**INTERNSHIP REPORT**

*A report submitted in partial fulfillment of the requirements for the Award of Degree of*

### BACHELOR OF TECHNOLOGY

**in**

**COMPUTER SCIENCE AND ENGINEERING**

**by**

**Name**

**Reg. No.:**

**Under Supervision of Mr. Srinivas, HR**

**NanoMindz Technologies pvt.Ltd, Vishakapatanam.**

**(Duration: )**

A blue and white sign with a gold circle and a blue and white text

AI-generated content may be incorrect.

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**RAMAPURAM, CHENNAI -600089**

**MAY 2025**

**BONAFIDE CERTIFICATE**

This is to certify that the “**Internship report”** submitted by **K.SIREESHA(Regd. No.: 14A51A0565)** is work done by her and submitted during 2023 – 2024 academic year, in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING,** in **SRM Institute of Science and Technology, Ramapuram** at **NanoMindz Technologies pvt.Ltd, Vishakapatanam.**

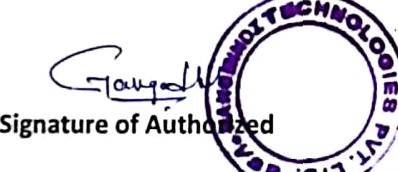
|  |  |
| --- | --- |
| **INTERNSHIP COORDINATOR** | **HEAD OF THE DEPARTMENT** |
| Dr.K.SUJATHA | HOD NAME |
| Assistant Professor, | Designation, |
| Department of Computer Science and  Engineering, | Department of Computer Science and Engineering, |
| SRM IST Ramapuram.  Chennai. | SRM IST Ramapuram.  Chennai. |

NAN0flNDZTf(MN0t0GlfSéW.tlD.

( A n I S 0 8 0 0 I : 2 0 0 8 C e r I I f I e d C o m p a n y )

CERTIFICATE OF INTERNSHIP

This is to certify that Ms. **K.SIREESHA**, student of ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, Tekkali, having ID No. 14A51A0565 has successfully completed the internship programme from 08" May 2017 to 07" June 2017 in our organization Nanomlndz Technologies Pvt. Ltd. at Visakhapatnam.



# TF3, 3rd Floor, 1st Lane, Dwarakanagar, Visakhapatnam -16. Ph: 0891-6633070 [E-mall:hr@nanomIndz.com,](mailto:hr@nanomIndz.com) Web: [www.nanomlndz.com](http://www.nanomlndz.com/)

# 

# ACKNOWLEDGEMENT

My sincere appreciation goes out to Incedo Inc. for giving me the chance to work on the Korporate Kitchen project during my internship. For their unwavering leadership, encouragement, and support during this time, I genuinely thank my mentors, coworkers, and the entire Incedo team. My gratitude also goes out to my college professors, whose wise counsel and unwavering encouragement have enabled me to close the gap between academic theory and practical implementation. It has been a rewarding experience that has helped me improve both professionally and personally.

# ABSTRACT

# Korporate Kitchen is an innovative online food delivery system developed during my internship at Incedo Inc., a leading technology firm renowned for its expertise in digital transformation. This project represents a significant step forward in the evolution of the food delivery industry, aiming to streamline and enhance the experience for all stakeholders involved. The primary goal of Korporate Kitchen is to provide a seamless, end-to-end solution that integrates all components of the food delivery ecosystem. This includes connecting users, restaurants, delivery agents, and administrators into one cohesive platform. By leveraging cutting-edge technology, the system ensures a smooth and efficient process from the moment a customer places an order to the final delivery at their doorstep. Key features of the platform include a User-Friendly interface where customers can effortlessly browse menus, place orders, and track deliveries in real time. Next is Restaurant Integration where the system allows restaurants to manage orders, update menus, and optimize delivery schedules with ease. The process after Restaurant integration is Efficient Delivery Management where Delivery agents can quickly receive, process, and complete orders with enhanced route optimization. Then, the Admin Dashboard where Administrators have full control over the platform, including managing users, restaurants, and deliveries, ensuring a seamless operation. Korporate Kitchen not only aims to simplify the food ordering process but also to provide a reliable, fast, and personalized service for both customers and service providers. Through this project, I had the opportunity to apply my technical skills in a real-world setting, contributing to the digital transformation of the food service industry.

### Organization Information

### Incedo’s emphasis on innovation and rapid technological advancement were likely involved in projects where we had to analyse complex problems, devise scalable solutions, and ensure that these solutions aligned with industry best practices. Encountering real-world challenges helped to think critically and adapt to fast-paced environments and to develop a strong ability to troubleshoot and optimize systems efficiently. Given Incedo’s focus on staying at the forefront of technology, I have worked with AI/ML models, big data analytics, cloud computing, IoT, or blockchain technologies. Incedo’s emphasis on fostering a culture of innovation and valuing creative solutions helped to think outside the box and experiment with new ideas.

### Programs and Opportunities

During my tenure at Incedo Inc., the Korporate Kitchen project provided you with an exceptional opportunity to not only enhance my technical expertise but also develop a strong foundation in agile methodologies and collaborative problem-solving. This experience gave us a well-rounded exposure to full-stack development, integrating back-end and front-end technologies seamlessly, while reinforcing essential software engineering practices. Being assigned to a full-stack application like Korporate Kitchen meant that you had to work on multiple layers of the application, including Backend Development with Spring Boot, Leveraging Spring Boot, we built robust and scalable RESTful APIs, ensuring efficient communication between the client and the server. We worked with features such as dependency injection, aspect-oriented programming (AOP), and Spring Security to enhance the security and modularity of the application. Developing APIs using Spring Boot taught us how to manage complex business logic, implement exception handling, and optimize response times. Database Management with Hibernate ORM, we effectively mapped Java objects to database tables, allowing seamless interaction between the application and the relational database. Developing dynamic web interfaces with JSP allowed me to create interactive and user-friendly interfaces that communicated with the backend seamlessly.Our core values focus on delivering exceptional service and building lasting relationships with our clients. By maintaining a commitment to delivering projects on time and within budget, we ensure that every solution we offer is not just a service, but a partnership aimed at long-term success. We take pride in our ability to collaborate, communicate transparently, and offer ongoing support, ensuring that our clients always feel confident and empowered in their digital transformation journey.

### Methodologies

The project was developed using the Model-View-Controller (MVC) architecture, ensuring a clear separation of concerns between business logic, the presentation layer, and the data access layer, enhancing maintainability and scalability. Spring Boot streamlined application development by eliminating boilerplate code and enabling seamless deployment with its embedded server. Hibernate and Spring Data JPA simplified database management through efficient object-relational mapping (ORM) and reduced SQL complexity. JSP and JSTL facilitated dynamic content rendering in the front end, while Bootstrap ensured a responsive, mobile-friendly design. Agile methodologies, including Scrum, guided the development process through iterative sprints, regular reviews, and continuous feedback. User stories and backlog management ensured that the most valuable features aligned with business objectives. Continuous Integration and Deployment (CI/CD) pipelines automated build, testing, and deployment processes, ensuring high code quality. Git was used for version control, following a branching strategy to manage development and releases effectively. Automated testing, along with deployment tools like Jenkins or GitHub Actions, minimized downtime and ensured smooth production releases.

### Key Benefits Achieved

* Scalability and Maintainability: Clean separation of concerns made it easier to extend and modify individual application components.
* Reduced Development Time: Spring Boot’s auto-configuration and dependency management significantly reduced manual effort.
* Improved Collaboration: Agile practices ensured that business needs were met through continuous feedback and iterative releases.
* Enhanced User Experience: Bootstrap’s responsive design ensured that the application adapted well to various devices and screen sizes.

### Key Parts Of The Report

The project report is structured to provide a comprehensive and detailed analysis of the development process, highlighting the project's purpose, technical architecture, and overall impact. It begins with an introduction that outlines the primary objective of the project, emphasizing its goal to create a seamless and efficient platform that caters to different stakeholders, including users, restaurants, delivery personnel, and administrators. The requirement analysis section delves into the meticulous process of gathering both functional and non-functional requirements. This phase involved extensive interaction with stakeholders through interviews and surveys to define user stories and use cases that reflected real-world needs. During this phase, critical design considerations, including scalability, security, and performance optimization, were also taken into account to ensure that the system was capable of handling increasing user loads and adapting to future business requirements.

**Benefits Of The Company**

Incedo specializes in delivering industry-specific solutions across multiple domains using cutting-edge technologies. The company focuses on building strong client relationships by ensuring timely delivery of projects within budget. With expertise in emerging technologies like AI, ML, and Cloud Computing, Incedo provides scalable and future-ready solutions. Its agile approach ensures flexibility, allowing clients to adapt to changing business needs efficiently.

### Bottom of Form

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **ABSTRACT** | | v |
| **LIST OF FIGURES** | | xi |
| **LIST OF TABLES** | | xii |
| **LIST OF ACRONYMS AND ABBREVIATIONS** | | xiii |
| **SNO** | **CONTENTS** | **PAGE NO.** |
| **1** | **INTRODUCTION** | 17 |
|  | **1.1 MODULE DESCRIPTION** | 18 |
|  | **1.1.1 USER** | 18 |
|  | **1.1.2 RESTAURANT** | 19 |
|  | **1.1.3 DELIVERY** | 19 |
|  | **1.1.4 ADMIN** | 20 |
| **2** | **SYSTEM ANALYSIS** | 21 |
|  | **2.1 REQUIREMENT ANALYSIS** | 21 |
|  | **2.2 USE CASE DEVELOPMENT** | 21 |
|  | **2.2.1 DATAFLOW ANALYSIS** | 21 |
|  | **2.2.2 FEASIBILITY STUDY** | 22 |
|  | **2.2.3 PROPOSED SYSTEM** | 22 |
| **3** | **SOFTWARE REQUIREMENTS SPECIFICATION** | 23 |
|  | **3.1 FUNCTIONAL REQUIREMENT** | 23 |

|  |  |  |
| --- | --- | --- |
|  | **3.2 SOFTWARE REQUIREMENTS** | 23 |
|  | **3.3 HARDWARE REQUIREMENTS** | 23 |
| **4** | **TECHNOLOGY** | 24 |
|  | **4.1 BACKEND TECHNOLOGIES** | 24 |
|  | **4.1.1 SPRING BOOT** | 24 |
|  | **4.1.2 HIBERNATE (JPA)** | 24 |
|  | **4.1.3 SPRING DATA JPA** | 25 |
|  | **4.2 FRONTEND TECHNOLOGIES** | 25 |
|  | **4.2.1 JSP and JSTL** | 25 |
|  | **4.2.2 BOOTSTRAP** | 26 |
|  | **4.2.3 FONT AWESOME** | 26 |
|  | **4.3 DATABASE AND INFRASTRUCTURE** | 27 |
|  | **4.3.1 MYSQL** | 27 |
|  | **4.3.2 MYSQL WORKBENCH** | 27 |
|  | **4.3.3 MAVEN** | 28 |
| **4** | **CODING** | 29 |
| **5** | **SCREENSHOTS** | 38 |
| **6** | **CONCLUSION** | 42 |
| **7** | **BIBLIOGRAPHY** | 43 |

|  |  |  |
| --- | --- | --- |
| **FIG NO.** | **FIGURE NAME** | **PAGE NO.** |
| 5.1 | User Portal | 38 |
| 5.2 | User Login Page | 38 |
| 5.3 | User Profile | 38 |
| 5.4 | User Home Page | 38 |
| 5.5 | Orders | 39 |
| 5.6 | Admin Dashboard | 39 |
| 5.7 | Admin Portal | 39 |
| 5.8 | User Management | 39 |
| 5.9 | Restaurant Management | 39 |
| 5.10 | Delivery Agent Login/Sign in portal | 40 |
| 5.11 | Delivery Agent Dashboard | 40 |
| 5.12 | Delivery Agent Registration Page | 40 |
| 5.13 | Restaurant Portal | 41 |
| 5.14 | Home Page | 41 |
| 5.15 | Menu Management Page | 41 |
| 5.16 | Active Orders List | 41 |

**LIST OF FIGURES**

**LIST OF TABLES**

**WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES**

|  |  |
| --- | --- |
| **WEEK NO.** | **PAGE NO.** |
| WEEK-1 | xiii |
| WEEK-2 | xiii |
| WEEK-3 | xiv |
| WEEK-4 | xiv |

**LIST OF ACRONYMS AND ABBREVIATIONS**

|  |  |
| --- | --- |
| JSP | Java Server Pages |
| JSTL | Java Server Pages Standard Tag Library |
| JDK | Java Development Kit |
| IDE | Integrated Development Environment |
| MVC | Model-View-Controller |
| REST | Representational State Transfer |

Learning Objectives/Internship Objectives

* Internships are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships in order to receive real world experience and develop their skills.
* An objective for this position should emphasize the skills you already possess in the area and your interest in learning more
* Internships are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.
* Some internship is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.
* Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

### WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES

|  |  |  |  |
| --- | --- | --- | --- |
| 1st WEEK | DATE | DAY | NAME OF THE TOPIC/MODULE COMPLETED |
| 08/01/25 | Wednesday | Introduction to Korporate Kitchen |
| 09/01/25 | Thursday | Understanding Business Requirements |
| 10/01/25 | Friday | Setting up Development Environment |

|  |  |  |  |
| --- | --- | --- | --- |
| 2nd WEEK | DATE | DAY | NAME OF THE TOPIC/MODULE COMPLETED |
| 13/01/25 | Monday | Database Schema and Planning |
| 14/01/25 | Tuesday | Setting up Spring Boot & Data Base Connection |
| 15/01/25 | Wednesday | Implementing User Authentication & Authorization |
| 16/01/25 | Thursday | Designing User Management Module |
| 17/01/25 | Friday | Continuing windows services |

|  |  |  |  |
| --- | --- | --- | --- |
| 3rd WEEK | DATE | DAY | NAME OF THE TOPIC/MODULE COMPLETED |
| 20/01/25 | Monday | Integrating User Profiles & Settings |
| 21/01/25 | Tuesday | Implementing API Security & JWT Authentication |
| 22/01/25 | Wednesday | Developing order Management Module |
| 23/01/25 | Thursday | Implementing Order Placement |
| 24/01/25 | Friday | Integrating Payment Gateway |

|  |  |  |  |
| --- | --- | --- | --- |
| 4th WEEK | DATE | DAY | NAME OF THE TOPIC/MODULE COMPLETED |
| 27/01/25 | Monday | Handling Order Status and Notifications |
| 28/01/25 | Tuesday | Implementing Order History & Tracking |
| 29/01/25 | Wednesday | Conducting System – Wide Testing |
| 30/01/25 | Thursday | Implementing Performance Enhancements |
| 31/01/25 | Friday | User Acceptance Testing |

# INTRODUCTION

Korporate Kitchen is an online food delivery system designed to streamline the process of ordering food online efficiently and seamlessly. Developed during my internship at Incedo Inc., the project encompasses the complete workflow, starting from user registration and authentication to restaurant menu management, order placement, secure payment processing, and real-time delivery tracking. The system is built using Java technologies and follows the Model-View-Controller (MVC) architecture, ensuring a clear separation of concerns, making the application highly scalable, maintainable, and easy to extend. Advanced features such as dynamic menu updates, personalized recommendations, and order history tracking were integrated to enhance the overall user experience, making the platform both intuitive and reliable.

Korporate Kitchen not only simplifies the online food ordering process but also enhances user satisfaction by providing a seamless and intuitive interface. The system incorporates robust security measures to protect sensitive user data and ensure secure payment transactions. Role-based access control was implemented to differentiate permissions for customers, administrators, and delivery personnel, ensuring smooth management of user roles and responsibilities. The application uses JSP and JSTL for the front-end, delivering a dynamic and responsive user interface, while Hibernate and Spring Data JPA handle database interactions efficiently, ensuring data integrity and consistency. RESTful APIs were developed to facilitate seamless communication between the front-end and back-end systems, promoting modularity and scalability. Additionally, the system supports real-time order status updates and delivery tracking, providing users with transparency and control over their orders. To ensure high performance and reliability, extensive unit and integration testing were conducted throughout the development process. The adoption of Agile methodologies enabled continuous feedback and iterative improvements, resulting in a high-quality application that met business requirements effectively. Through this project, I gained valuable hands-on experience in Java technologies and full-stack application development, enhancing my technical expertise and problem-solving abilities.

Top of Form

Bottom of Form

1. Top of Form
2. Bottom of Form

### 1.1 Module Description:

The Korporate Kitchen project is designed as a comprehensive online food delivery system that brings together multiple stakeholders through distinct yet integrated modules. Each module has been developed to address specific functionalities and workflows within the system. Below is an in-depth description of each module:

* + 1. User   
       The User Module is the primary interface for the end customers. It encompasses functionalities that allow users to:
       1. Register and Log In: New users can create an account by providing their personal details such as name, email, password, contact information, and address. Existing users log in to access the system.
       2. Profile Management: Users can view and update their profile details including their contact information and delivery address.
       3. Browse Restaurants: Once logged in, users are presented with a dashboard that lists all the available restaurants. Each restaurant is displayed with key details like name, address, and a link to view its menu.
       4. Restaurant Menu and Cart Management: Users can click on a restaurant to view its menu. They have the option to add selected menu items to their cart. The cart module keeps track of items, quantities, and calculates the total price.
       5. Payment and Order Placement: The module integrates a payment gateway that allows users to securely make payments for their orders. After payment confirmation, the order is placed and stored in the system.
       6. Order History and Repeat Orders: Users can view a detailed history of their past orders and have the option to repeat previous orders with a single click.
       7. Remarks Feature: To improve service quality, users can provide feedback or remarks about their experience. They can view all submitted remarks and add new ones directly through the interface.
    2. Restaurant   
       The Restaurant Module caters to the needs of restaurants participating in the platform. Its functionalities include:
       1. Registration and Login: Restaurants can register on the platform by providing necessary details such as restaurant name, contact information, and address. After registration, they can log in to access their dashboard.
       2. Dashboard: The restaurant dashboard offers an overview of incoming orders, sales, and other key performance indicators. It also provides links to manage the menu and view orders.
       3. Menu Management: Restaurants can add, update, or delete menu items through a dedicated interface. Each menu item includes details such as name, description, price, type, and availability timings (available from and available until).
       4. Order Tracking: The module enables restaurants to view and manage orders placed by users. This includes order details such as items ordered, quantities, total price, and order status.
       5. Bank Details Management: To facilitate payment reconciliation, restaurants can view their bank details and transaction history related to customer orders.
    3. Delivery   
       The Delivery Module is designed for the delivery agents who are responsible for order pickup and delivery. This module includes:
       1. Registration and Login: Delivery agents register on the platform by providing personal and contact details. After registration, they can log in to access their delivery dashboard.
       2. Dashboard: The delivery dashboard displays orders that are awaiting delivery. Delivery agents can view order details, including the customer’s name, items ordered, quantity, total price, and order date.
       3. Order Management: Delivery agents can accept or reject orders. Once an order is accepted, the status is updated accordingly and the user is informed about the delivery progress.
       4. Profile Management: Delivery agents have access to a profile page where they can view and update their personal information.
       5. Delivery Status Updates: The module allows agents to update the delivery status (e.g., pending, delivered, or failed) as they progress through the delivery process.
    4. Admin   
        The Admin Module provides administrative oversight of the entire system. Its key functionalities include:
       1. Dashboard: Administrators have access to a central dashboard that aggregates critical system information, including counts of registered users, restaurants, and delivery agents.
       2. User Management: Admins can view, edit, or delete user accounts, ensuring that the platform remains secure and well-managed.
       3. Restaurant Management: This section allows administrators to monitor restaurant registrations, menu updates, and overall restaurant performance on the platform.
       4. Delivery Agent Management: Admins can oversee the registration and activity of delivery agents, ensuring that deliveries are executed efficiently.
       5. System Monitoring: Additional tools and views are provided to monitor transactions, order flow, and system performance, facilitating quick resolution of any issues that may arise.

# 2. SYSTEM ANALYSIS

### 2.1 Requirement Analysis

### Meetings with stakeholders and end-users were conducted to gather valuable insights and identify the core functionalities required for each module. These discussions provided a clear understanding of business needs, user expectations, and system requirements, enabling the development team to define precise objectives and prioritize essential features. The collaborative approach ensured that the system was aligned with the stakeholders' goals while addressing the end-users' pain points effectively.

2.2 Use Case Development:

Use case development involved creating detailed use case diagrams and flowcharts to map the entire process flow, covering interactions between users and the system as well as backend processing. These diagrams provided a visual representation of various scenarios, outlining how different user roles interact with the system and how the system responds to their actions. By defining each use case, the development team was able to identify potential system behaviors, validate functional requirements, and ensure that all critical workflows were accounted for in the system design.

2.2.1 Data Flow Analysis:

Data Flow Analysis involved creating detailed data flow diagrams (DFDs) to illustrate how data moves between different modules and the database. These diagrams provided a clear visualization of the system’s data inputs, processes, and outputs, ensuring seamless communication between various components. By mapping the data flow, the development team was able to identify potential bottlenecks, optimize data processing, and maintain data integrity across the system. This analysis ensured that the system architecture was efficient, scalable, and capable of handling future data growth effectively.

2.2.2 Feasibility Study:

This involved a thorough evaluation of the technical, economic, and operational aspects of the project to determine its viability. Technical feasibility assessed whether the available technologies, tools, and infrastructure could support the development and deployment of the system. Economic feasibility analysed the cost-effectiveness of the project by comparing the estimated development, maintenance, and operational costs against the expected benefits. Operational feasibility evaluated whether the system would integrate smoothly into the existing workflows and meet the needs of end-users. This comprehensive assessment helped identify potential risks, establish mitigation strategies, and ensure that the project was both practical and sustainable. The analysis of Non-Functional Requirements focused on evaluating critical aspects such as performance, scalability, security, and maintainability to ensure that the system could meet both current and future demands. Performance considerations involved optimizing response times, minimizing latency, and ensuring the system could handle high loads efficiently. Scalability was assessed to ensure the system could accommodate increasing user traffic and data volume without compromising functionality. Security measures were analysed to protect sensitive data, prevent unauthorized access, and ensure compliance with industry standards. Maintainability was prioritized to facilitate easy updates, bug fixes, and system enhancements over time. This thorough evaluation ensured that the system was not only reliable and secure but also capable of adapting to evolving business needs.

1. Top of Form
2. Bottom of Form

### 2.2.3 Proposed System

The Korporate Kitchen project optimizes workflow by restructuring operations for efficiency. It applies key transformations such as Swap, which reorders adjacent processes to enhance execution, and Distribute/Factorize, which either distributes or consolidates similar operations. Additionally, Merge/Split is used to combine multiple tasks or divide complex ones for better modularity. The Re-associate transformation adjusts the execution order of functionally associative operations to improve performance. These transformations ensure that the system remains scalable, efficient, and well-maintained.

# 3. SOFTWARE REQUIREMENTS SPECIFICATIONS

### 3.1 Functional Requirements

The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements.

* 1. Software Requirements:
* Operating system: Windows 11 (64-bit).
* Coding Language : Java 11
* IDE : Spring Tool Suite (STS) 4
* Application Server**:** Apache Tomcat 10.1.39
* Database management tool : MySQL Workbench 8.0.41
* Build Tool**:** Maven 3.9.9
  1. Hardware Requirements:
* Processor**:** Intel Core i5
* Hard Disk**:** Minimum 500 GB
* RAM**:** 16 GB

# TECHNOLOGY

4.1 Backend Technologies

4.1.1 Spring Boot

Spring Boot is a powerful framework used for building enterprise-grade RESTful web services and microservices with minimal configuration. It is built on top of the Spring Framework and simplifies dependency management, auto-configuration, and application setup.

**Features:**

* Embedded servers (Tomcat, Jetty) for easy deployment.
* Built-in support for RESTful APIs and JSON processing.
* Auto-configuration that reduces boilerplate code.
* Supports Microservices Architecture.

Spring Boot applications are often configured using application.properties or application.yml files, making it highly flexible and customizable. It follows the "convention over configuration" principle, allowing developers to get started quickly without extensive configuration.

4.1.2 Hibernate (JPA)

Hibernate is an Object-Relational Mapping (ORM) framework that simplifies database interactions by mapping Java objects to database tables. It implements the Java Persistence API (JPA) and provides a high-level abstraction for CRUD operations.

**Features:**

* Automatic mapping of Java objects to database tables.
* Lazy and Eager loading of data to optimize performance.
* Caching to improve query performance.
* Supports HQL (Hibernate Query Language) for complex queries.

Hibernate eliminates the need for manually writing SQL queries by using annotations and XML configurations. It ensures seamless interaction between Java applications and relational databases.

4.1.3 Spring Data JPA

Spring Data JPA is part of the Spring ecosystem that provides an abstraction layer for working with databases and reduces boilerplate code in data access layers. It uses JPA repositories to perform CRUD operations and offers derived query methods for quick implementation.

**Features:**

* Simplifies database interactions with minimal coding.
* Supports custom query methods and pagination.
* Provides an abstraction layer over Hibernate and JPA.
* Supports both synchronous and asynchronous database operations.

Spring Data JPA works seamlessly with Spring Boot to provide an efficient and robust data access layer in modern applications.

4.2 Frontend Technologies

4.2.1 JSP and JSTL

Java Server Pages (JSP) is a technology used to create dynamic web pages using Java. It allows embedding Java code in HTML pages to dynamically generate content. JSP works with Java Servlets to process requests and generate responses.

**Features:**

* Easy integration with backend Java applications.
* Allows inclusion of reusable components and custom tags.
* Provides implicit objects to manage request and response.

JSTL (Java Server Pages Standard Tag Library) provides a set of commonly used tags to simplify JSP page development and encourage code reuse.

4.2.2 Bootstrap

Bootstrap is a popular front-end framework for developing responsive and mobile-first web applications. It provides pre-designed CSS and JavaScript components to create modern UI designs quickly.

**Features:**

* Responsive grid system for layout management.
* Ready-to-use components such as buttons, modals, and forms.
* Easy integration with custom CSS and JavaScript.
* Consistent design across different screen sizes and devices.
* Ready-to-use components such as buttons, modals, and forms.

Bootstrap enables faster front-end development while maintaining consistency and compatibility across various browsers.

4.2.3 Font Awesome

Font Awesome is a widely used icon library that provides scalable vector icons that can be customized with CSS. It offers a vast collection of icons that can be easily integrated into web applications.

**Features:**

* Provides over 7,000+ icons.
  + - * Customizable through CSS for size, color, and position.
* Lightweight and easy to include in projects.
* Supports SVG, web fonts, and accessibility features.

4.3 Database & Infrastructure

4.3.1 MySQL

MySQL is a reliable and widely-used relational database management system (RDBMS) that serves as the primary database for storing application data. It is known for its high performance, reliability, and scalability.

**Features:**

* Supports ACID transactions to ensure data integrity.
* Provides replication and clustering for high availability.
* Compatible with multiple platforms and programming languages.
* Provides robust security features for protecting data.

MySQL is used extensively in enterprise applications, e-commerce platforms and content management systems.

4.3.2 MySQL Workbench

MySQL Workbench is a visual tool used for designing, managing, and administering MySQL databases. It provides a unified environment for database development, administration, and performance monitoring.

**Features:**

* Visual database design and schema modelling.
* Query building and execution using SQL.
* Backup and restoration of database instances.
* Performance tuning and monitoring.

MySQL Workbench simplifies the management of MySQL databases by providing an intuitive user interface.

4.3.3 Maven

Maven is a powerful build automation tool used primarily for Java projects. It manages project dependencies and provides a standardized way to build and package applications.

**Features:**

* Dependency management using pom.xml.
* Supports building, testing, and deploying applications.
* Easy integration with IDEs like Spring Tool Suite (STS).
* Generates project structures and documentation automatically.
  + Create special effects with images that give the impression that a button is either highlighted or depressed whenever the mouse pointer is hovered over it.
* Validate information that users enter into your web forms
* Open pages in new windows and customize the appearance of those new windows
* Detect the capabilities of the user’s browser and alter your page’s content appropriately.

5. CODING  
The Korporate Kitchen project follows the Model-View-Controller (MVC) architecture, ensuring a clear separation between the business logic, data access, and presentation layers. The key components are:

* **Models:**  
  These classes represent the business entities such as User, Restaurant, Order, Cart, MasterMenuItem, Payment, Bank, Delivery, and Remark. Hibernate annotations are used to map these classes to the corresponding database tables.
* **Controllers:**  
  Controllers handle HTTP requests and coordinate interactions between the frontend (JSP pages) and backend services. For example, the UserController manages user authentication, profile management, order placement, and remarks; the RestaurantController handles restaurant-specific operations; the DeliveryController manages the order delivery process; and the AdminController provides system management capabilities.
* **Services:**  
  The service layer contains business logic and abstracts interactions with the repository layer. This ensures that business rules are consistently applied across the application.
* **Repositories:**  
  Using Spring Data JPA, the repository layer handles all database interactions, allowing developers to focus on business logic rather than SQL queries.

**1. Base Model Code Example (User Entity) :**package com.example.demo.model;

import jakarta.persistence.\*;

@Entity

@Table(name = "user")

public class User {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int userId;

@Column(nullable = false, length = 50)

private String userName;

@Column(nullable = false, unique = true, length = 100)

private String email;

@Column(nullable = false, length = 255)

private String password;

@Column(length = 15)

private String contact;

@Column(columnDefinition = "TEXT")

private String address;

// Constructors, Getters and Setters

public User() { }

public int getUserId() {

return userId;

}

public void setUserId(int userId) {

this.userId = userId;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getContact() {

return contact;

}

public void setContact(String contact) {

this.contact = contact;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

}

**2. Base Repository Code Example (User Repository)** :  
  
package com.example.demo.repository;

import com.example.demo.model.User;

import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<User, Integer> {

// Custom method to find a user by email address

User findByEmail(String email);

}  
3. Base Service Code Example (User Service and Implementation)  
package com.example.demo.service;

import com.example.demo.model.User;

public interface UserService {

User saveUser(User user);

User getUserByEmail(String email);

}  
---------------------------------------------------  
package com.example.demo.service.impl;

import com.example.demo.model.User;

import com.example.demo.repository.UserRepository;

import com.example.demo.service.UserService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class UserServiceImpl implements UserService {

@Autowired

private UserRepository userRepository;

@Override

public User saveUser(User user) {

return userRepository.save(user);

}

@Override

public User getUserByEmail(String email) {

return userRepository.findByEmail(email);

}

}  
  
**4. Base Controller Code Example (Home Controller)** :  
  
package com.example.demo.controller;

import com.example.demo.model.User;

import com.example.demo.service.UserService;

import jakarta.servlet.http.HttpSession;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestParam;

@Controller

public class HomeController {

@Autowired

private UserService userService;

@GetMapping("/")

public String home() {

return "home"; // Resolves to /WEB-INF/jsp/home.jsp

}

@PostMapping("/login")

public String login(@RequestParam String email, @RequestParam String password, HttpSession session, Model model) {

User user = userService.getUserByEmail(email);

if (user != null && user.getPassword().equals(password)) {

session.setAttribute("user", user);

return "dashboard"; // Resolves to /WEB-INF/jsp/dashboard.jsp

} else {

model.addAttribute("error", "Invalid credentials");

return "login"; // Resolves to /WEB-INF/jsp/login.jsp

}

}  
  
The code is organized into packages by function (controllers, models, services, repositories, and JSP views), which enhances modularity and maintainability. The project was developed using agile methodologies, ensuring iterative improvements and rapid integration of feedback.

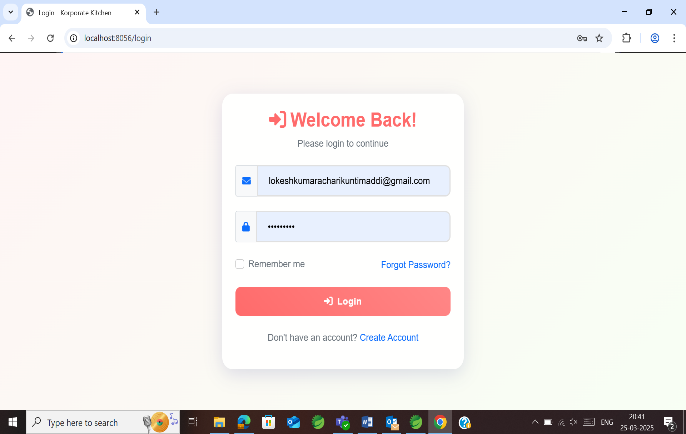
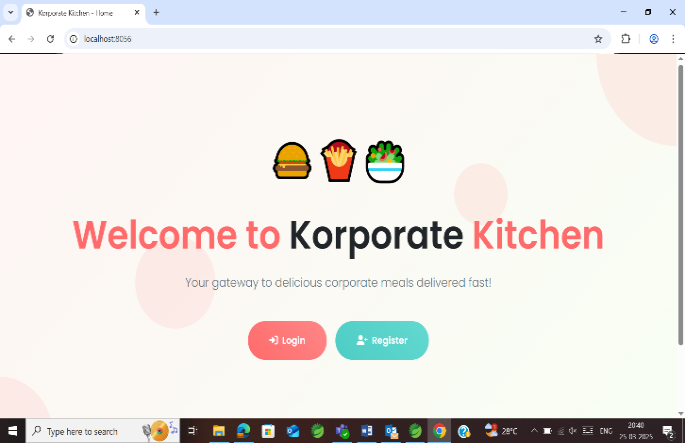
****  
  
5. SCREENSHOTS  
  
User Side:

Fig 5.1 User Portal Fig 5.2 User Login Page

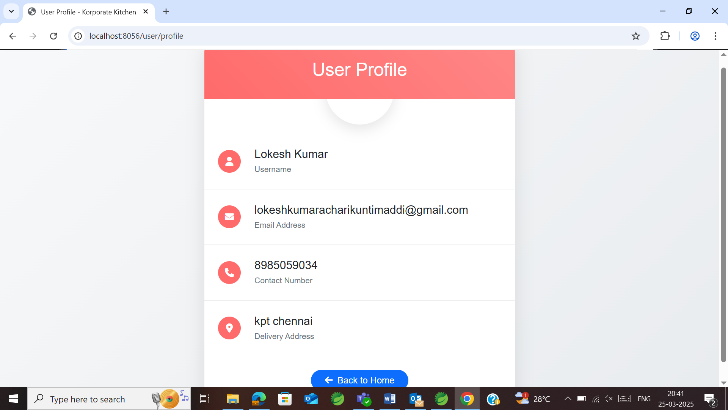
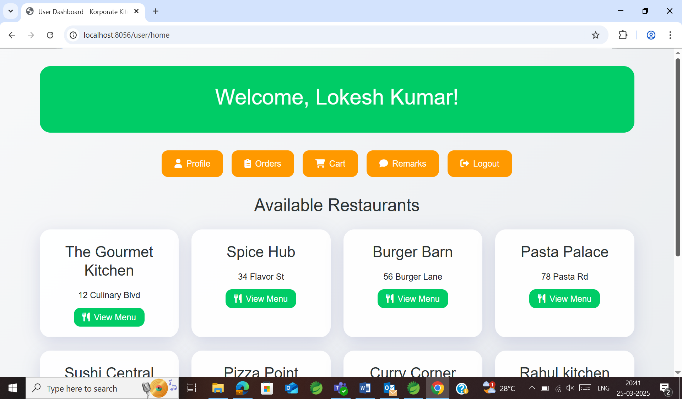


Fig 5.3 User Profile Fig 5.4 User Home Page

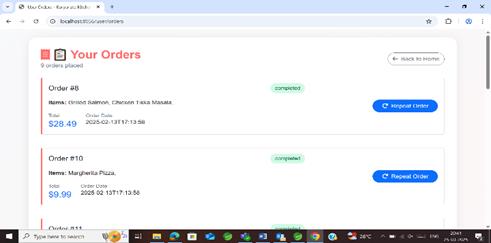
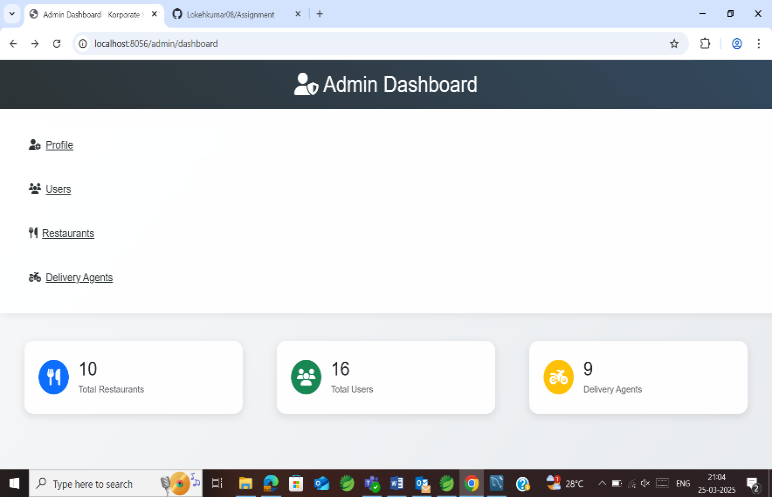


Fig 5.5 Orders

Admin Side:

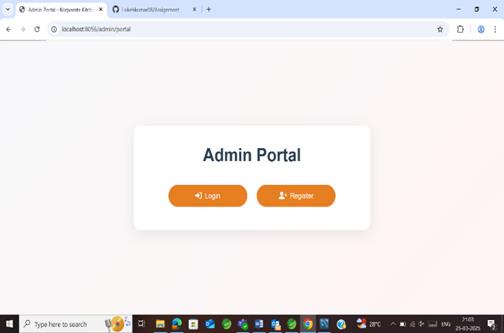


Fig 5.6 Admin Dashboard Fig 5.7 Admin Portal

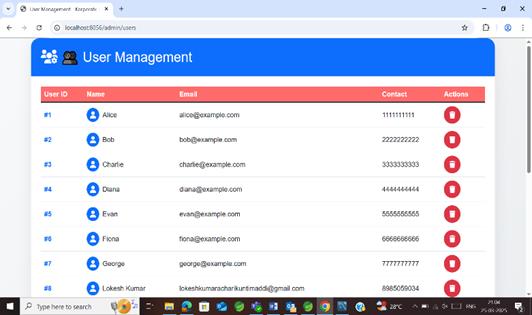
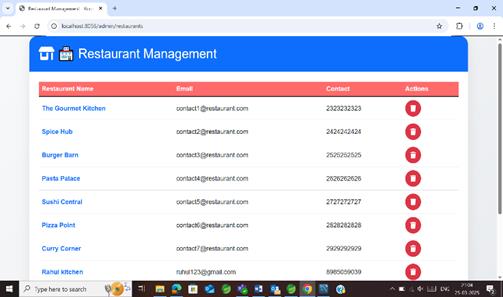
 

Fig 5.8 User Management Fig 5.9 Restaurant Management

Delivery Agent Side :

Delivery Agent Login/Sign in portal Delivery Agent Dashboard

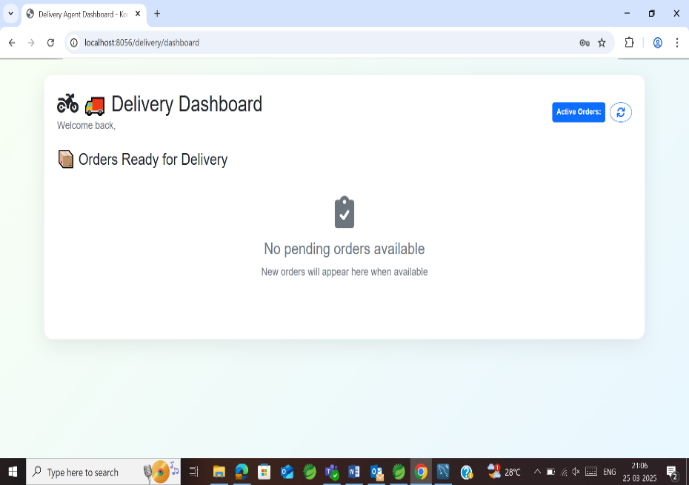
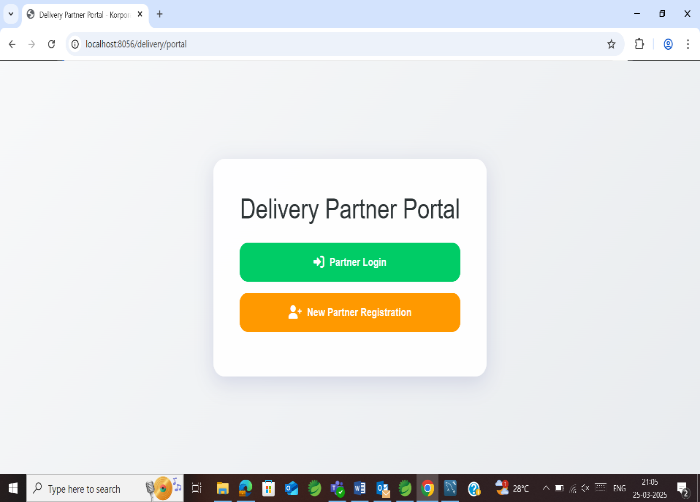
****

Fig 5.10 Delivery Agent Login/Sign in portal Fig 5.11 Delivery Agent Dashboard

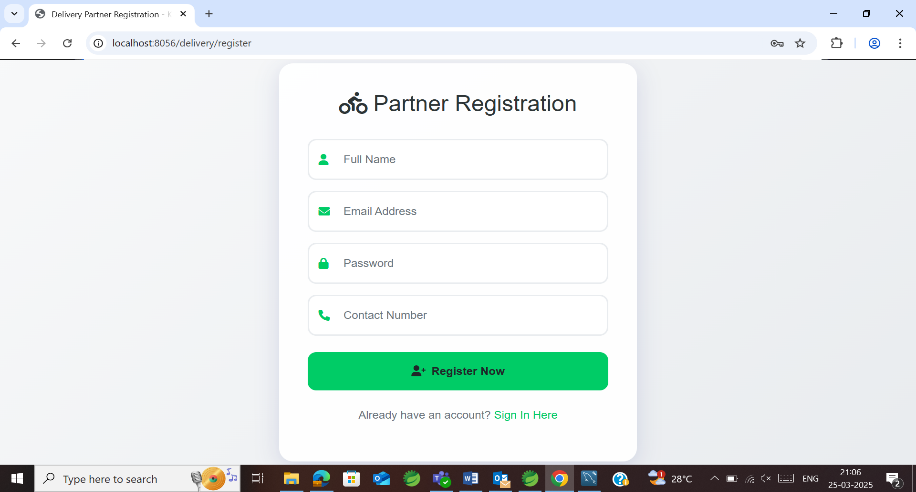
 Delivery Agent Registration page

Fig 5.12 Delivery Agent Registration Page

# Restaurant Side :

# Restaurant Portal Home Page

Fig 5.13 Restaurant Portal Fig 5.14 Home Page

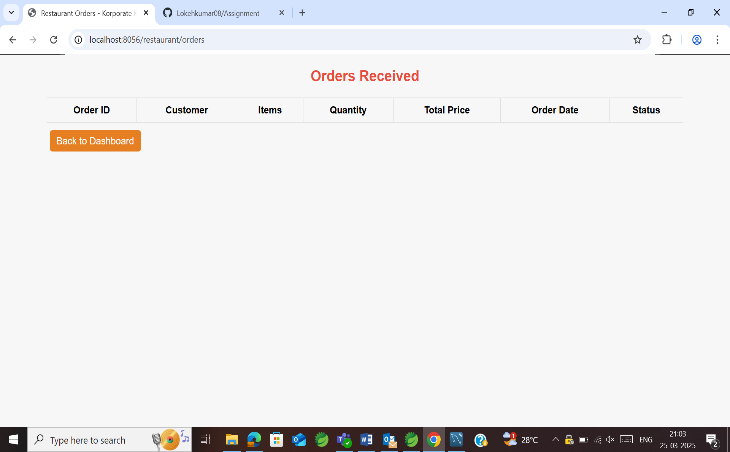
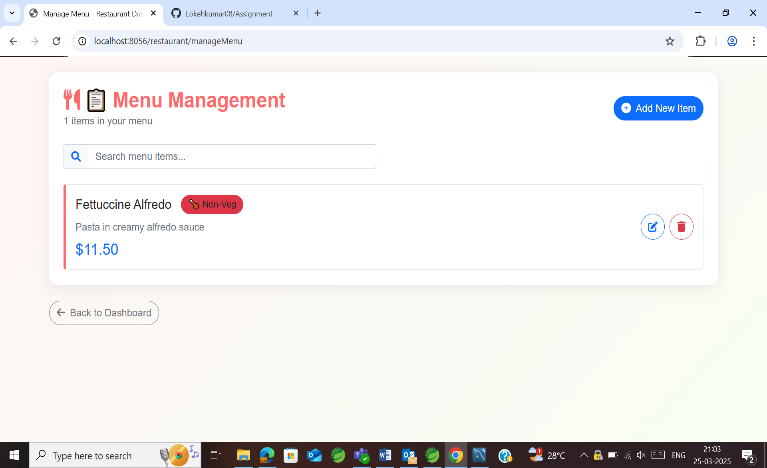
****Menu Management Page Active Orders List

Fig 5.15 Menu Management Page Fig 5.16 Active Orders List

7. CONCLUSION  
  
The Korporate Kitchen project successfully demonstrates the development of a comprehensive online food delivery system using modern web technologies. Through this project, I gained valuable experience in full-stack development, including the use of Spring Boot for backend services, Hibernate for ORM, and JSP with Bootstrap for building dynamic and responsive user interfaces.

The modular design, following the MVC architecture, ensures that the application is scalable, maintainable, and easy to enhance. The project not only meets the functional requirements of an online food delivery system but also incorporates robust security, performance, and scalability measures.

Overall, this internship project has been a significant learning experience, bridging the gap between academic theory and real-world application. The skills and knowledge I acquired during this project will undoubtedly serve as a strong foundation for my future career in software development

8. BIBILOGRAPHY

* Spring Boot Reference Documentation: <https://spring.io/projects/spring-boot>
* Hibernate ORM Documentation: <https://hibernate.org/orm/documentation/>
* MySQL Documentation: <https://dev.mysql.com/doc/>
* JSP and JSTL Guides: Various online tutorials and official documentation.
* Bootstrap Documentation: <https://getbootstrap.com/>
* Additional Resources:
* Spring Data JPA Documentation: <https://spring.io/projects/spring-data-jpa>
* Maven Documentation: <https://maven.apache.org/guides/index.html>

C:\Users\kuntimaddi.lokesh\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\CB00E718.tmp

# 