**ATTRACTION - Backend Part**

# Introduction

The ATTRACTION-BACKEND Django project is the core server-side component of a modern Mobility-as-a-Service (MaaS) platform designed to facilitate seamless trip planning, booking, and user management for urban transportation. This backend serves as the foundation for both web and mobile clients, providing secure, scalable, and flexible APIs that integrate multiple transport modes and external services.

The system is built to support both authenticated users (who have registered accounts) and anonymous users (who interact without logging in), ensuring inclusivity and ease of access. It leverages robust authentication mechanisms using JWT tokens, supports password management and account activation via email, and maintains user privacy and data integrity.

A key feature of the platform is its integration with external transit data providers, specifically the OpenTripPlanner (OTP) server, accessed via GraphQL. This allows users to plan trips across various modes—walking, bus, bicycle, scooter, and more—while receiving real-time itineraries and transit information. The backend also tracks user searches, bookings, and feedback, enabling personalized experiences and continuous service improvement.

Administrators benefit from a powerful Django admin interface, which provides conditional data views, meaningful labels, and the ability to manage users, trips, feedback, and favorite places efficiently. The backend is designed with best practices in error handling, security, and API documentation, ensuring reliability and maintainability for developers and operators.

Overall, the ATTRACTION-BACKEND project is a comprehensive solution for urban mobility, combining user-centric features, advanced integrations, and administrative controls to deliver a next-generation transportation experience.

# Main Functionalities

## User Management (accounts/)

* Registration: Users can register via (/api/auth/register/).
* Login: JWT-based authentication via (/api/auth/login/).
* Account Activation: Email-based activation using tokens.
* Password Reset: Request and confirm password reset via email.
* Profile Management: Retrieve and update user info and favorite places.
* Logout: Blacklists JWT refresh tokens for secure logout.

## Trip Planning & Search (activity/)

* Plan Trip: (/api/auth/plan-trip/) accepts trip details, queries external OTP GraphQL API, and returns itineraries grouped by mode (walk, bus, bicycle, scooter, other).
* Search History: Stores trip searches for both authenticated and anonymous users, linking anonymous searches to user accounts upon login.
* Track Activity: (/api/auth/track/) retrieves user search activity.
* Favorite Places: Users can add, update, and delete favorite places.
* Booking: Book trips and retrieve booking history.
* Feedback: Submit feedback via (/api/auth/feedback/).

## Transit Data (activity/)

* Stops: (/api/auth/stops/) fetches all transit stops from the OTP server.
* Station Schedules: (/api/auth/station/<stop\_id>/) returns bus arrivals/departures for a given stop.

## Admin Features

* User Table: Shows all registered users with their ID, email, type (worker or student or other), and fiscal code.
* Search Table: Displays user ID/session key, origin/destination coordinates, trip date, requested time, and modes.
* Conditional Columns: Shows user ID for authenticated searches, session key for anonymous.
* Read-only Fields: (Can be configured) to prevent editing of trip date/requested time.
* Favorite Places Table: Shows user ID, address, type, latitude, longitude.
* Feedback Table: Shows the feedback of each user.

## Security & Error Handling

* JWT Authentication: Uses “rest\_framework\_simplejwt” for secure token management.
* Token Blacklisting: Prevents reuse of refresh tokens after logout.
* Consistent Error Responses: All APIs return structured JSON with HTTP status codes and messages.
* Uses standard HTTP codes: 200, 201, 204, 400, 401, 403, 404, 409, 415, 422, 429, 500, 503.
* Handles External API Failures: Returns 503 Service Unavailable if OTP server is unreachable.

## Templates

* stop\_schedule.html: Displays bus schedules for a stop. It’s used as a URL for a QR code at a station to show the trips of this station.

# API Endpoints

* /api/auth/register/ — Register new user
* /api/auth/login/ — Login and obtain JWT tokens
* /api/auth/token/refresh/ — Refresh JWT access token
* /api/auth/activate/<uidb64>/<token>/ — Activate account
* /api/auth/password-reset/ — Request password reset
* /api/auth/password-reset-confirm/<uidb64>/<token>/ — Confirm password reset
* /api/auth/search/ — Search history
* /api/auth/places/ — Manage favorite places
* /api/auth/track/ — Track user activity
* /api/auth/booking/ — Book trips
* /api/auth/feedback/ — Submit feedback
* /api/auth/profile/ — Get/update user profile
* /api/auth/logout/ — Logout
* /api/auth/plan-trip/ — Plan a trip (OTP integration)
* /api/auth/stops/ — Get all stops
* /api/auth/station/<stop\_id>/ — Get station schedule

# API-Specific Error Codes

1. /api/auth/register/ (POST)

* 201: User registered
* 400: Invalid input, missing fields, weak password
* 409: Email/username already exists

1. /api/auth/login/ (POST)

* 200: Login successful
* 400: Invalid credentials, malformed request
* 401: Authentication failed

1. /api/auth/token/refresh/ (POST)

* 200: Token refreshed
* 400: Invalid/expired refresh token
* 401: Not authenticated

1. /api/auth/activate/// (GET)

* 200: Account activated
* 400: Invalid token or user ID
* 403: User is banned
* 404: User not found

1. /api/auth/password-reset/ (POST)

* 200: Reset link sent
* 400: Invalid email format
* 404: Email not found (may still return 200 for security)

1. /api/auth/password-reset-confirm/// (POST)

* 200: Password reset successful
* 400: Invalid/expired token, weak password
* 403: User is banned

1. /api/auth/search (GET/POST)

* 200: Success
* 400: Invalid input
* 401: Not authenticated
* 404: Search not found

1. /api/auth/places (GET/POST)

* 200: Success
* 201: Place added
* 400: Invalid input
* 401: Not authenticated
* 404: Place not found

1. /api/auth/track (GET)

* 200: Success
* 400: Missing 'id' parameter
* 404: No search activity found

1. /api/auth/booking (GET/POST)

* 200: Success
* 201: Booking created
* 400: Invalid input
* 401: Not authenticated
* 409: Booking conflict

1. /api/auth/feedback (POST)

* 201: Feedback submitted
* 400: Invalid input

1. /api/auth/profile (GET/PUT)

* 200: Success
* 400: Invalid input
* 401: Not authenticated
* 404: User not found
* 415: Unsupported media type

1. /api/auth/places/ (PUT)

* 200: Success
* 400: Invalid input
* 401: Not authenticated
* 404: Place not found

1. /api/auth/logout (POST)

* 205: Successfully logged out
* 400: Missing/invalid refresh token
* 401: Not authenticated

1. /api/auth/plan-trip/ (POST/GET)

* 200: Success
* 400: Invalid input, validation error
* 401: Not authenticated (if required)
* 503: External API/network error

1. /api/auth/stops/ (GET/POST)

* 200: Success
* 400: Invalid input
* 503: External API/network error

1. /api/auth/station// (GET/POST)

* 200: Success
* 400: Invalid input
* 404: Stop not found
* 503: External API/network error