Institute of Space Technology, Islamabad

<u>OPEN ENDED LAB</u>

Web Technologies

COURSE INSTRUCTOR

Ms. Maryam

SUBMITTED BY

Name: Ariba Shuaib

Reg no: 230201075

Computer Science Department

Backend Technologies & Server-Side Development

Project Name: Culinary Archive – A Full-Stack Recipe Management and Generation Web Application

Introduction

"Culinary Archive", a MERN-stack web application that allows users to register, authenticate, and manage recipes with image uploads. The focus of this lab is to understand backend systems end-to-end and justify the choice of technologies used in the project.

Backend System Architecture (Exploratory Task)

Client–Server–Database Architecture

The Culinary Archive project follows a three-tier architecture:

- **Client (Frontend – React.js):** Handles user interface, form input, and API requests.
- **Server (Backend – Node.js & Express.js):** Handles business logic, authentication, validation, and secure routing.
- **Database (MongoDB):** Stores persistent data such as users and recipes.

Why direct client–database access is unsafe:
Direct access would expose database credentials, allow unauthorized data manipulation, and bypass authentication and validation logic. Using a backend server ensures security, controlled access, and proper data handling.

Server-Side Programming

Backend Runtime Selection: Node.js

Node.js is used as the backend runtime because it allows JavaScript to run outside the browser. It uses a non-blocking, event-driven model, making it suitable for scalable web applications.

Reasons for choosing Node.js:

- High performance and scalability
- Same language (JavaScript) for frontend and backend
- Strong ecosystem for REST APIs

Backend Framework: Express.js

Express.js is used to build the backend server and APIs. It simplifies routing, middleware usage, and request handling.

Why Express.js was chosen:

- Lightweight and flexible
- Easy API creation
- Strong middleware support for authentication and logging

REST APIs & Routing (Exploratory Task)

The project uses RESTful APIs to expose backend functionality. Each API corresponds to a specific resource and uses appropriate HTTP methods.

Examples of REST APIs in Culinary Archive:

- POST /register – Create a new user account
- POST /login – Authenticate user
- GET /recipes – Fetch user recipes
- POST /recipes – Add a new recipe
- PUT /recipes/:id – Update a recipe
- DELETE /recipes/:id – Delete a recipe

Why authentication is required:
APIs that access or modify user-specific data require JWT-based authentication to ensure only authorized users can perform actions.

Database Concepts & Justification

SQL vs NoSQL (Exploratory Comparison)

For the Culinary Archive project, a NoSQL database **(MongoDB)** is used.

Justification:

- Recipe data has flexible structure (ingredients list, images, instructions)
- MongoDB supports JSON-like documents, making it ideal for such data
- Better scalability compared to traditional SQL databases

For applications like banking or accounting, SQL databases are preferred due to strict relational constraints, whereas MongoDB is better suited for content-based systems like this project.

Database Integration with Backend

MongoDB is connected to the backend using Mongoose, an Object Data Modeling (ODM) library.

Why Mongoose is used:

- Schema definition and validation
- Cleaner interaction with MongoDB
- Simplified CRUD operations

Database connection logic location:
The database connection is placed in a separate configuration file to maintain modularity and improve maintainability.

Risk of hardcoded credentials:
Hardcoding database credentials can lead to security breaches. Environment variables are used instead to protect sensitive information.

CRUD Operations Mapping (Exploratory Task)

CRUD operations are a core part of the backend system:

- **Create:** Add new users and recipes
- **Read:** View saved recipes
- **Update:** Edit recipe details
- **Delete:** Remove unwanted recipes

Operations requiring logging:
Authentication events and data modification operations should be logged to monitor system usage and detect suspicious activity.

Full Stack Integration (Lab 12)

The Culinary Archive project demonstrates full-stack integration using the MERN stack.

Flow:
React UI → API Request → Express Server → MongoDB → Response → React UI

Axios is used on the frontend to consume backend APIs. Protected routes ensure secure communication between frontend and backend using JWT authentication.

Conclusion

The Culinary Archive project successfully applies concepts of server-side programming, RESTful API design, database integration, authentication, and CRUD operations. The chosen backend technologies provide scalability, security, and maintainability, making them suitable for real-world web applications.