# \*\*Technical Report: Integrating Django REST API with Flutter for a Car Services App\*\*

## \*\*1. Introduction\*\*

This report outlines a solution for building a \*\*car services platform\*\* with:

- A \*\*Django backend\*\* (hosted on \*\*cPanel\*\*)

- A \*\*Flutter mobile app\*\*

- \*\*Shared SQL database\*\* (MySQL/PostgreSQL)

- \*\*Authentication\*\* (Email/Password + Google OAuth)

---

## \*\*2. System Architecture\*\*

### \*\*Components\*\*

| Component | Role |

|--------------------|----------------------------------------------------------------------|

| \*\*Django (cPanel)\*\* | Backend API, database management, authentication |

| \*\*Flutter App\*\* | Mobile interface for users to book services |

| \*\*MySQL DB\*\* | Stores user accounts, service requests, and bookings |

| \*\*REST API\*\* | Connects Flutter app to Django backend |

### \*\*Data Flow\*\*

1. \*\*User signs up\*\* (via email or Google) → \*\*Flutter sends request to Django API\*\*

2. \*\*Django validates & saves data\*\* → \*\*Returns auth token\*\*

3. \*\*User books a service\*\* → \*\*Flutter sends request to Django\*\*

4. \*\*Django updates database\*\* → \*\*Website & app sync in real-time\*\*

---

## \*\*3. Authentication Methods\*\*

### \*\*Option 1: Pure Django REST API\*\*

✅ \*\*Pros\*\*

- Full database control

- No Firebase dependency

- Works on cPanel

🔧 \*\*Implementation\*\*

1. \*\*Packages Used\*\*:

- `django-rest-framework` (API endpoints)

- `django-allauth` (Google OAuth)

- `django-rest-auth` (Token auth)

2. \*\*Flutter Integration\*\*

- HTTP requests to `/api/auth/login/`, `/api/auth/google/`

- Store Django-generated \*\*JWT/Token\*\* for session management

### \*\*Option 2: Firebase + Django Hybrid\*\*

🔥 \*\*Pros\*\*

- Easier Google Auth setup

- Firebase handles mobile auth smoothly

🔧 \*\*Implementation\*\*

1. \*\*Flutter uses Firebase Auth\*\* for Google login.

2. \*\*Firebase UID sent to Django\*\* → Django creates a matching user.

3. \*\*Custom tokens\*\* sync Firebase & Django auth.

---

## \*\*4. Database Design (MySQL Example)\*\*

### \*\*Key Tables\*\*

| Table | Purpose |

|-------------------|----------------------------------------|

| `users` | Stores user accounts (email, password) |

| `service\_requests`| Car service bookings (user\_id, car\_model, service\_type) |

| `auth\_tokens` | JWT tokens for Flutter app sessions |

---

## \*\*5. Deployment (cPanel)\*\*

### \*\*Steps\*\*

1. \*\*Upload Django project\*\* via cPanel Python App.

2. \*\*Configure MySQL DB\*\* in cPanel.

3. \*\*Set up REST API endpoints\*\* (`/api/auth/`, `/api/services/`).

4. \*\*Secure API\*\* with HTTPS (SSL certificate).

---

## \*\*6. Flutter App Workflow\*\*

```mermaid

sequenceDiagram

Flutter->>Django: POST /api/auth/google/ (Google token)

Django->>Google: Verify token

Google-->>Django: Valid token + user data

Django->>MySQL: Save user

Django-->>Flutter: Return auth token

Flutter->>Django: POST /api/services/ (New booking)

Django->>MySQL: Save booking

Django-->>Flutter: Success response

```

---

## \*\*7. Security Considerations\*\*

- \*\*Use HTTPS\*\* for all API calls.

- \*\*Sanitize inputs\*\* to prevent SQL injection.

- \*\*Rate-limit API\*\* to prevent brute-force attacks.

- \*\*Store tokens securely\*\* in Flutter (flutter\_secure\_storage).

---

## \*\*8. Conclusion & Recommendation\*\*

### \*\*Best Approach\*\*

✅ \*\*Use Django REST API\*\* if:

- You want \*\*full control\*\* over the database.

- You’re \*\*hosting on cPanel\*\*.

- You prefer \*\*no third-party dependencies\*\*.

🔥 \*\*Use Firebase + Django\*\* if:

- You want \*\*easier Google Auth\*\* in Flutter.

- You need \*\*real-time updates\*\* (Firestore).

### \*\*Next Steps\*\*

1. Set up Django REST API on cPanel.

2. Design MySQL tables for car services.

3. Implement Flutter HTTP client for API calls.

Would you like a \*\*step-by-step deployment guide\*\* for cPanel? 🚀