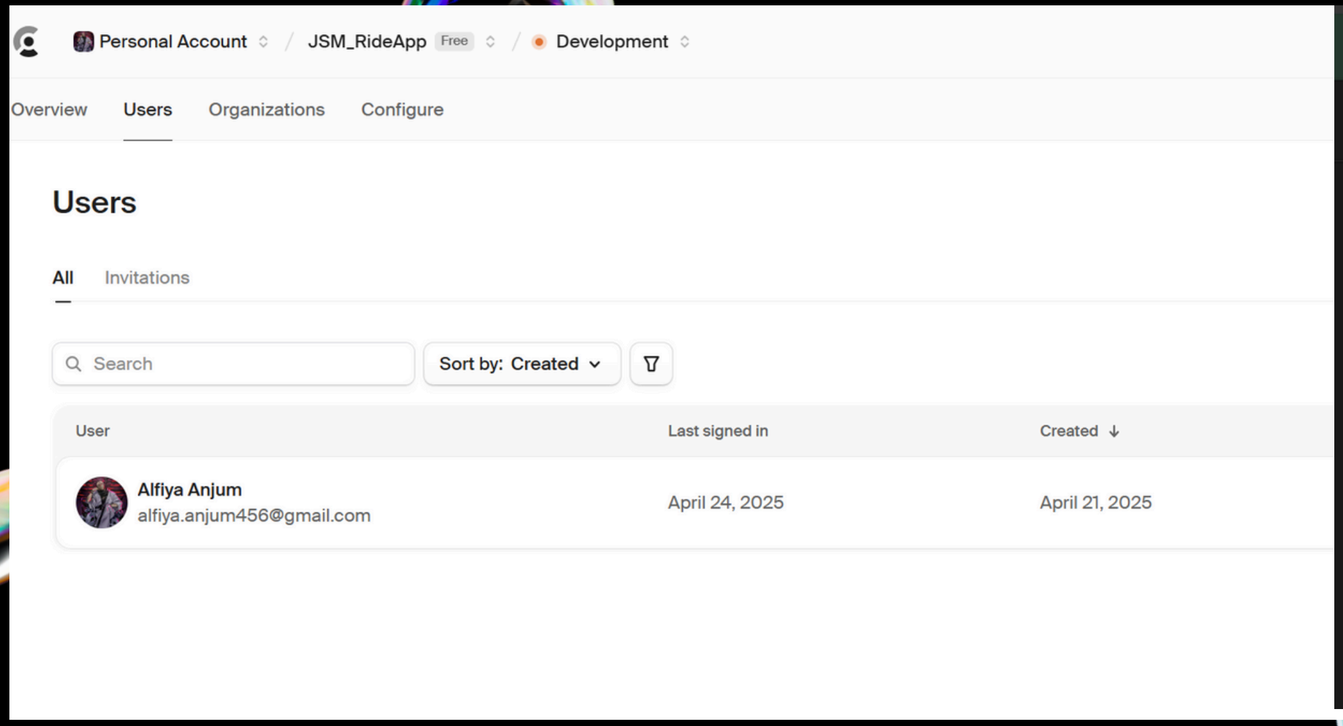
Anjum

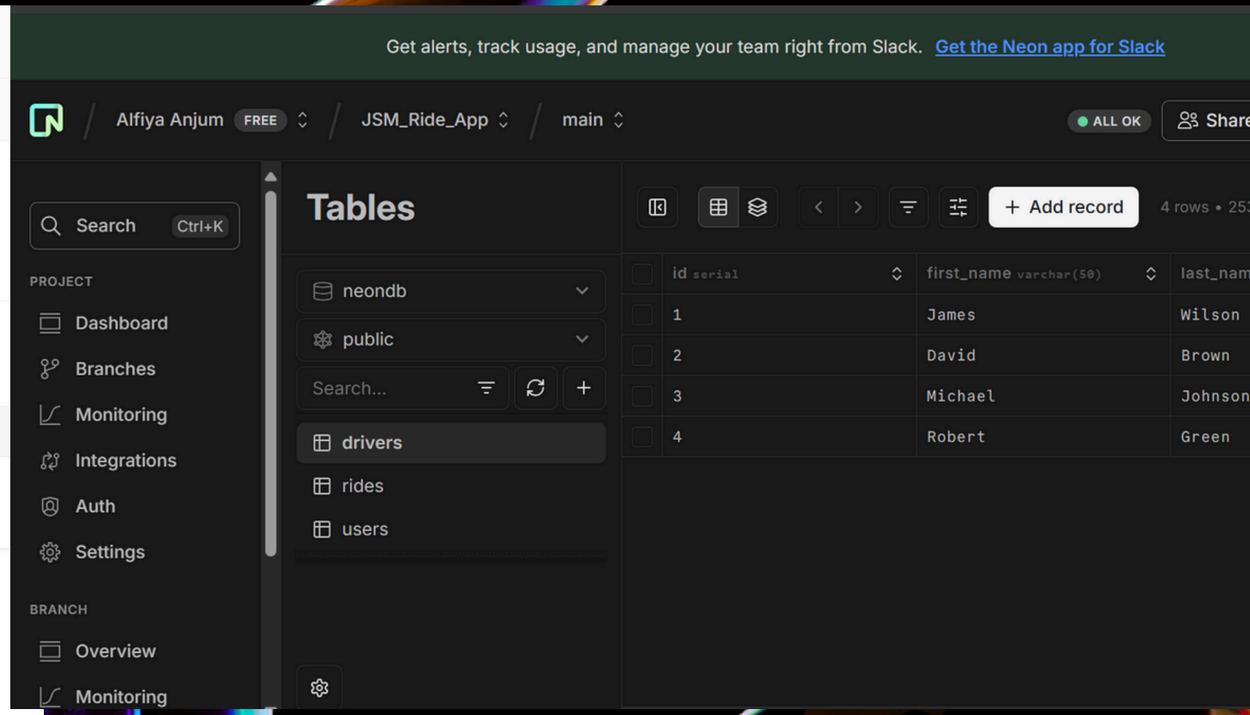
**Email**

alfiya.anjum456@gmail.com

# CLERK DASHBOARD

# NEON DB DATABASE



stripe

business sandbox
business

Search

Setup guide

# Balances

Manage payouts

Overview | All activity | Payouts

Add a bank account to pay out your BGN balance

+ Add bank account

BGN
лв0.00

USD
$2,659.98

## Balance summary

Incoming лв0.00

Available лв0.00

## Reports

Balance summary
Mar 2025

Payout reconciliation
Mar 2025

New business sandbox
New business

Search

Setup guide

# Transactions

+ Create payment | Analyze

All 29 | Succeeded 8 | Refunded 0 | Disputed 0 | Failed 0

Date and time | Amount | Currency | Status | Payment method | More filters | Export | Edit columns

| Amount | Payment method | Description | Customer | Date |
|---|---|---|---|---|
| $2.33 USD Succeeded | •••• 4242 | pi_3RHRdpFdBPK5JXfQ15tixquX | alfiya.anjum456@gmail.com | Apr 24, 3:25 PM |
| $1.75 USD Incomplete | — | pi_3RHMr0FdBPK5JXfQ19hXyzMC | alfiya.anjum456@gmail.com | Apr 24, 10:18 AM |
| $3.50 USD Succeeded | •••• 4242 | pi_3RGasfFdBPK5JXfQ0lrHD2aS | alfiya.anjum456@gmail.com | Apr 22, 7:04 AM |
| $6.00 USD Incomplete | — | pi_3RGRCvFdBPK5JXfQ1kQbBxGv | alfiya.anjum456@gmail.com | Apr 21, 8:44 PM |
| $29.00 USD Succeeded | •••• 4242 | pi_3RGDjPFdBPK5JXfQ0yoHr7x2 | alfiya.anjum456@gmail.com | Apr 21, 6:21 AM |
| $29.00 USD Incomplete | — | pi_3RGDjPFdBPK5JXfQ03vBB9aV | alfiya.anjum456@gmail.com | Apr 21, 6:21 AM |

Viewing 1–20 of 29 results

Previous | Next

- Home
- Balances
- Transactions
- Customers
- Product catalog

Shortcuts
- Terminal
- Payment Links

Products
- Payments
- Billing
- Reporting
- Developers


Google Gemini API Key Failure

- Encountered major issues while integrating Gemini AI – the API key was not working, causing the chatbot to crash.
- → Debugging took significant time and delayed progress.

Billing Issues with Google Cloud Console

- Faced problems setting up billing for Google AI Services.
- → Required contacting support and verifying payment settings.

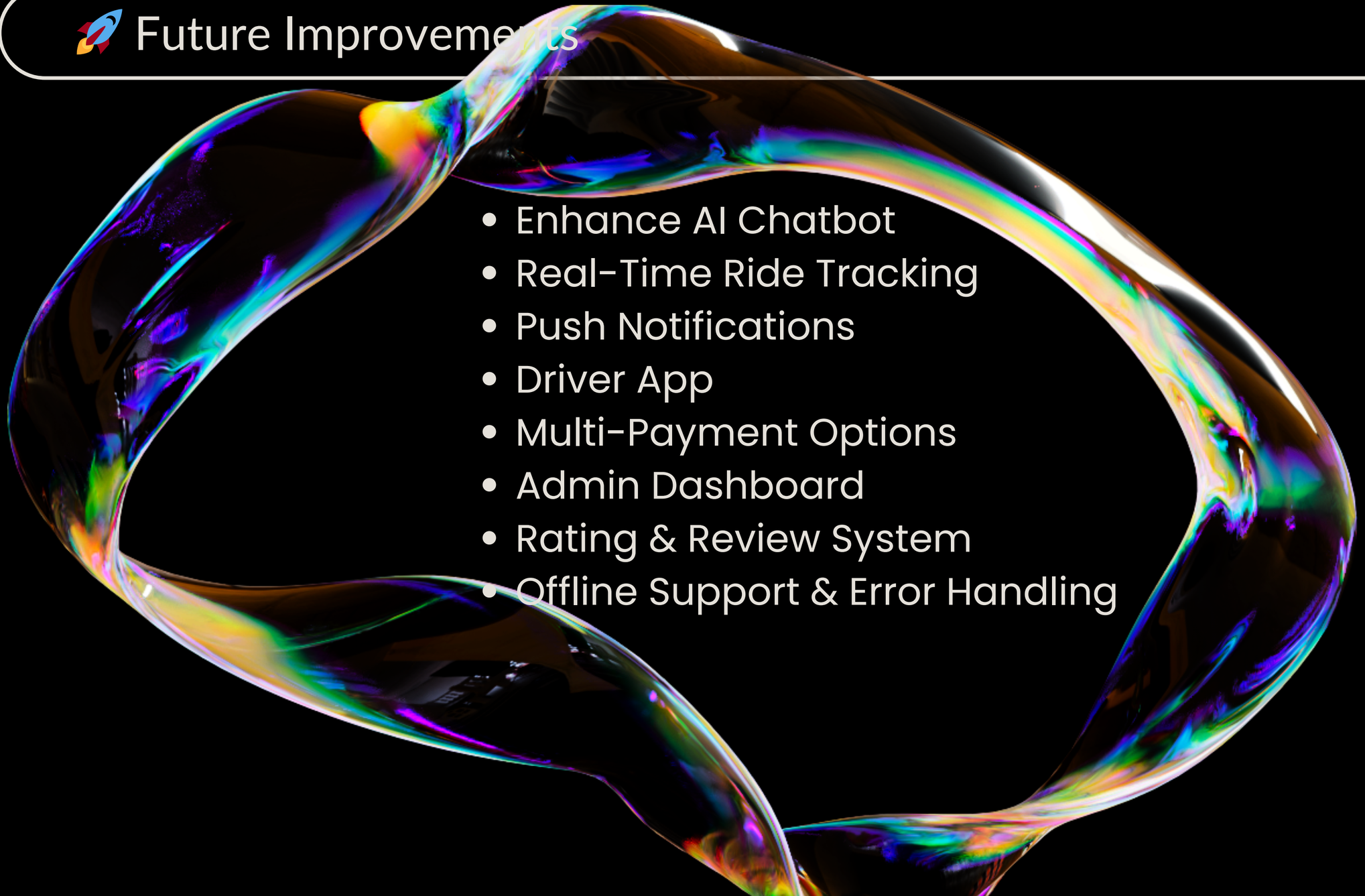
Integration Complexity

Testing Third-Party APIs

Expo Configuration & Permissions

Dealing with location permissions, navigation routes, and environment configs for Expo needed a lot of trial and error.

# 🚀 Future Improvements



- Enhance AI Chatbot
- Real-Time Ride Tracking
- Push Notifications
- Driver App
- Multi-Payment Options
- Admin Dashboard
- Rating & Review System
- Offline Support & Error Handling

# THANK YOU

for your time and attention

＼(＾▽＾)／

Presented by Alfiya Anjum