**Project Design Phase**

**Problem – Solution Fit Template**

|  |  |
| --- | --- |
| Date | 26 June 2025 |
| Team ID | LTVIP2025TMID59601 |
| Project Name | OrderOnTheGo: Your On-Demand Food Ordering Solution |
| Maximum Marks | 2 Marks |

**Problem – Solution Fit Template:**

The project is a food delivery system designed to connect customers with local restaurants for efficient food ordering and delivery. The solution aims to solve the problem of easily accessing menus, placing orders, and tracking deliveries, while also providing restaurants with a system to manage their operations.

**Purpose:**

❑ Solve complex problems in a way that fits the state of your customers.

❑ Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behaviour.

❑ Sharpen your communication and marketing strategy with the right triggers and messaging.

❑ Increase touch-points with your company by finding the right problem-behaviour fit and building trust by solving frequent annoyances, or urgent or costly problems.

❑ **Understand the existing situation in order to improve it for your target group.**

**Solution Requirements**

The project has been defined with both functional and non-functional requirements.

* **Functional Requirements:** The functional requirements are broken down into epics and user stories, which detail the system's functionality from a user's perspective.
  + **User Registration & Login:** As a user, you can register by entering your email and password or through Gmail and log into the application. You will receive a confirmation email upon registration.
  + **Ordering & Tracking:** Customers can view restaurants and menus, add items to a cart, place an order, and track its status.
  + **Restaurant Management:** Restaurant owners can register, manage their details, add/edit/delete food items on their menus, and process orders.
  + **Admin Functions:** The administrator can view users, approve restaurant accounts, and manage categories.
* **Non-functional Requirements:** The system's non-functional requirements include usability, security, reliability, performance, availability, and scalability.

**Technical Architecture & Technology Stack**

The project's architecture is a multi-tier solution that separates the user interface, application logic, and data storage. The architecture includes external interfaces for third-party APIs.

* **User Interface:** The user interface is a web UI built using React.js, HTML, CSS, and JavaScript.
* **Application Logic (Backend):** The core logic is handled by a backend built with Node.js and Express.js.
* **Database:** A MongoDB database is used for data storage.
* **External API:** A payment gateway API is used for transactions.
* **Infrastructure:** The application is deployed on a local server but is designed to be deployed on a cloud environment using technologies like Docker.
* **Scalability:** The architecture is designed for scalability with features like load balancers and distributed servers to ensure high availability and handle a high volume of requests per second.

**Project Planning & Agile Methodology**

The project follows an agile methodology, using sprints, user stories, and story points for planning and estimation.

* **Sprints:** The project is divided into fixed-duration sprints to complete a set of tasks. The sprints are based on user stories and story points.
* **Story Points:** A story point is a number that represents the effort a task requires. A "Very Easy task" might be 1 point, while a "Difficult task" is 5 points.
* **Velocity:** The team's velocity is calculated based on the total story points completed per sprint.

**Data Flow**

A Data Flow Diagram (DFD) visually represents the flow of information within the system. Data enters and leaves the system, is changed, and is stored in different places. In the food delivery system, data flows from the user, through the application logic, and to the database, with interactions for processing orders and payments.

**Template:**

