**Environment Setup**

**1) Install PostgreSQL**

Set up the database with the following credentials:

* **Username:** postgres
* **Password:** postgres

Or if somebody has created some other credential of prostgress db then we have to change credential in following path:

1. BACKEND\loganalyser\src\main\resources
2. BACKEND\ loglistenercollector\src\main\resources
3. BACKEND\ logwarnpersistor\src\main\resources
4. BACKEND\ loginfopersistor\src\main\resources
5. BACKEND\ logdebugpersistor \src\main\resources
6. BACKEND\ logerrorpersistor \src\main\resources

And change following properties:

spring.datasource.username=<your username>

spring.datasource.password=<your password>

**2) Create Database**

Run the following command in PostgreSQL:

CREATE DATABASE logdata;

**Log Details**

Logs will be created at the following path:

C:\Users\<username>\log\app-logs.log

**Example:**

C:\Users\rakesh\log\app-logs.log

**Starting Microservices**

Please start the microservices in the following order:

1. **logListenerCollector**
2. **logGenerator**
3. **logInfoPersistor**
4. **logDebugPersistor**
5. **logWarnPersistor**
6. **logErrorPersistor**
7. **logAnalyser**

**Testing**

**1. Testing Log Generator**

When the logGenerator application starts, it will create a log folder and file:

C:\Users\<username>\log\app-logs.log

Example:

C:\Users\rakesh\log\app-logs.log

**2. Testing Log Listener Collector**

* When this application starts, it populates the table log\_data.
* Run the following query to verify:

SELECT \* FROM log\_data;

* Data will appear **only if logGenerator microservice is running**.

**3. Testing Log Analyser**

**a) Fetch all logs**

**Request:**

GET http://localhost:9097/logs

**Response Example:**

[

{

"id": 1,

"timestamp": "2025-09-04T19:29:27.878941",

"message": "Cache refreshed successfully",

"level": "ERROR"

},

{

"id": 2,

"timestamp": "2025-09-04T19:29:57.653724",

"message": "User login successful",

"level": "WARN"

},

{

"id": 3,

"timestamp": "2025-09-04T19:30:27.663144",

"message": "External API call failed",

"level": "WARN"

}

]

**b) Error type and count**

**Request:**

GET http://localhost:9097/logs/count

**Response Example:**

{

"ERROR": 23,

"INFO": 17,

"DEBUG": 22,

"WARN": 23

}

**c) Line graph data**

**Request:**

GET http://localhost:9097/logs/linegraph

**Response Example:**

[

{

"timestamp": "2025-09-04T19:25",

"ERROR": 1,

"WARN": 1,

"INFO": 0,

"DEBUG": 0

},

{

"timestamp": "2025-09-04T19:30",

"ERROR": 3,

"WARN": 4,

"INFO": 0,

"DEBUG": 3

},

{

"timestamp": "2025-09-04T19:35",

"ERROR": 1,

"WARN": 2,

"INFO": 2,

"DEBUG": 5

},

{

"timestamp": "2025-09-04T19:40",

"ERROR": 4,

"WARN": 4,

"INFO": 2,

"DEBUG": 0

}

]