**ADITYA COLLEGE OF ENGINEERING AND TECHNOLOGY**

**Department of Computer Science & Engineering**

**Surampalem**

**INTERNSHIP REPORT**

**ON**

**“OrderOnTheGo: Your On-Demand Food Ordering Solution”**

**Submitted in partial fulfillment of the requirements of the**

**Virtual Internship Program**

**Organized by**

**SMART BRIDGE**

**Submitted by**

**Kandula Vinay Gupta (23P31A05G5)**

**Kasa Hema Mounika (23P31A05G8)**

**Naga Satya Manasa Pabbu (23P31A05H7)**

**Padala Shanmuka Reddy (23P31A05H8)**

**TEAM ID:**

**LTVIP2025TMID42639**

**Under the Mentorship of**

**Mr. Anji Babu**

**Smart Bridge**

**June 2025**

**Index**

| **S. No.** | **Section Title** | **Page No.** |
| --- | --- | --- |
| **1.** | **BRAINSTORMING & IDEATION** | **1-2** |
| **2.** | **REQUIREMENT ANALYSIS** | **3-5** |
| **3.** | **PROJECT DESIGN** | **5-7** |
| **4.** | **PROJECT PLANNING** | **7-9** |
| **5.** | **PROJECT DEVELOPMENT** | **9-12** |
| **6.** | **FUNCTIONAL & PERFORMANCE TESTING** | **12-14** |
| **7.** | **DEPLOYMENT** | **14** |

# **BRAINSTORMING &IDEATION**

**Objective:**

**Problem Context**

In the fast-paced modern lifestyle, users often face challenges in accessing convenient, reliable, and timely food delivery services. Many local restaurants do not have a digital presence or a unified platform to showcase their offerings and manage online orders efficiently.

**Purpose of the Project**

The purpose of this project is to bridge the gap between hungry customers and local restaurants through a unified digital food ordering system. It empowers:

* Customers to order food effortlessly.
* Restaurants to manage menus and orders efficiently.
* Admins to monitor and manage the ecosystem centrally.

**Key Points:**

* **Problem Statement**
* Customers face difficulties in ordering food, especially at odd hours or from non-chain local restaurants.
* Existing platforms are often too generic or costly for smaller vendors and lack customization.
* Restaurants also struggle to manage orders and menu visibility online without technical expertise.
* **Proposed Solution**

A user-friendly, full stack web application with:

* Real-time menu listings and cart system.
* Admin-controlled platform with restaurant approvals.
* Role-based login(customer, restaurant, admin).
* Secure ordering, order tracking, and delivery details.4
* Restaurant dashboards for inventory/order control.
* **Target Users**
* General users and customers looking to order food online.
* Local restaurants wishing to showcase teir menu and receive online orders.
* Admins responsible for system moderation and restaurant approval.
* **Expected Outcomes**
* Faster and more convenient food ordering experience for users.
* Enhanced digital presence for local food vendors.
* Centralized control for admin to ensure quality and safety.
* Increased business visibility for partner restaurants.

**REQUIREMENT ANALYSIS**

**Objective:**

**Functional Requirements:**

Functionally, the SB Foods platform must allow users to register and log in with unique credentials, browse food menus, add items to their cart, and place orders by providing delivery address and payment method. After placing an order, users should receive confirmation and be able to track their order status from their profile. Restaurants should be able to register, get approval from the admin, and then access a dashboard where they can add, update, or remove food items, view orders placed by customers, and monitor their product listings. Admins must be able to log in to a secure dashboard where they can view all registered users, manage restaurant approvals, and monitor orders placed across the platform. The system should also provide real-time notifications upon order placement and allow smooth navigation for users across various pages. Responsiveness across devices and ease of use are core expectations, ensuring a user-friendly experience for all user roles—customers, restaurants, and admins.

**Technical Requirements:**

The SB Foods application is developed using the MERN stack, requiring the integration of MongoDB, Express.js, React.js, and Node.js. On the backend, Node.js with Express.js is used to build RESTful APIs for handling routing, user authentication, order management, and restaurant operations. MongoDB, connected via Mongoose ODM, serves as the primary database for storing collections such as users, restaurants, food items, and orders. On the frontend, React.js is used to create a dynamic and responsive user interface with components such as Navbar, Cart, Order Summary, Admin Dashboard, and Restaurant Panel. JWT (JSON Web Tokens) is implemented for secure, role-based authentication and authorization of users, restaurants, and administrators. The system relies on npm for package management and uses environment variables for secure configuration. Development tools such as Visual Studio Code, Git for version control, and MongoDB Atlas or Compass for database management are essential. The platform is designed to support deployment on services like Vercel (for the React frontend) and Render or Heroku (for hosting the backend and APIs).

**Key points:**

### **Technical Requirements**:

* Frontend built using React.js with routing and components.
* Backend API with Node.js and Express.js.
* MongoDB as the database with Mongoose ODM.
* RESTful API integration for client-server communication.
* JWT for secure login/authentication.
* Hosting support (e.g., Vercel for frontend, Render for backend).
* Environment variables for secure configuration (.env file).
* **Functional Requirements:**
* User registration/login with role-based authentication.
* Add/remove items to/from cart.
* Place orders with delivery address and payment info.
* Restaurants can add/edit/delete food items.
* Admin can approve/reject restaurants and view reports.
* Display real-time order status updates.
* High availability with mobile-friendly UI.
* Fast response time (<2s for API endpoints).
* Intuitive dashboard for all roles (admin, restaurant, user).
* Notification system for order confirmation.
* Reusable frontend components for UI consistency.
* **Constraints & Challenges:**
* Ensuring Real-time updates without complex WebSockets (basic polling used).
* cross-browser responsiveness
* Limited access to cloud payment gateways in academic prototype.
* Maintaining clean code separation between frontend and backend.
* Handling role-based data access securely on the client side.

**PROJECT DESIGN**

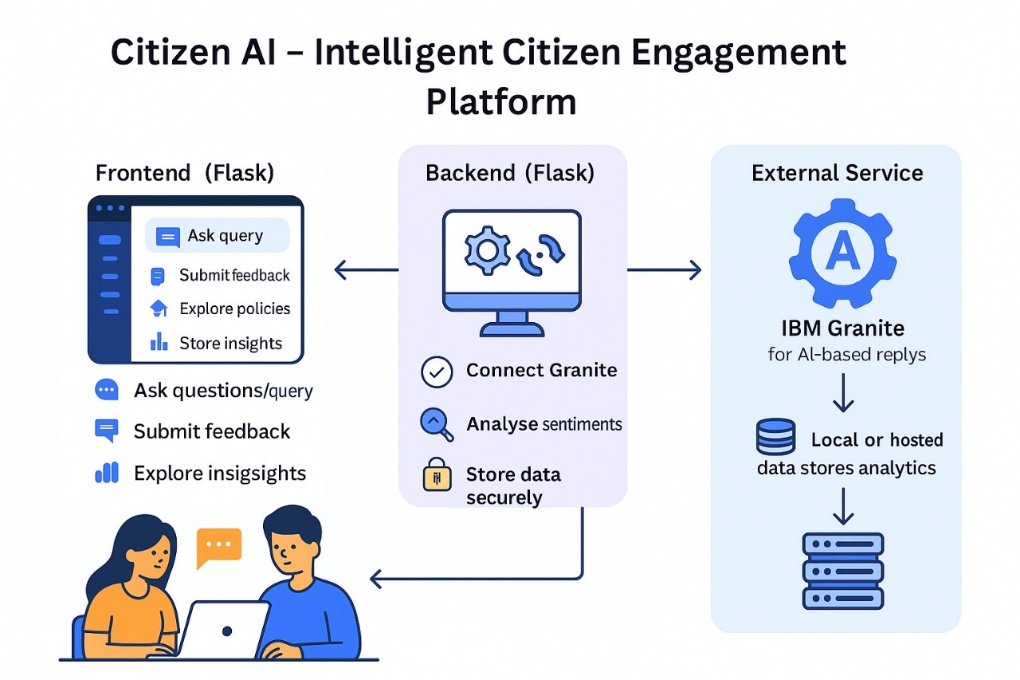
**Objective:**

The primary objective of the SB Foods project is to develop a scalable, responsive, and user-centric food ordering web application using the MERN stack. The platform aims to simplify the process of online food ordering for customers while providing restaurants with tools to manage their menus and orders effectively. It also includes a centralized admin panel to oversee the platform’s operations, ensuring data consistency, user security, and business integrity. The application seeks to bridge the gap between customers and local food vendors, especially during unconventional hours, by offering a reliable, real-time ordering experience.

**Key points:**

The SB Foods application is built with a focus on modular design, real-time usability, and role-based functionality. It includes three primary user roles—customers, restaurant owners, and administrators—each with a tailored interface and access controls. The app supports secure login, menu browsing, cart management, and order placement from the customer side. For restaurants, it enables product management and order monitoring. For admins, the system provides full visibility and control over users, restaurants, and orders. The project emphasizes code maintainability, database efficiency, and a seamless user experience across all devices.

* **System Architecture Diagram:**



Bottom of Form

* **User flow:**

1. **Customer Flow**:
   * The user visits the application, registers or logs in.
   * They can browse available restaurants and menu items.
   * Selected items are added to the cart.
   * The user enters delivery details and selects a payment method.
   * Upon placing the order, the user receives a confirmation and can track the order status from their profile.
2. **Restaurant Flow**:
   * Restaurant owners log in or register and await admin approval.
   * Once approved, they can access the restaurant dashboard.
   * They can manage food items (add, update, delete) and view incoming orders.
3. **Admin Flow**:
   * Admins log in and access a dashboard with tabs for users, restaurants, and orders.
   * They can approve or reject restaurant registrations, manage user accounts, and review all system activities.

* **UI/UX Consideration:**

User Interface (UI) and User Experience (UX) design in SB Foods focuses on intuitive navigation, responsiveness, and visual clarity. The application uses modern web design principles with a clean layout, consistent component styling, and clear visual cues. Important features such as login forms, product listings, and order buttons are prominently placed for ease of use. The application ensures responsiveness across various screen sizes using CSS frameworks and media queries, making it accessible on desktops, tablets, and mobile devices. Visual feedback such as loaders, error messages, and success notifications enhance interactivity and improve usability.

**PROJECT PLANNING**

**Objective:**

The objective of the Project Planning phase is to structure the development of the Citizen AI platform into manageable tasks and timeframes using an Agile methodology. By organizing work into focused sprints and distributing responsibilities across the team, the project ensures steady progress, timely feedback, and successful completion of each module. This structured plan includes sprint goals, task allocation, and milestone tracking.

**Key Points:**

* **Sprint Planning:**
* **Sprint 1: Setup & Authentication**
  + Initialize frontend and backend environment.
  + Connect MongoDB database with backend.
  + Implement user authentication(login/register).
  + Role-based access(admin, restaurant, user)
* **Sprint 2: Restaurant and Product Module**
  + Create restaurant registration and admin approval flow.
  + Implement product management features(add/edit/delete).
  + Set up database schemas for restaurants and food items.
* **Sprint 3: Cart & Order Management**
  + Develop cart functionality with add/remove features.
  + Implement order placement with delivery info.
  + Create customer order history and status tracking.
* **Sprint 4: Admin Dashboard & UI Polishing**
  + Admin views for users, restaurants, and orders
  + Polish UI/UX across components for all roles
  + Add validations, error handling, and user notifications.

**Task Allocation:**

|  |  |
| --- | --- |
| **Members** | **Tasks** |
| K. Vinay Gupta | * **Frontend developer tasks** * Design UI components * Implement routing and protected routes for different roles * Integrate APIs * Handle state management |
| Padala Shanmuka Reddy | * **Backend Developer Tasks** * Set up Express server and connect to MongoDB * Build RESTful API endpoints * Implement JWT * Handle data validation, error responses, and secure access |
| Kasa Hema Mounika | * **Database/Schema Designer** * Define Mongoose * Set up model relationships and validation logic * Ensure optimized querying and indexing |
| Naga Satya Manasa Pabbu | * **Admin & Documentation Lead** * Manage admin dashboard development * Oversee role-based view restrictions * Prepare project documentation * Support deployment setup and testing |

**TimeLine and Milestones:**

|  |  |
| --- | --- |
| **Date** | **Milestone** |
| Week-1 | * Base project setup * Authentication * DB connection |
| Week-2 | * Restaurant product management system |
| Week 3 | |  | | --- | |  |  |  | | --- | | * Cart and complete order workflow | |
| Week 4 | |  | | --- | |  |  |  | | --- | | * Final testing, admin dashboard, UI polish, documentation, Github | |

**PROJECT DEVELOPMENT**

**Objective:**

The development of the SB Foods food ordering application followed a **modular and iterative approach**, aligning with real-world Agile principles. The project was divided into **four key phases (sprints)** over a span of four weeks, each targeting specific milestones in the application lifecycle — from setup and authentication to feature integration and final deployment.

**Key Points:**

**1. ⚙️ Environment & Project Setup**

* Initialized frontend using React.js and backend with Node.js + Express.js.
* Set up MongoDB using Mongoose ODM for defining models and connecting collections.
* Installed necessary dependencies (axios, jsonwebtoken, cors, mongoose, bcrypt, etc.).
* Created a structured folder architecture for clarity and scalability**.**

**2. 🔐 Authentication & Role Management**

* Implemented JWT-based authentication to securely log in users.
* Established Role-Based Access Control (RBAC) for three roles: Customer, Restaurant, and Admin.
* Used route protection on frontend (React) and middleware in backend (Express) to restrict access.

**3. 🧱 Backend API Development**

* Designed and developed RESTful APIs for:
* User Registration & Login
* Product CRUD Operations
* Restaurant Approval Workflow
* Cart Management
* Order Placement & History
* Used Express routers to modularize API logic and ensure maintainability.

**4. 🎨 Frontend Component Development**

* Built reusable React components such as:
* Login Form, Register Form, Product Card, Cart Item, Order Summary
* Integrated frontend with backend APIs using Axios.
* Applied conditional rendering based on user roles for dashboard access and feature visibility.

**5. 🛍️ Core Features Implementation**

* Developed product browsing and detailed views with dynamic filtering.
* Enabled cart operations (add, update, remove) with session persistence.
* Implemented order checkout with address and payment method inputs.
* Created profile pages for users to view order history and status**.**

**6. 📊 Restaurant & Admin Dashboards**

* Restaurant users can:
* Add/Edit/Delete menu items
* View incoming orders
* Admins can:
* Approve new restaurants
* View all users, restaurants, and orders
* Monitor system activity through a centralized panel

**7. 📱 Responsive UI/UX**

* Applied Bootstrap grid system and custom media queries for responsive design.
* Ensured mobile-first design approach for accessibility across devices.
* Added feedback elements (alerts, loaders, notifications) for better UX**.**

**8. ✅ Testing & Validation**

* Performed manual testing for all user flows (customer, restaurant, admin).
* Handled form validations and error responses on both client and server sides.
* Tested CRUD APIs using tools like Postman**.**

**9. 🚀 Deployment & Documentation**

* Frontend deployed on Vercel, backend deployed on Render.
* Managed environment variables using .env files.
* Finalized project documentation including:
* README file
* ER Diagram
* System Architecture
* Sprint Plan and Timeline
* Uploaded complete project to GitHub with proper version control**.**

**FUNCTIONAL & PERFOMANCE TESTING**

**Objective:**

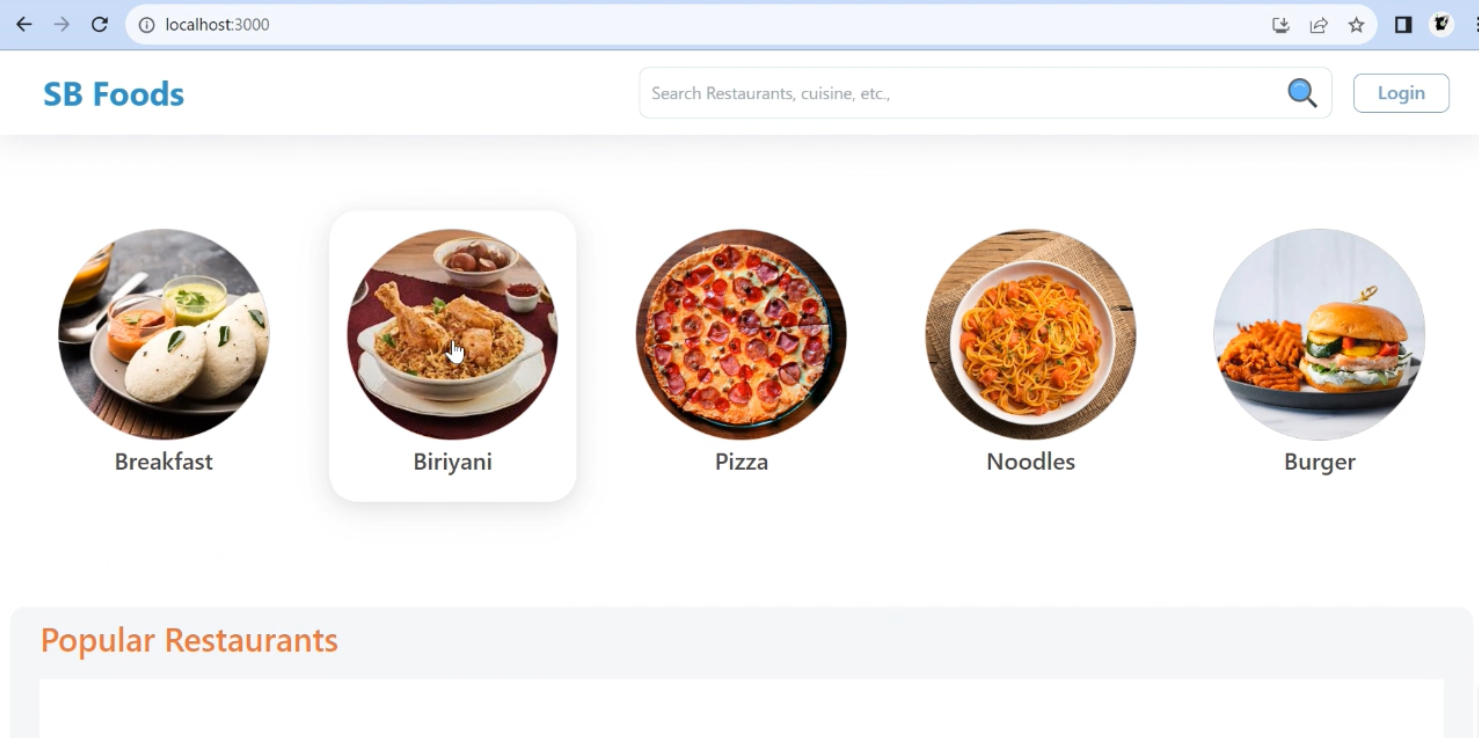
The objective is to ensure that all features of the SB Foods app work correctly as intended (functional testing) and that the app performs efficiently under different user loads without lag or crashes (performance testing). This helps deliver a smooth, secure, and reliable user experience for customers, restaurants, and admins.

**Key Points:**

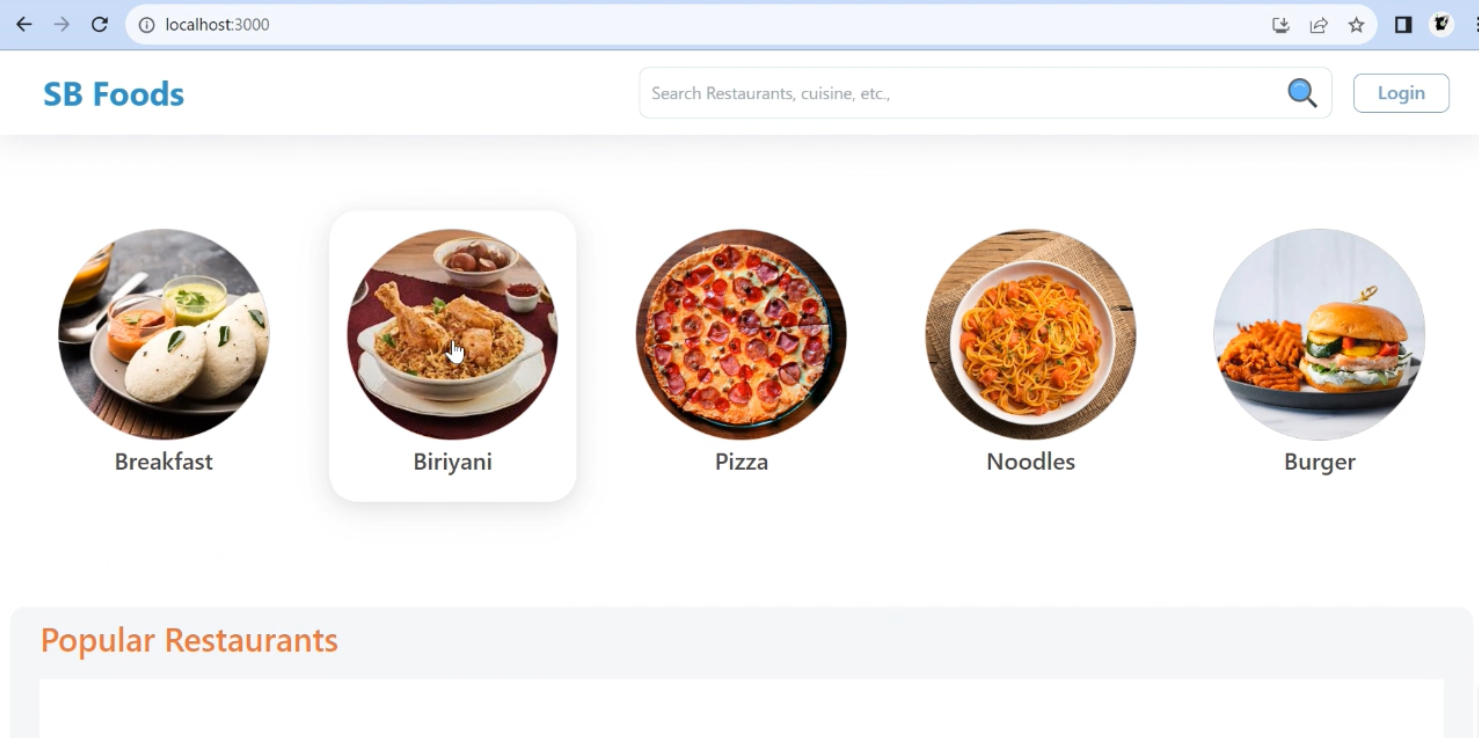
* **Test Cases Executed:**
* Tested user login/logout with valid and invalid inputs.
* Verified role-based dashboard access control.
* Checked product listing and search filter functionality.
* Ensured cart operations (add, remove, update) worked correctly.
* Tested full order placement flow from cart to confirmation.
* Validated admin approval and rejection of restaurants.
* Confirmed restaurant menu management (CRUD operations).
* Checked UI responsiveness on different screen sizes.
* Ran form validation tests across all input forms.
* Verified session handling and token expiration..
* **Bug Fixes & Improvements:**
* Fixed role-based redirection after login.
* Resolved product not displaying in the cart issue.
* Corrected API mismatch between frontend and backend.
* Improved loading speed with backend query optimization.
* Fixed broken navigation links in restaurant dashboard.
* Enhanced form error messages for better clarity.
* Fixed order history not showing for users.
* Corrected mobile UI alignment in cart page.
* Patched missing validation on admin forms.
* Improved user feedback using toast notifications.

.

* **Final Validation:**
* All major user flows (login, cart, checkout) tested end-to-end successfully.
* Role-based access (user, restaurant, admin) validated without conflicts.
* UI responsiveness verified on mobile, tablet, and desktop.
* Final deployment tested for functionality and stability.
* No critical bugs found; app ready for submission and production use
* **Deployment:**

Top of Form

Bottom of Form

****

