Software Testing:

Diagram:
https://drive.google.com/file/d/1eigV7XVn9qVNxT5wrXGLhkCtsjFpV_Br/view?usp=sharing

Layered Arch.

1. Presentation Layer (UI)
2. Business Logic (APIs)
3. Database (Tables)

Framework:

1. TypeScript
2. Next.js

Presentation Layer Components:

1. General
    - Login
    - Register
2. Admin
    - Manage Orders
    - Manage bookings
    - Manage Products
3. Client
    - View Menu
    - View Order(cart)
    - Booking

Business Logic:

1. Admin
    - Get Bookings
    - Update Booking
    - Delete Booking
    - Update order status
    - Get order
    - Delete order
    - Get product
    - Delete product
    - Create product
    - Update product
2. Client
    - Get products (view menu)
    - Get products in cart
    - Delete product in cart
    - Update product in cart

- Create order
- Delete order

Database:

1. Users (ID, Name, isAdmin, username, password, PhoneNo)
2. Orders (ID, items [], Address, status, user_id)
3. Products (ID, Name, price, description, image)
4. Booking (ID, customer name, no of people, Date, Time, customer_id)

Why layered architecture?

Layered architecture is a type of design pattern used to split code into layers, each handling and focusing on a specific aspect of the code. It applies the concept of separation of concerns, with each layer focusing on its own purpose: the presentation layer handles the UI and provides an abstract idea of the website's webpages, the business layer handles logic such as APIs and CRUD operations for these webpages, and finally, the database layer contains the databases that handle and store the operations of the website.

As each layer interacts with the adjacent layer, it offers scalability, which is especially useful for restaurants receiving a large amount of load from orders. The independence of the layers allows for easy scaling. Additionally, the separation of layers facilitates testing, as each layer can be tested and debugged independently. The separation of concerns also offers the advantage of changing the layer while keeping the others intact, with changes affecting only the respective layer. This makes these layers reusable with slight modifications for similar-goal projects, making the design pattern maintainable, reusable, flexible, and efficient. Teamwork can be enhanced since developers can work on each layer separately and at the same time, dividing the workload efficiently. For these reasons, we choose layered architecture for our restaurant website.

To compare it with other options, microservice architecture uses independent and relatively small modules, each targeting a specific goal in the application, which communicate via APIs. Scalability and modularity are present due to service independence, allowing ease of development and unit testing. However, it introduces overhead in communication between these services, which increases complexity. For a simple restaurant website, this overhead is unnecessary, and the idea of multiple services isn't required.

Monolithic architecture, on the other hand, oversimplifies the process by packing the whole functionality of the three layers into one unit. This simplifies resource management initially, as it is low in complexity, and speeds up the development process, enabling developers to work on the whole unit simultaneously and removing inter-unit communication. However, this simplicity causes problems with scalability due to the lack of modularity and tight coupling. All parts are strongly integrated, and a single component, such as adding a restaurant offer or a new product to the menu, cannot be easily done. This change would require propagating and reflecting it across other parts of the single unit. So, although it can be easier, simpler, and quicker initially, in the long run, it makes bug fixes, updates, and version releases more complex and difficult to maintain.

Bonus: (Use Case Diagram)

https://drive.google.com/file/d/1nYxGGLK60bo1conyV1zRmxICvYvivxsT/view?usp=sharing

Names and IDs:

Hagar Adel 10005448

Farah Hani 10003337

Jana Tamer 10004798

Shahd Ahmed 10003660

Lojine Ahmed 10006100