PES UNIVERSITY, BANGALORE

Department of Computer Science and Engineering

B. Tech (CSE) – 5th Semester – Aug-Dec 2024

UE22CS341A - Software Engineering

PROJECT PLAN
For
Restaurant Database Management System

Team #: ____3___

PES2UG22CS042	Aathil Nishad
PES2UG22CS042	Ajaybir Singh

General Instructions (Delete these instructions from the document before submission) :

- Formatting Guidelines : Submit in pdf format
- Naming convention : **Plan SRN1-SRN2-SRN3-SRN4** (Write SRNs in ascending order)
- In case of any confusion regarding what to fill under each heading, read this Format. It has all been explained.
- Don't forget to fill the Project Title on the **first page** as well as the **footer**

Life-cycle followed (PES2UG22CS011)

- Project Initiation:
  - o Project Definition:
    - Clearly define the project scope and objectives, including the specific functionalities (e.g., table management, order management, billing)
  - o Stakeholder Identification:
    - Identify key stakeholders, including restaurant owners, managers, staff, and customers.
    - Determine their roles, responsibilities, and how they will be involved throughout the project. o Requirements Gathering: Collect and document functional and non-functional requirements from stakeholders.
  - o Feasibility Analysis:
    - Assess the technical and operational feasibility of the project.

  - o The project requirements for the Restaurant Management System are characterized by high demands for product functionality, high process specifications, and relatively low resource requirement.
  - o The Restaurant Management System requires various functions such as
    - Table Management
    - Order Management
    - Reservation Manager Menu Management
    - Inventory Management
    - Billing and Payment Processing
    - Employee Management o The deployment model used is Iterative model.
    - **Frequent Requirement Changes:**

- Restaurant management systems often need to adapt to dynamic business needs, such as new features, customer preferences, and changes in the restaurant’s workflow and menu changes.

 Complex and Interconnected Features:

- Functions like table management, order management, inventory tracking, and billing are interconnected.
- The iterative model let us focus on specific parts of the system and gradually introduce other features.

 Risk Reduction:

- In the restaurant business, processes like payment handling or inventory management are crucial and risky.
- Handling potential issues with usability, system security, or resource management before finalizing the product can be done through iterative model.

 Flexibility in Deployment:

- With the Iterative model, we can deploy partial versions of the website (such as a version that only manages orders and payments) while still working on other features, making the development more flexible and adaptable.

○ Initial Project Planning:

- Our first focus is to build a secure login and authentication page for the users.

   Role-Based Access Control:

  - Create the login system such that it supports different user roles (e.g., administrators, waitstaff, kitchen staff, customers).

- Next task is to create the important functions sequentially, along with its database in

MySQL.

   Each function has its own dedicated database.

- In our next step, we will prioritize the development of the payment and billing feature due to its critical importance and the necessity for exceptional accuracy in managing associated risks.

   Prevent Transaction Errors

   Reduce System Downtime

Once we have completed all these steps, we will start final part of the project that is Closing.

○ **Final Report Submission:**

- Prepare a comprehensive final report detailing the problem statement, objectives, methodology, results, and conclusions.

○ **Project Presentation:**

o Create a clear and concise presentation that highlights the project's key aspects.

- **Project Evaluation and Feedback:**

  o Be prepared to answer questions from faculty or reviewers regarding our methodology, decisions, and outcomes.

Tools Used for this Project

Identify the tools you want to use throughout the lifecycle, such as planning tools, design tools, version control, development tools, bug tracking, and testing tools.

Deliverables classified as reuse/build components **(PES2UG22CS042)**

1. User Interface (UI) Design ○

Deliverables:

o Web-based UI for order placement and reservations o

Admin dashboard for managers and staff.

- **Category**: **Reuse Component**
- **Justification**: The UI can be developed using existing web development frameworks and libraries such as React, Angular, or Bootstrap. These frameworks provide pre-built components like forms, tables, and navigation menus, reducing development time and effort.

2. Backend API Development

- **Deliverables**:

  o RESTful APIs for managing orders, reservations, menus, inventory, and employee management.

- **Category**: **Build Component**
- **Justification**: The APIs need to be custom-built to integrate all the required functionalities and ensure seamless interaction between the UI, database, and third-party services. Custom APIs are essential to handle specific business logic, such as order processing, payment integration, and inventory updates.

3. Database Design and

Management ○ **Deliverables**:

o Database schema for all

entities (e.g., customers, orders,

tables, inventory).

  - Scripts for database initialization and migration.

- **Category**: **Build Component**
- **Justification**: The database schema must be designed to meet the specific requirements of the RMS. Custom tables, relationships, and queries will be created to ensure efficient data storage and retrieval. Although a relational database like MySQL is reused, the schema and logic will be custombuilt.

4. Table Management Module

- **Deliverables**:
  - Real-time table availability management.
  - Assignment of customers to tables.
- **Category**: **Reuse Component**
- **Justification**: This module could leverage existing libraries or plugins for real-time data handling and visualization (e.g., WebSockets or Firebase for real-time updates). Basic CRUD operations can use reusable components from the backend framework.

5. Order Management Module ○

Deliverables:

  - Digital order placement and modification.
  - Real-time kitchen notifications.

- **Category**: **Build Component**
- **Justification**: While some components for order placement can be reused, this module will require custom development to handle specific business processes, such as kitchen notification systems and order tracking.

6. Reservation Management

Module ○ **Deliverables**:

  - Online table booking and cancellation.

- **Category**: **Reuse Component**
- **Justification**: Common functionalities like booking and cancellation can be implemented using reusable components from existing libraries or plugins (e.g., calendar or booking libraries). However, some customization may be needed to align with specific restaurant policies.

7. Menu Management Module ○

Deliverables:

  - Digital menu creation and updates.

- **Category**: **Reuse Component**
- **Justification**: Basic CRUD operations for managing menus can leverage existing UI and backend libraries, reducing development time. However, any specific logic for handling real-time updates will need custom work.

8. ***Inventory Management Module***

- **Deliverables**:
  - Inventory tracking and low-stock alerts.
- **Category**: **Build Component**
- **Justification**: Inventory management involves specific business rules and logic (e.g., reducing stock levels based on orders) that need to be built from scratch. Although some parts (like notification services) can be reused, the core functionality needs custom development.

9. ***Billing and Payment Processing***

Module ○ **Deliverables**:

  - Payment gateway integration.
  - Generation of itemized bills.
- **Category**: **Reuse Component**
- **Justification**: Payment processing can be handled using existing payment gateway APIs (e.g., Stripe, PayPal, UPI). Reusing these APIs ensures secure and efficient transactions, while the billing system can use existing libraries for generating itemized bills.

10. ***Employee Management Module***

- **Deliverables**:
  - Staff scheduling, attendance tracking, and payroll management.
- **Category**: **Build Component**
- **Justification**: Employee management requires custom logic, such as dynamic scheduling and payroll calculation based on hours worked and tips. While some UI components can be reused, the backend logic must be custom-built.

11. ***Security and Authentication*** ○

Deliverables:

  - Secure login and role-based access control.
- **Category**: **Reuse Component**

- **Justification**: Reusable components and libraries (e.g., JWT, OAuth) for authentication and authorization can be implemented to ensure secure access management. These are standard practices and do not require custom development.

12. Testing and Quality Assurance ○

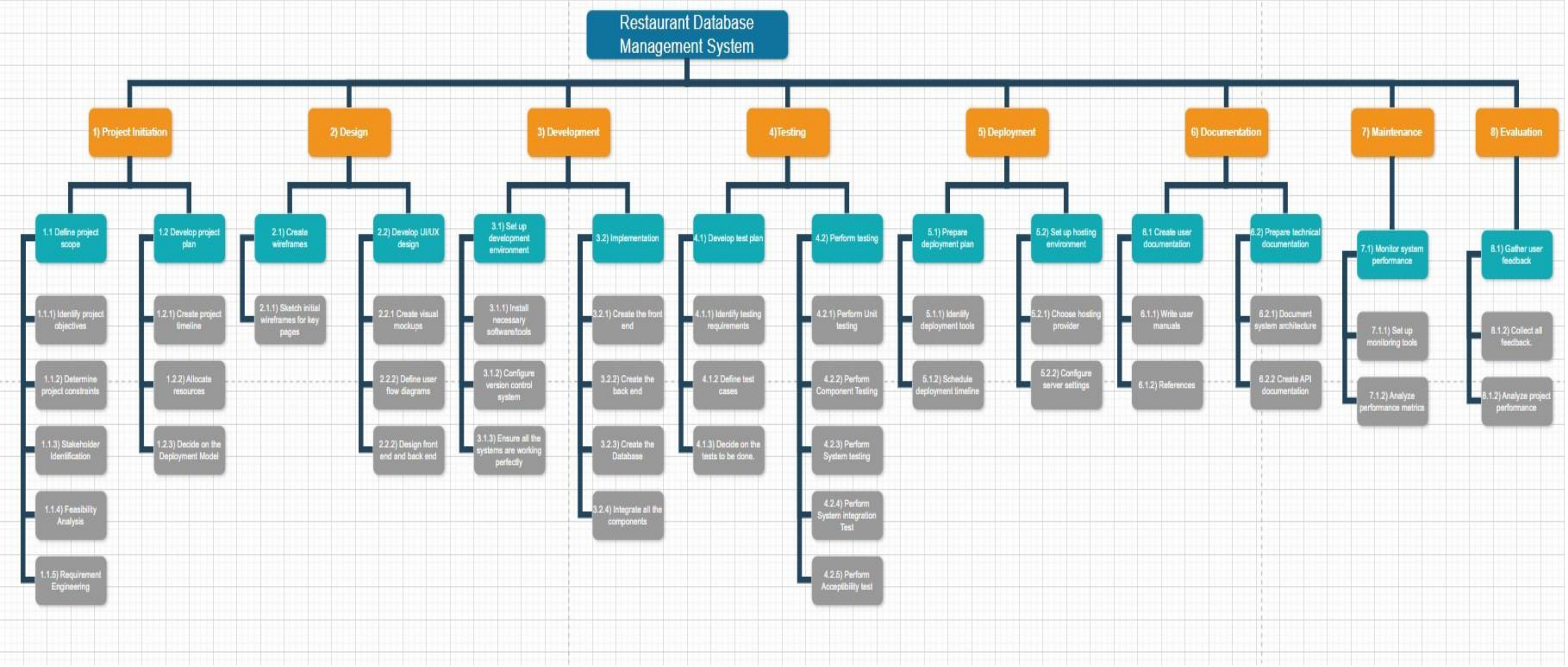
Deliverables: ○ Test cases for all

modules.

○ Performance, security, and usability testing.

- **Category**: **Build Component**
- **Justification**: While some test automation tools (e.g., Selenium, JUnit) can be reused, the test cases need to be built from scratch to cover the specific functional and non-functional requirements of the RMS.

Work Breakdown Structure


Restaurant Database Management System
1) Project Initiation
1.1 Define project scope
1.1.1) Identify project objectives
1.1.2) Determine project constraints
1.1.3) Stakeholder Identification
1.1.4) Feasibility Analysis
1.1.5) Requirement Engineering
1.2 Develop project plan
1.2.1) Create project timeline
1.2.2) Allocate resources
1.2.3) Decide on the Deployment Model
2) Design
2.1) Create wireframes
2.1.1) Sketch initial wireframes for key pages
2.2) Develop UI/UX design
2.2.1 Create visual mockups
2.2.2) Define user flow diagrams
2.2.2) Design front end and back end
3) Development
3.1) Set up development environment
3.1.1) Install necessary software/tools
3.1.2) Configure version control system
3.1.3) Ensure all the systems are working perfectly
3.2) Implementation
3.2.1) Create the front end
3.2.2) Create the back end
3.2.3) Create the Database
3.2.4) Integrate all the components
4)Testing
4.1) Develop test plan
4.1.1) Identify testing requirements
4.1.2 Define test cases
4.1.3) Decide on the tests to be done.
4.2) Perform testing
4.2.1) Perform Unit testing
4.2.2) Perform Component Testing
4.2.3) Perform System testing
4.2.4) Perform System integration Test
4.2.5) Perform Acceptibility test
5) Deployment
5.1) Prepare deployment plan
5.1.1) Identify deployment tools
5.1.2) Schedule deployment timeline
5.2) Set up hosting environment
5.2.1) Choose hosting provider
5.2.2) Configure server settings
6) Documentation
6.1 Create user documentation
6.1.1) Write user manuals
6.1.2) References
6.2) Prepare technical documentation
6.2.1) Document system architecture
6.2.2 Create API documentation
7) Maintenance
7.1) Monitor system performance
7.1.1) Set up monitoring tools
7.1.2) Analyze performance metrics
8) Evaluation
8.1) Gather user feedback
8.1.2) Collect all feedback.
8.1.2) Analyze project performance

Effort Estimation (in person - months)

Provide a rough effort estimate for each task in person-months

working days for one person would be 7/21.66 = 0.323 person-months.

Given Information:

- **Project Type:** Semi-Detached

- **Estimated KLOC:** Let's assume **10 KLOC** for calculations (you can adjust this based on your specific requirements).
- **Number of People:** 2
- **Working Days Available:** Approximately 45 days until the submission date.

COCOMO Model Constants:

- a = 3.0
- b = 1.12

Effort Calculation:

E = a × (KLOC)^b

E = 3.0 × (10)^1.12

E = 39.547 person-days

Duration Calculation:

T = 2.5 × (E)^0.38

T = 2.5 × (39.547) ^0.38

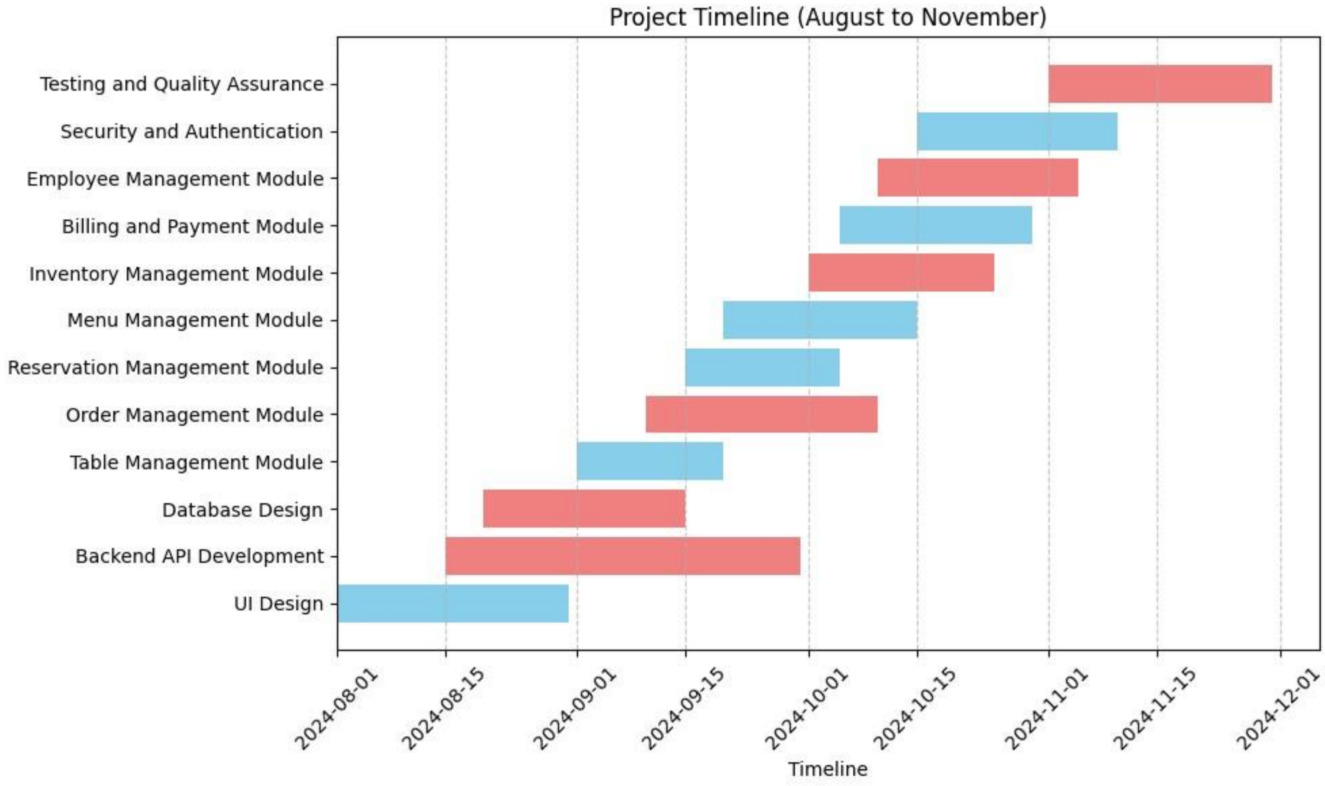
T = 10.11 days

Summary:

- Total Effort Estimate: 39.547 person-days

- Total Duration: 10.11 days

Gantt Chart


Project Timeline (August to November)
Testing and Quality Assurance
Security and Authentication
Employee Management Module
Billing and Payment Module
Inventory Management Module
Menu Management Module
Reservation Management Module
Order Management Module
Table Management Module
Database Design
Backend API Development
UI Design
2024-08-01
2024-08-15
2024-09-01
2024-09-15
2024-10-01
2024-10-15
2024-11-01
2024-11-15
2024-12-01
Timeline