API Documentation: MySQL RDS Lab Project

# 1. Introduction

This document describes the RESTful API built for the MySQL RDS Lab project. The API is designed to interface with a MySQL database hosted on AWS RDS and provides endpoints for retrieving customer analytics, sales metrics, and product insights.

# 2. Base URL

Development Base URL:  
http://localhost:3000/

Production Base URL:  
(To be defined upon deployment)

# 3. Authentication

Currently, this API is publicly accessible and does not implement authentication. Future versions may include token-based (JWT) or OAuth2 authentication mechanisms.

# 4. Endpoints

## 4.1 GET /top-customers

Description:  
Retrieves the top 5 customers sorted by total spend.

Response Example:

[{"customer\_id": 1, "full\_name": "John Doe", "total\_spend": 1580.50}]

## 4.2 GET /monthly-sales

Description:  
Returns monthly sales figures for orders with status 'shipped' or 'delivered'.

Response Example:

[{"month": "2025-04", "total\_sales": 4200.75}]

## 4.3 GET /products-never-ordered

Description:  
Lists products that have never been ordered by any customer.

Response Example:

[{"product\_id": 10, "product\_name": "Gaming Keyboard"}]

## 4.4 GET /avg-order-value

Description:  
Provides the average order value, grouped by customer country.

Response Example:

[{"country": "Germany", "average\_order\_value": 250.45}]

## 4.5 GET /frequent-buyers

Description:  
Returns customers who have placed more than one order.

Response Example:

[{"customer\_id": 5, "full\_name": "Jane Smith", "order\_count": 3}]

# 5. Error Handling

All API errors are returned in a consistent format:

{"error": "Description of the error"}

Common HTTP Status Codes:  
- 200 OK: Request was successful  
- 400 Bad Request: Input validation failed  
- 404 Not Found: The requested resource could not be found  
- 500 Internal Server Error: An unexpected error occurred on the server

# 6. Codebase Structure

|  |  |
| --- | --- |
| Folder/File | Description |
| app.js | Initializes Express and imports routes |
| routes/ | Modular route files for each endpoint |
| queries/ | Raw SQL query definitions |
| db.js | Database connection logic using MySQL pool |

# 7. Naming & Design Conventions

Endpoints: Use lowercase and kebab-case (e.g., /top-customers)  
Methods: Follows RESTful standards (GET, POST, etc.)  
Responses: Always returned in JSON format

# 8. Future Enhancements

- Implement token-based authentication (JWT)  
- Add pagination, filtering, and sorting to endpoints  
- Integrate Swagger (OpenAPI 3.0) for automatic API documentation  
- Deploy via containerized architecture (Docker + ECS/Fargate)