AeroAspire

SDE Intern

Goutham V

Week 4 – Day2 (14th October)

Task:

Connect Flask to SQLite or MySQL; implement CRUD using raw SQL or ORM (SQLAlchemy)

Reflections,

1. What is ORM, what are its advantages & disadvantages?

ORM (Object Relational Mapping) is a tool that allows developers to interact with the database using Python objects instead of writing raw SQL queries.

Advantages:

- It simplifies database operations — you can use Python classes instead of SQL.
- It reduces human errors from writing long queries.
- It helps maintain cleaner and more readable code.

Disadvantages:

- It can be slower than raw SQL for large, complex queries.

- It hides what's happening underneath, so debugging SQL performance issues can be harder.

2. How does parameterized query prevent SQL injection?

A **parameterized query** uses placeholders (`?` or `:param`) for values instead of inserting user input directly into SQL statements.
This ensures that user input is treated as **data**, not as part of the SQL command — preventing malicious users from injecting harmful SQL code.
For example:

```
cursor.execute("SELECT * FROM tasks WHERE id =
?", (task_id,))
```

Here, even if `task_id` contains special characters, it won't break or change the SQL command.

3. What is the flow from request → ORM / SQL → DB → return result → commit / rollback?

The process works like this:

1. *Client sends a request* (for example, POST a new task).
2. *Flask receives the request* and processes it in a route handler.
3. The handler uses *ORM or SQL commands* to interact with the database.
4. The *database executes the command* and sends back a result.
5. If everything is fine, Flask *commits the transaction* (saves changes).
6. If something goes wrong, it *rolls back* the transaction to prevent bad data.

7. Finally, a *response is returned* to the client (for example, “Task created successfully”).