Industrial Internship Report

on

"Food Delivery System"

Prepared by

Kusa Navya

| *Execu ve Summary* |
| --- |
| This report provides details of the Industrial Internship provided by upskill Campus and The IoT Academy in collabora on with Industrial Partner UniConverge Technologies Pvt Ltd (UCT).

 This internship was focused on a project/problem statement provided by UCT. We had to finish the project including the report in 6 weeks' me.

 My project was Food Delivery System. Food delivery is a courier service in which a restaurant, store, or independent food-delivery company delivers food to a customer.

 This internship gave me a very good opportunity to get exposure to Industrial problems and design/implement solu on for that. It was an overall great experience to have this internship. |

TABLE OF CONTENTS

1 Preface

It was a good experience learning from this internship.

It is very important to have relevant Internship in career development and to explore the things deeper.

My project was Food Delivery System. Food delivery is a courier service in which a restaurant, store, or independent food-delivery company delivers food to a customer.

Opportunity given by USC/UCT.

How Program was planned
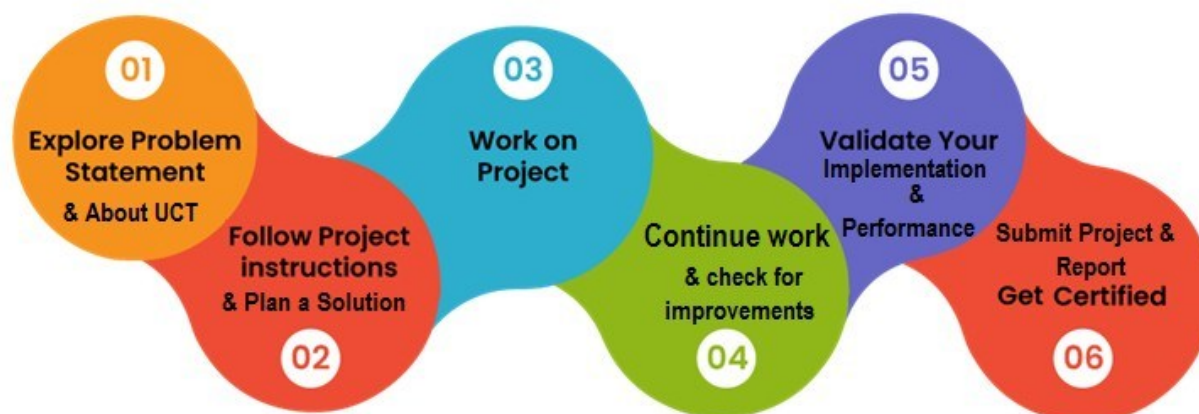
I have done internship on Full stack Development and learnt a lot from it and also built an applica on using python.

Thanks to all especially Ujjwal Mishra sir, and also who have helped you directly or indirectly.

My sugges on for juniors : it is the best way to learn new technologies and gain hands on experience.

2 Introduction

2.1 About UniConverge Technologies Pvt Ltd

A company established in 2013 and working in Digital Transforma on domain and providing Industrial solu ons with prime focus on sustainability and RoI.

For developing its products and solu ons it is leveraging various **Cu ng Edge Technologies e.g. Internet of Things (IoT), Cyber Security, Cloud compu ng (AWS, Azure), Machine Learning, Communica on Technologies (4G/5G/LoRaWAN), Java Full Stack, Python, Front end** etc.

i. UCT IoT Platform (

**UCT Insight** is an IOT pla orm designed for quick deployment of IOT applica ons on the same me providing valuable "insight" for your process/business. It has been built in Java for backend and ReactJS for Front end. It has support for MySQL and various NoSql Databases.

- It enables device connec vity via industry standard IoT protocols - MQTT, CoAP, HTTP, Modbus TCP, OPC UA

- It supports both cloud and on-premises deployments.

 It has features to
- Build Your own dashboard
- Analy cs and Repor ng
- Alert and No fica on
- Integra on with third party applica on(Power BI, SAP, ERP)
- Rule Engine
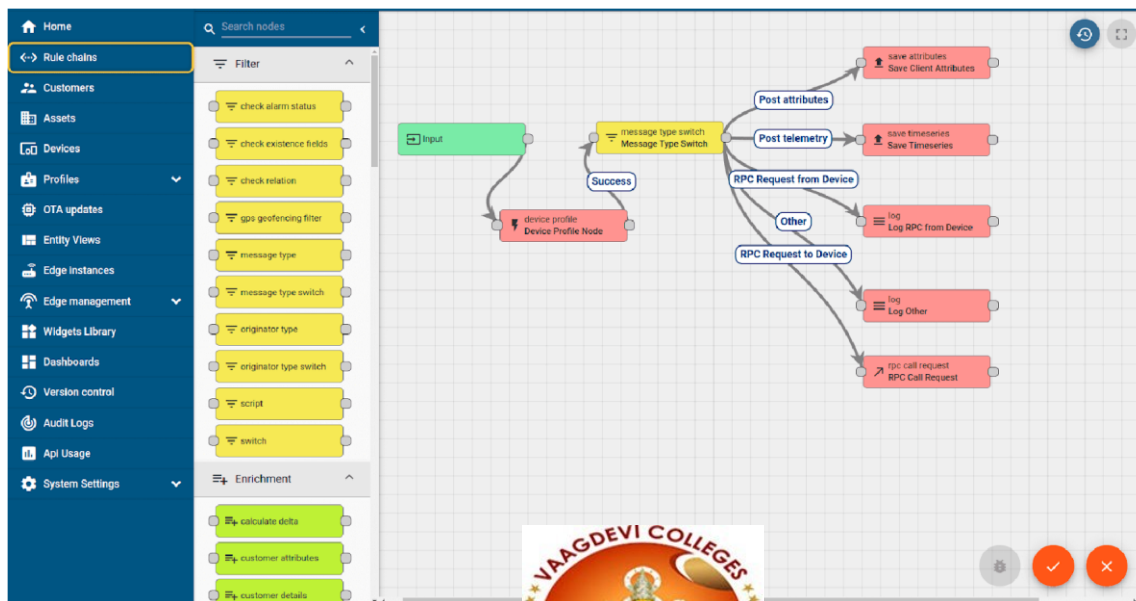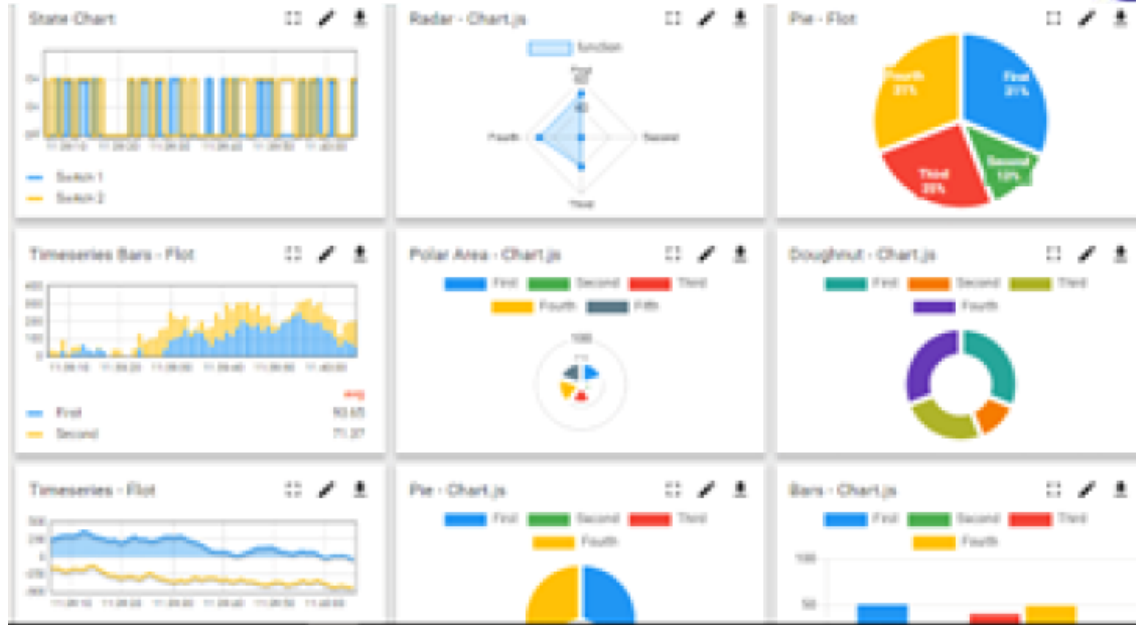
ii.   **Smart Factory Pla orm (** **FACTORY WATCH** **)**

Factory watch is a pla orm for smart factory needs.

It provides Users/ Factory

- with a scalable solu on for their Produc on and asset monitoring

- OEE and predic ve maintenance solu on scaling up to digital twin for your assets.

- to unleased the true poten al of the data that their machines are genera ng and helps to iden fy the KPIs and also improve them.

- A modular architecture that allows users to choose the service that they what to start and then can scale to more complex solu ons as per their demands.

Its unique SaaS model helps users to save me, cost and money.

Machine	Operator	Work Order ID	Job ID	Job Performance	Job Progress		Output		Rejection	Time (mins)				Job Status	End Customer
					Start Time	End Time	Planned	Actual		Setup	Pred	Downtime	Idle		
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
CNC_S7_81	Operator 1	WO0405200001	4168	58%	10:30 AM		55	41	0	80	215	0	45	In Progress	i
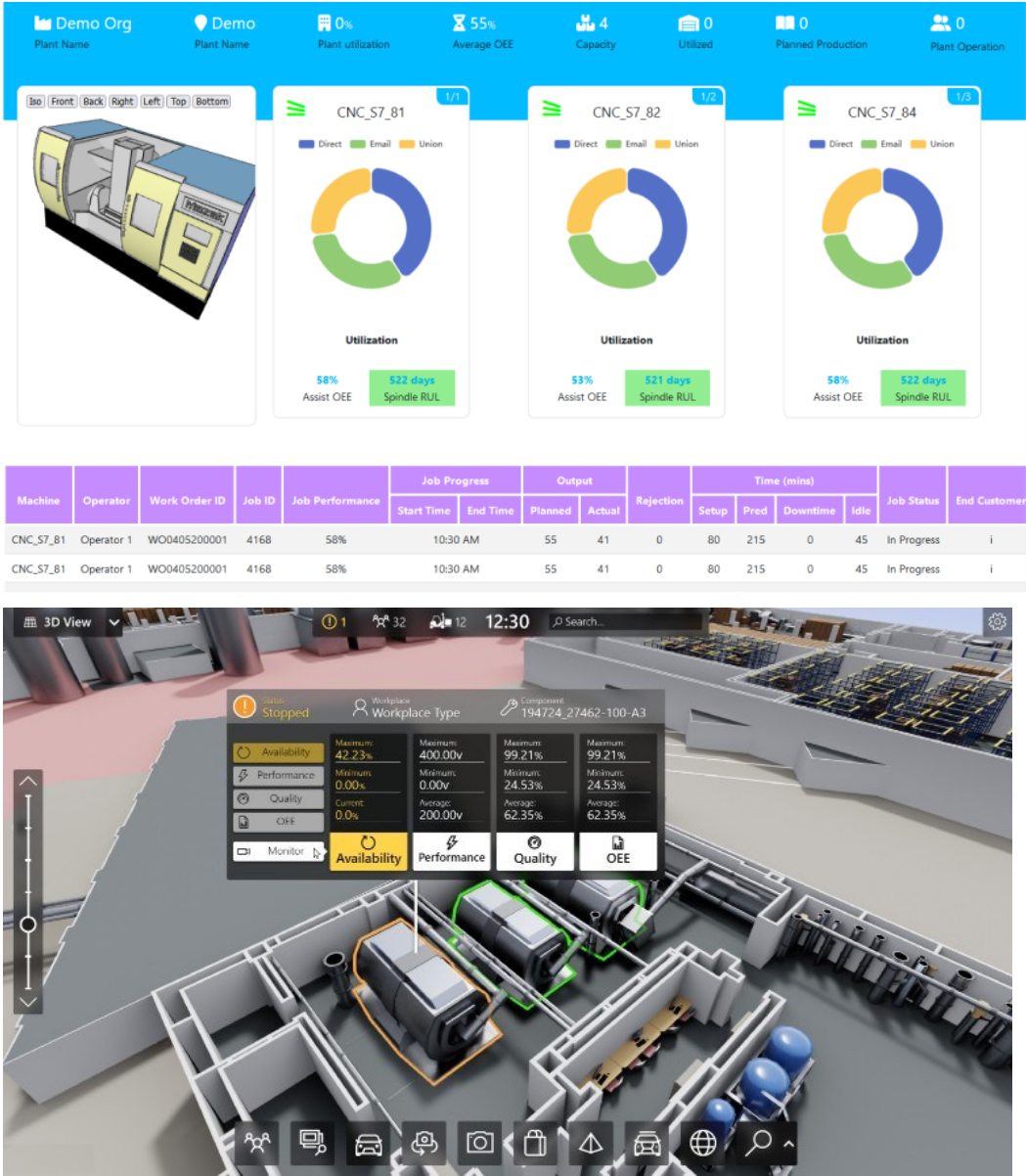
iii.   LoRaWAN based Soluon

UCT is one of the early adopters of LoRAWAN teschnology and providing solu on in Agritech, Smart ci es, Industrial Monitoring, Smart Street Light, Smart Water/ Gas/ Electricity metering solu ons etc.

iv.   Predic ve Maintenance

UCT is providing Industrial Machine health monitoring and Predic ve maintenance solu on leveraging Embedded system, Industrial IoT and Machine Learning Technologies by finding Remaining useful life me of various Machines used in produc on process.
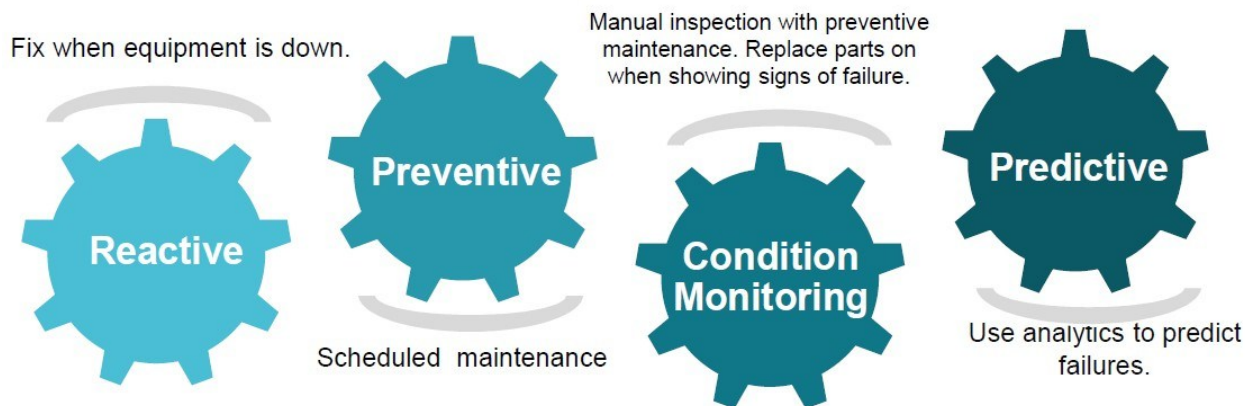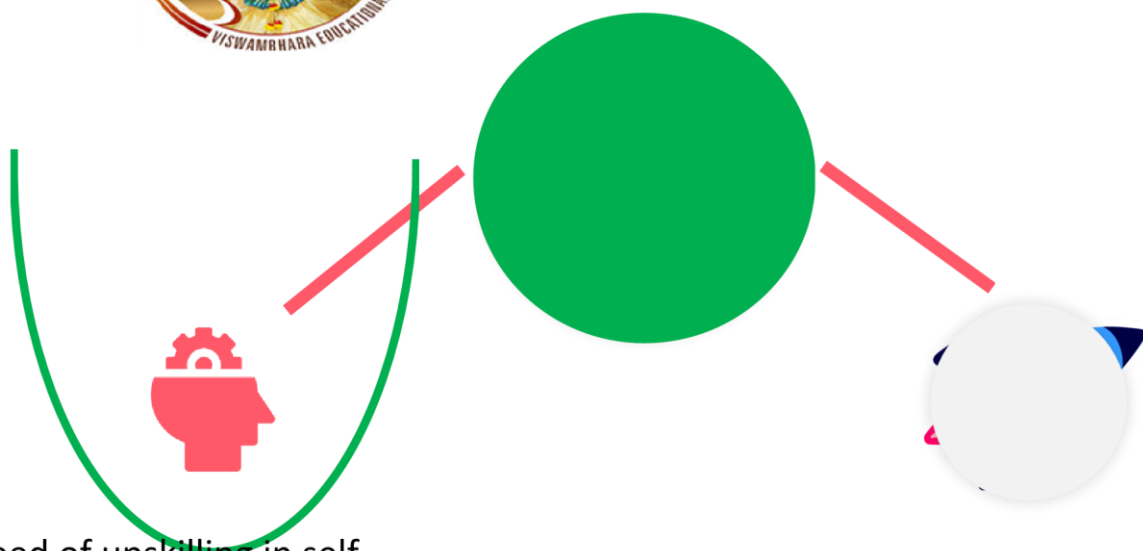

2.2 About upskill Campus (USC)

upskill Campus along with The IoT Academy and in association with Uniconverge technologies has facilitated the smooth execution of the complete internship process.

USC is a career development platform that delivers **personalized executive coaching** in a more affordable, scalable and measurable way.

College Logo]

Seeing need of upskilling in self paced manner along-with additional support services e.g. Internship, projects, interaction with Industry experts, Career growth Services

upSkill Campus aiming to upskill 1 million learners in next 5 year

https://www.upskillcampus.com/

2.3 The IoT Academy

The IoT academy is EdTech Division of UCT that is running long execu ve cer fica on programs in collabora on with EICT Academy, IITK, IITR and IITG in mul ple domains.

2.4 Objectives of this Internship program

The objec ve for this internship program was to ☛

get prac cal experience of working in the industry.

☛ to solve real world problems.

☛ to have improved job prospects.

☛ to have Improved understanding of our field and its applica ons.

☛ to have Personal growth like be er communica on and problem solving.

2.5 Reference

[1]   Reference1

[2]   Reference2

[3]   Reference3

2.6 Glossary

Terms	Acronym
P **2C**	Platform to Customer
R **2C**	Restaurant-to-Consumer
G **PC**	Global Positioning System

U **I/UX**	User Interface/User Experience
P **OS**	Point of Sale

3 Problem Statement

In the assigned problem statement

Developing a food delivery system involves addressing challenges that ensure seamless interactions between customers, restaurants, and delivery partners. Below is a concise problem statement:

"Design a comprehensive and scalable food delivery system that connects customers with a wide variety of restaurants, facilitates efficient order placement and real-time tracking, and ensures timely delivery while maintaining food quality and minimizing operational costs."
Key Challenges to Address:

1. **User Experience (UX):** Creating an intuitive platform for users to browse menus, customize orders, and make payments.

2. **Delivery Logistics:** Optimizing delivery routes and managing delays due to traffic or weather conditions.

3. **Restaurant Integration:** Ensuring smooth integration with restaurant systems for inventory, order management, and menu updates.

4. **Real-Time Tracking:** Providing accurate GPS tracking for orders to enhance transparency.

5. **Scalability:** Building a system that can handle a growing user base and fluctuating demand during peak hours.

6. **Payment Security:** Implementing secure and diverse payment methods to build customer trust.

7. **Customer Support:** Offering prompt assistance to address complaints and queries efficiently.

This problem statement and its components can serve as a foundation for creating a functional and user-friendly food delivery system. Let me know if you'd like examples or references for similar systems!

4 Existing and Proposed solution

Exis ng Solu ons for Food Delivery Systems

Aggregator Pla orms (e.g., Uber Eats, DoorDash, Grubhub):

- o   How they work: These pla orms act as intermediaries between customers and restaurants. Customers can browse a variety of menus from different restaurants and

place their orders through a central app or website. The pla orm handles order placement, payment processing, and delivery logis cs via third-party drivers.

- o Challenges: Issues include high delivery fees, quality inconsistency, dependency on third-party delivery drivers, and late deliveries. Addi onally, restaurants o en have to pay a commission fee, which can impact their profitability.

==Proposed / Op mized Delivery Logis cs Using AI and Automa on :==

- **How it works** : AI and machine learning can be used to op mize delivery routes, predict busy mes, and even automa cally dispatch the best-suited delivery drivers. This would help reduce delays and increase the efficiency of food deliveries.

- **Proposed Benefits** : Faster deliveries, reduced delivery costs, and improved customer sa sfac on.

- **Example** : AI-powered systems like those being implemented by Uber Eats and DoorDash to predict demand and streamline delivery .

==Hybrid Delivery Combina on of Aggregators and In-House Delivery:==

- **How it works** : Restaurants could partner with pla orms for marke ng and order aggrega on while also using their own delivery fleet. This hybrid system would allow restaurants to maintain be er control over delivery mes and customer service while s ll benefi ng from the reach of pla orms.

4.1 Code submission link

4.2 Report submission link

4.2

5 Proposed Design/ Model

 The proposed food delivery system incorporates various components to provide an efficient, scalable, and user-friendly service. Below is a breakdown of the design structure, which includes both high-level architecture and technical components.

The system can be broken down into four primary components:

1. **User Interface (Front-End)** :

    o   Customers interact with the pla orm through a **web applica on** or **mobile app** (iOS/Android).   o   **Restaurant dashboard** for restaurant owners to manage orders, update menus, and track deliveries.

    o   **Admin panel** to manage users, restaurant lis ngs, and overall system opera ons.

2. **Back-End (Server-Side)** :

    o   The back-end handles the **business logic** , user requests, order processing, and API endpoints.

    o   **Database** stores user, order, restaurant, and payment informa on.   o   **Payment**

    Gateway integrates with third-party services to handle transac ons securely.

    o   **Order Management System (OMS)** manages order status and coordinates between restaurants, customers, and delivery drivers.

3. **Delivery Logis cs** :

    o   **AI-powered route op miza on** ensures drivers take the most efficient paths.   o   **Driver**

    Tracking provides real- me GPS updates to customers and admins.

    o   **Green Logis cs** : Electric vehicles or bicycles used for deliveries to reduce carbon footprint.

4. **External Integra ons** :

o **Payment Gateways** : Supp ... rds, digital wallets, and even cryptocurrency.
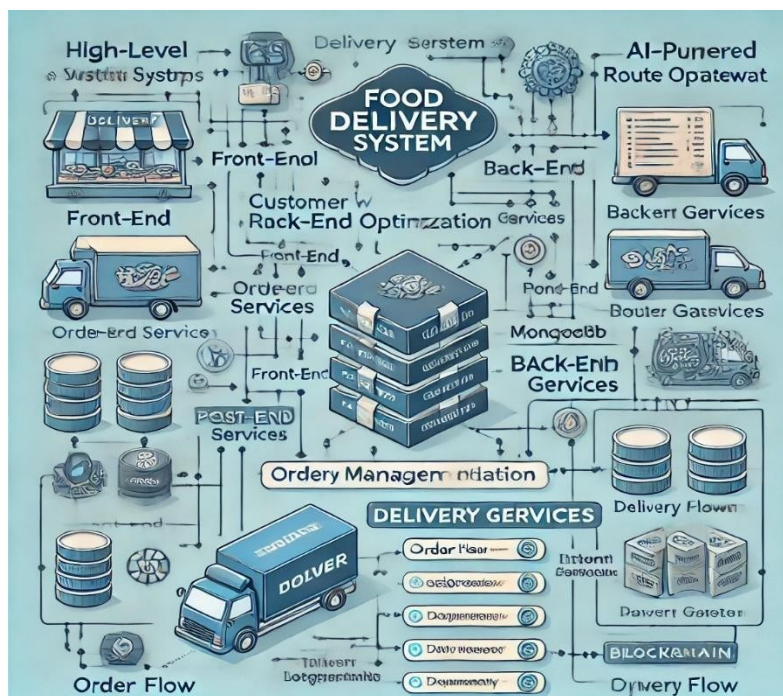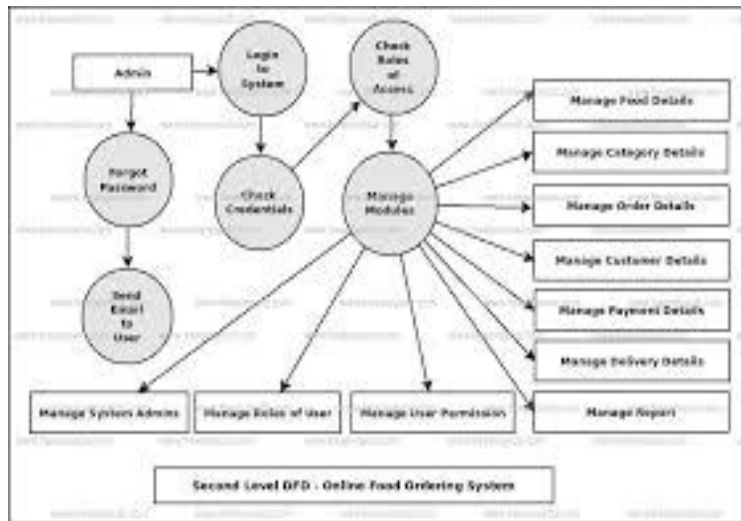
5.1 High Level Diagram (if applicable)

Figure 1: HIGH LEVEL DIAGRAM OF THE SYSTEM

5.2 Low level Diagram (if applicable)

Second Level DFD - Online Food Ordering System

5.3 Interfaces (if applicable)

A food delivery app typically has several interfaces that cater to both customers and service providers (drivers, restaurants). Here are the key interfaces and their features:

- **1. User (Customer) Interface:**

- **Home Screen:**

    o   Search bar for restaurants, cuisines, or dishes.

    o   Categories (e.g., fast food, fine dining, desserts, etc.).

    o   Featured or recommended restaurants and promo ons.

- **Restaurant Lis ngs:**

    o   Restaurant name, image, and ra ngs.   o   Filter op ons (price range, delivery me, ra

    ngs).

    o   Sor ng op ons (distance, po

-

  - **Restaurant (Merchant) Interface:**

  - **Dashboard:**

Overview of recent orders.

  - o Current order status (pending, in progress, delivered).

  - **Order Management:**

    - o Accept/reject orders.

    - o Real- me updates for the kitchen (order received, preparing, ready to deliver).

  - **Menu Management:**

    - o Add, edit, or remove dishes.

    - o Set pricing and availability (e.g., special items or daily deals).

  - **Reports and Analy cs:**

    - o Sales data and performance metrics.

    - o Customer feedback and ra ngs.

  - **Driver (Delivery Partner) Interface:**

  - **Dashboard:**

    - o Available orders nearby. o Ac ve orders with pickup and delivery loca ons. ●

      Naviga on:

    - o In-app GPS naviga on to restaurants and delivery addresses. o Traffic and route op

      miza on.

6 Performance Test

This is very important part and defines why this work is meant of Real industries, instead of being just academic project.

Here we need to first find the constraints.

Load tes ng assesses how the system performs under expected traffic volumes. For a food delivery system, the typical traffic includes:

- **Peak Load** : A large number of users browsing restaurants, placing orders, and making payments simultaneously (e.g., during meal mes).

- **Concurrent Users** : Test how many users can interact with the system concurrently without degrada on in performance.

Tools for Load Tes ng :

- **Apache JMeter** : Can simulate mul ple users accessing the pla orm, browsing, ordering, and making payments.

- **Gatling** : Another powerful tool for simula ng user interac ons and analyzing system performance under load.

Test Scenarios :

- Simula ng 100,000 users browsing the restaurant menu at once.

- Simula ng 50,000 users placing orders in a short period.

Key Metrics to Measure :

- Response me (me to load the web page or complete a transac on).

- Throughput (requests per second).

- Resource u liza on (CPU, memory, disk, etc.).

6.3 Test Plan/ Test Cases

1. **Test Case 1: User Registra on**   o   **Descrip on** : Verify that customers and restaurant admins can register successfully.

- o **Steps** :

  1. Open the registra on page.

  2. Enter valid user details.

  3. Submit the form.

- o **Expected Result** : User account is created successfully, and a confirma on message is shown.

2. **Test Case 2: Browse Restaurants**   o   **Descrip on** : Ensure that customers can view available restaurants.

- o **Steps** :

  1. Open the app/website.

  2. Search or browse through restaurants.

- o **Expected Result** : Restaurants appear with valid details like name, ra ngs, menu items.

3. **Test Case 3: Place an Order**   o   **Descrip on** : Verify customers can place an order.

- o **Steps** :

  1. Add items to the cart.

  2. Proceed to checkout.

  3. Select delivery method and payment op on.

  4. Complete the order.

- o **Expected Result** : Order   and a confirma on message is displayed.

4. **Test Case 4: Order Status Update**

- o **Descrip on** : Verify that the restaurant can update the status of an order (e.g., Preparing, Out for Delivery).

- o **Steps** :

  1. The restaurant logs into the system.

  2. Select the order and update the status.

- o **Expected Result** : The order status is updated and visible to the customer.

5. **Test Case 5: Payment Processing** o **Descrip on** : Validate the system handles payments correctly.

- o **Steps** :

  1. Customer places an order.

  2. Selects a payment method (e.g., credit card, PayPal).

  3. Completes the payment process.

- o **Expected Result** : Payment is processed successfully, and the transac on is recorded.

---

6. **Test Case 6: Load Tes ng (Concurrent Users)** o **Descrip on** : Test system performance when handling mul ple users.

- o **Steps** :

  1. Simulate 1000, 5000, and 10000 concurrent users browsing and placing orders.

- o **Expected Result** : The system should maintain acceptable performance (response me under 3 seconds per page).

[Your College Logo]

6.4 Test   Procedure

The test procedure outlines the systema c steps to be followed when tes ng the food delivery system. It covers func onal, performance, security, and usability tests, ensuring the system works as intended and meets the specified requirements. Here is a detailed test procedure for various tes ng types:

- **1. Prepara on Phase**

- **Install and configure required tools** (e.g., JMeter, Selenium, Postman).

- **Set up the test servers** (produc on, staging, or tes ng environments) where the system components will be deployed.

- **Install the applica on** (web/mobile client, server-side, database).

- **Configure databases** (MySQL/PostgreSQL for rela onal data, MongoDB for unstructured data).

- **Prepare test data** (sample user profiles, restaurant menus, orders, and payment details).

- **Set up monitoring tools** (e.g., New Relic, Prometheus, or Grafana) to observe system performance.

1.2 Test Case Iden fica on:

- Iden fy the test cases to be executed based on the requirements and test plan (func onal, performance, security, and database tests).

- **2. Func onal Tes ng Procedure**

2.1 User Registra on Tes ng:

- **Objec ve** : Ensure the user can successfully register in the system.

- **Steps** :

   1. Navigate to the registra on page on the website or app.

   2. Input valid user details (name, email, password, etc.).

   3. Submit the registra on form.

   4. Verify that the system sends a confirma on email or message to the user.

- **Expected Result** : Registra on is successful, and the user is redirected to the login screen.

2.2 Order Placement Tes ng:

- **Objec ve** : Ensure that the order process works correctly.

- **Steps** :

  1. Log in as a customer.

  2. Browse the menu and select items.

  3. Add items to the cart and proceed to checkout.

  4. Select the delivery method (pickup or delivery).

  5. Enter payment details and complete the purchase.

- **Expected Result** : The order is placed successfully, and the customer receives an order confirma on with an es mated delivery me.

2.3 Payment Gateway Integra on Tes ng:

- **Objec ve** : Test the integra on of the payment gateway (e.g., Stripe, PayPal).

- **Steps** :

  1. During the order placement, select a payment op on.

  2. Enter payment details (credit card, debit card, PayPal).

  3. Complete the payment.

  4. Verify that the payment is processed and recorded.

- **Expected Result** : Payment is processed securely, and the transac on is recorded in the system.

- **3. Performance Tes ng Procedure**

3.1 Load Tes ng:

- **Objec ve** : Evaluate system performance under expected traffic.

- **Steps** :

  1. Use **JMeter** or **Gatling** to simulate a number of concurrent users (e.g., 1000 users).

  2. Simulate users browsing the menu, placing orders, and comple ng payments.

3. Monitor server performance (response me, CPU, memory usage).

4. Gradually increase the number of users to simulate peak traffic.

- **Expected Result** : The system should be able to handle the load without significant performance degrada on.

6.5 Performance Outcome

The **performance outcome** of a food delivery system is a key indicator of its reliability, scalability, and efficiency. Performance outcomes are measured across various domains, including **response me** , **throughput** , **scalability** , **resource usage** , and **error rates** . Here's an outline of what the expected performance outcomes should be for a food delivery system:

- **1. Response Time**

- **Expected Outcome** : The system should respond within acceptable me frames even under load.

  - **Key metrics** :

    - **Page Load Time** : Less than 2-3 seconds for menu browsing, restaurant details, and checkout pages.

    - **Order Placement** : Order confirma on should be received within 5 seconds a er submission.

    - **Payment Processing** : Payment confirma on should be completed within 5-10 seconds.

  - **Impact** : Longer response mes can lead to a nega ve user experience, poten ally abandoning carts or orders.

- **2. Throughput**
- **Expected Outcome** : The system should handle a high volume of requests per second without performance degrada on.

---

o **Key metrics** :

- **Orders per Minute** : The system should process hundreds

  or thousands of orders per minute, depending on peak traffic.

- **Simultaneous Users** : The system should handle thousands of concurrent users

  browsing menus, placing orders, and making payments without any slowdown or

  errors.

o **Impact** : If throughput is too low, the system may become unresponsive during peak
traffic mes, such as lunch or dinner hours.

- **3. Scalability**

- **Expected Outcome** : The system should scale horizontally or ver cally to handle increased load.

  o **Key metrics** :

  - **Horizontal Scaling** : Ability to add more web servers or services without

    significant performance loss.

  - **Ver cal Scaling** : Ability to upgrade server resources (CPU, memory) without

    significant disrup ons to service.

  o **Impact** : The system must scale effec vely during high-demand periods (e.g., holidays,
  sales) to maintain service levels.

- **4. Resource U liza on**

- **Expected Outcome** : The system should use resources efficiently under normal and peak loads.

  o **Key metrics** :

  [Your College Logo]

  - **CPU Usage** : Should stay under 80% under normal load; higher during peak

    traffic but should not reach 100%.

---

7. My learnings

Over the course of working with the food delivery system project, I have gained insights into several cri cal areas of so ware development, performance op miza on, and system design, all of which are pivotal for my career growth. Here is a summary of key takeaways and how they will shape my professional development:

==So ware Development & Full-Stack Design==

I've learned how to structure and build a food delivery system from both the client and server sides. This includes:

- **Front-end technologies** : Building intui ve user interfaces for customers (HTML, CSS, JavaScript).

- **Back-end technologies** : Designing robust back-end systems using Python (Flask/Django), handling requests, payments, and data storage (MySQL, MongoDB).

- **APIs** : Integra ng third-party services, such as payment gateways and map services, to enhance func onality and user experience.

==Career Impact== :

This full-stack development experience is invaluable for posi ons such as **Full Stack Developer** , **Back End Engineer** , and **So ware Architect** , where building integrated systems is essen al. The ability to handle both client-facing and server-side concerns will make me a versa le asset in tech teams.

8. Future work scope

Here are some ideas that could not be implemented due to me limita ons but can be explored and incorporated in the future development of the food delivery system:

- **1. Advanced AI for Personalized Recommenda ons**

- **Idea** : Integrate a machine learning algorithm to suggest personalized food op ons based on users' order history, preferences, dietary restric ons, and even mood (based on me of day or weather).

- **Future Benefits** : This would improve user experience by tailoring the pla orm to individual tastes and increase user reten on by providing relevant sugges ons.

- **2. Mul -Vendor Marketplace**

- **Idea** : Develop a mul -vendor feature that allows customers to order from mul ple restaurants in a single checkout process. This would be par cularly useful for groups who want different types of food (e.g., pizza from one restaurant and burgers from another).

- **Future Benefits** : Increased convenience for users, higher order volumes, and a broader range of customer needs being met in one place.

- **3. Real-Time Delivery Tracking with Augmented Reality (AR)**

- **Idea** : Use augmented reality to display the real- me loca on of the delivery in the user's app, overlaying the delivery route and es mated me of arrival (ETA) on a map or even within the real-world view through the phone's camera.

- **Future Benefits** : This would provide a more immersive and engaging customer experience, improving trust in delivery mes and adding a cu ng-edge feature to the pla orm.

[Your College Logo]