🚀 Testing Easebuzz with Postman

This guide details how to test the **Easebuzz Payment Integration** (Deposits & Payouts) using Postman.

🛠️ 1. Postman Setup

A. Environment Variables

Create a new Environment in Postman (e.g., `DriversKlub Local`) and add:

Variable	Current Value
`base_url`	`http://localhost:5000`
`driver_token`	*(Leave blank, we will fill this after login)*
`admin_token`	*(Leave blank, we will fill this after login)*
`rentalPlanId`	*(Leave blank, auto-filled by "Get Plans")*

🔑 2. Authentication (Get Tokens)

You need to login to get the `Authorization` tokens.

Driver Login

- **Method**: `POST`
- **URL**: `{{base_url}}/auth/verify-otp`
- **Body** (JSON):

```
{
    "phone": "+919999900002",
    "otp": "000000",
    "verifiedKey": "pass"
}
```

- **Action**: Copy `accessToken` from response -> Set as `driver_token`.

Admin Login

- **Method**: `POST`
- **URL**: `{{base_url}}/auth/verify-otp`
- **Body** (JSON):

```json
{
    "phone": "+919999900001",
    "otp": "000000",
    "verifiedKey": "pass"
}
```

- **Action**: Copy `accessToken` from response -> Set as `admin_token`.

💳 3. Test: Initiate Deposit (Driver Side)

This mimics a driver adding money to their wallet via Easebuzz.

- **Method**: `POST`
- **URL**: `{{base_url}}/payment/deposit`
- **Headers**:
  - `Authorization`: `Bearer {{driver_token}}`
- **Body** (JSON):

```json
{
    "amount": 100.00
}
```

- **Response**:

```json
{
    "data": {
        "paymentUrl": "https://testpay.easebuzz.in/pay/..."
    }
}
```

- **Testing Action**:
  1. Ctrl+Click the `paymentUrl`.
  2. It opens the Easebuzz Payment Gateway in your browser.
  3. Use the **Test Cards** from `PAYMENT_SYSTEM_DOCUMENTATION.md` (Section 6) to complete the payment.
  4. Upon success, you will be redirected to the configured `PAYMENT_SUCCESS_URL`.

� 4. Test: Rental Model (Driver Side)

This mimics a driver viewing and purchasing a subscription plan.

Step 1: Get Plans

- **Method**: `GET`
- **URL**: `{{base_url}}/payment/rental/plans`
- **Headers**:
  - `Authorization` : `Bearer {{driver_token}}`
- **Action**:
  - Send the request.
  - The **Postman Test Script** will automatically capture the first Plan ID (`rentalPlanId`) from the response.

Step 2: Purchase Plan

- **Method**: `POST`
- **URL**: `{{base_url}}/payment/rental`
- **Headers**:
  - `Authorization` : `Bearer {{driver_token}}`
- **Body** (JSON):

```
{
    "rentalPlanId": "{{rentalPlanId}}"
}
```

- **Response**: Returns a `paymentUrl` (similar to Deposit). Follow the link to pay.

� 💸 5. Test: Bulk Payout (Admin Side)

This mimics an admin uploading a CSV to pay drivers.

- **Method**: `POST`
- **URL**: `{{base_url}}/payment/admin/bulk-payout`
- **Headers**:
  - `Authorization` : `Bearer {{admin_token}}`
- **Body** (form-data):
  - Key: `file` | Type: `File` | Value: `[Select a test.csv file]`
- **Sample CSV Content (`test.csv`)**:

```
phone,amount,payoutCycle
+919999900002,50,Test Payout
```

- **Response**:

```
{
    "total": 1,
    "success": 1,
    "amountDisbursed": 50
}
```

🧪 6. Verification

After initiating a deposit or payout:

- **Check Transactions (Driver)**:
  - `GET {{base_url}}/payment/transactions`
  - Header: `Bearer {{driver_token}}`
- **Check Server Logs**:
  - Look for `Easebuzz Response:` or `Webhook received` logs in your terminal.

📷 7. Test: Generate Vehicle QR

This creates a Virtual Account and QR code for a vehicle.

- **Method**: `POST`
- **URL**: `{{base_url}}/payment/vehicle/:vehicleId/qr`
- **Headers**:
  - `Authorization`: `Bearer {{admin_token}}`
- **Response**:

```
{
    "success": true,
    "data": {
        "virtualAccountNumber": "TESTMH12AB1234",
        "qrCodeBase64": "mock_qr_base64_string",
        "upiId": "MH12AB1234@easebuzz"
    }
}
```

> ***Note (Test Mode)**: Since the Easebuzz Wire API is restricted, the **Test Environment uses a Mock Provider**. It returns a valid-looking dummy response so you can test the frontend display and downstream logic without needing a whitelisted IP.*