**FOOD KART**

**A PROJECT REPORT**

**for**

**Mini Project (KCA353)**

**Session (2023-24)**

**Submitted by**

**Group Code – (GA24)**

**RAVI KUMAR**

**(2200290140122)**

**Submitted in partial fulfillment of the**

**Requirements for the Degree of**

**MASTER OF COMPUTER APPLICATION**

**Under the Supervision of**

**Dr. Prashant Agrawal**

**Associate Professor**



**Submitted to**

**Department Of Computer Applications**

**KIET Group of Institutions, Ghaziabad**

**Uttar Pradesh-201206**

**(JAN 2024)**

**CERTIFICATE**

Certified that **Ravi Kumar( 2200290140122)** has/have carried out the project work having “**FOODKART**” (**Mini Project-KCA353**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU**)** (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

**Date:**

**Ravi kumar 2200290140122**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

**Mr. Prashant Agrawal Dr. Arun Tripathi**

**Associate Professor Head of Department**

**Department of Computer Applications Department of Computer Applications**

**KIET Group of Institutions, Ghaziabad KIET Group of Institutions, Ghaziabad**

**Foodkart**

**Ravi Kumar**

**ABSTRACT**

The project revolves around the development of a sophisticated online food ordering system using the Django web framework. This endeavour aims to address the growing demand for seamless and efficient food delivery services in an increasingly digital world. By leveraging Django's powerful features, the system promises robustness, scalability, and an intuitive user interface.

The system is designed to encompass various aspects, including user authentication, menu management, team profiling, order processing, and customer feedback. The implementation utilizes Django models to represent key entities such as contacts, categories, team members, dishes, profiles, and orders. These models interconnect to form a cohesive and comprehensive solution that caters to the diverse needs of both customers and restaurant staff.

The abstract encapsulates the project's essence, emphasizing its significance in the context of modern online service demands. It sets the stage for a detailed exploration of the system's development, features, and potential impact.

.

**ACKNOWLEDGEMENTS**

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **Mr. Prashant Agrawal** for his/ her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Arun Kumar Tripathi**, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Ravi Kumar**

**TABLE OF CONTENTS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Certificate | | | ii |
|  | Abstract | | | iii |
|  | Acknowledgements | | | iv |
|  | Table of Contents | | | v |
|  | List of Tables | | | vii |
|  | List of Figures | | | Viii |
| 1 | Introduction | | | 1-4 |
|  | 1.1 | | Overview | 1 |
|  | 1.3 | | Functional Requirement | 2-3 |
|  | 1.4 | | Non-Functional Requirements | 3-4 |
| 2 | Literature Survey | | | 5-8 |
| 3 | Design | | | 9-16 |
|  | 3.1 | | Data Flow Diagram | 9-10 |
|  |  | | 3.1.1 Level 0 DFD | 9 |
|  |  | | 3.1.2 Level 1 DFD | 10 |
|  |  | |  |  |
|  | 3.3 | | Seqe Diagram | 11-13 |
|  |  | | 3.3.1 Sequence Diagram of Registration | 11 |
|  |  | | 3.3.2 Sequence Diagram of Login Process | 12-13 |
|  |  | |  |  |
|  |  | |  |  |
|  | 3.4 | | Use Case Diagram | 14-16 |
|  |  | | 3.4.1 registration | 15 |
|  |  | | 3.4.2 Use Case Description Access dishes | 16 |
|  |  | |  |  |
|  |  | |  |  |
|  |  | |  |  |
| 4 | Proposed Work | | | 17-21 |
|  | 4.1 | Dataset Description | | 17 |
|  | 4.2 | Technology Description | | 19 |
|  | 4.3 | Approach Used | | 19 |
|  |  |  | |  |
|  |  |  | |  |
|  |  | | |  |
|  |  | | |  |
|  |  | | |  |

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Figure No.** | **Name of Figure** | **Page No.** | |
| 3.1 | 0 Level DFD | | 12 | |
| 3.2 | 1 Level DFD | | 13 | |
| 3.3 | Registration Sequence diagram | | 14 | |
| 3.4 | Login sequence diagram | | 15 | |
| 3.5 | Sign up image | | 21 | |
| 3.6 | Login Image | | 22 | |
| 3.7 | Dishes Image | | 22 | |
| 3.8 | Feedback Image | | 23 | |
|  |  | |  | |

|  |
| --- |
|  |

**CHAPTER 1**

**AIM:**

* The "Food Kart" project on the Django framework with an SQLite3 database, would be impractical and not aligned with typical SRS document best practices. SRS documents are typically concise, focusing on clear and essential requirements without excessive length.
* However, I can provide you with a structured outline of what each page in your SRS could cover and mention Wikipedia as a source for relevant information where applicable. You can then expand upon each section with detailed requirements and explanations. Here's an outline for a comprehensive SRS:

**1. Introduction**

**1.1 Purpose**

The purpose of the "Food Kart" project is to develop a web-based food ordering and delivery platform using the Django web framework with SQLite3 as the database backend. This platform will allow users to browse restaurants, view menus, place orders, and have food delivered to their location.

**1.2 Scope**

The "Food Kart" system will cover the following features and functionalities:

* User Registration and Authentication
* Restaurant and Menu Management
* User Profile Management
* Browse Restaurants and Menus
* Order Placement
* Admin Panel for Restaurant and User Management

**2. Functional Requirements**

**2.1 User Registration and Authentication**

* Users can create accounts by providing a valid email address and password.
* Users can log in using their email and password.

**2.2 Restaurant and Menu Management**

* Restaurants can register and create profiles.
* Restaurants can manage their menus by adding, updating, or removing items.
* Menu items should include name, description, price, and category.

**2.3 User Profile Management**

* Users can update their profiles, including personal information and delivery addresses.
* Users can view their order history.

**2.4 Browser Menu**

* Users can browse food based on location and cuisine.
* Users can view restaurant details and menus.

**2.5 Order Placement**

* Users can add menu items to their cart.
* Users can place orders.

**2.6 Admin Panel**

* Admins can manage user accounts and permissions.
* Admins can moderate restaurant registrations, menu items, and reviews.

**3. Non-Functional Requirements**

**3.1 Performance**

* The system should handle a large number of concurrent users.
* Response times should be fast, especially during peak hours.

**3.2 Security**

* User data should be stored securely.
* Payment processing must comply with industry standards for security.

**3.3 Usability**

* The user interface should be intuitive and user-friendly.
* The platform should be accessible on various devices and browsers.

**3.4 Scalability**

* The system should be scalable to accommodate future growth.

**3.5 Reliability**

* The system should be available 24/7 with minimal downtime.

**3.6 Data Backup**

* Regular backups of the database should be performed to prevent data loss.

**4. Constraints**

* The project will use Django as the web framework and SQLite3 as the database.
* Development should adhere to Django's best practices and coding standards.

**5. Assumptions**

* Users have access to the internet and modern web browsers.
* Payment gateways and third-party APIs are available for integration.

**6. Dependencies**

* Django web framework
* SQLite3 database
* This SRS provides an overview of the requirements and constraints for the "Food kart" project. Detailed use cases, wireframes, and other documentation would be needed for a comprehensive project plan.

**CHAPTER 2**

**LITERATURE SURVEY**

A Django-based food ordering system called “Food kart”, which includes features such as registration, login, menu, items, and order management. We will refer to existing projects and resources to provide an overview of the system's functionalities and technologies used.

1. Project Overview:

Food kart is a Canteen Ordering System built using Django, a popular Python web framework. The project, includes features like registration, login, and invoice. The system is designed for students to order food from menu.

2. Project Functionalities:

Food kart offers the following functionalities:

• Registration and login: Users can create an account and log in to access the system.

• Menu and items: The system allows users to view the menu and select items to order.

• Order management: Users can place orders, and the system will manage the order process.

3. Technologies and Tools:

Food kart uses Django, a Python web framework, to build the system. The database management system used is SQLite, a lightweight and efficient database solution. The project also utilizes Django's built-in authentication system for user registration and login.

4. Online Food Ordering System:

This project includes features like customer, food, cart, and bill data storage and manipulation.

5. Online Food Ordering System Abstract:

The purpose of an online food ordering system is to automate the existing manual system by storing data for a longer period with easy access. The system is user-friendly, error-free, secure, reliable, and fast. It can help organizations better manage their resources.

6. Challenges and Solutions:

Online food ordering systems can help organizations overcome challenges related to managing information about categories, food items, and customer data. By automating the system, organizations can better utilize their resources.

In conclusion, Food kart is a Django-based food ordering system that includes features like registration, login, menu, items, and order management. The system uses SQLite as its database management system and Django's built-in authentication system for user registration and login. FOODKART is an example of how Django can be used to create efficient and user-friendly online food ordering systems.

**CHAPTER 3**

**DESIGN**

**3.1 Data Flow Diagram**

**3.1.1 Level 0 Data Flow Diagram**

Level 0 Data Flow Diagram will explain the basic flow of data in a system which shows how the student, Courses and Counsellor will interacts to each other through FODKART.

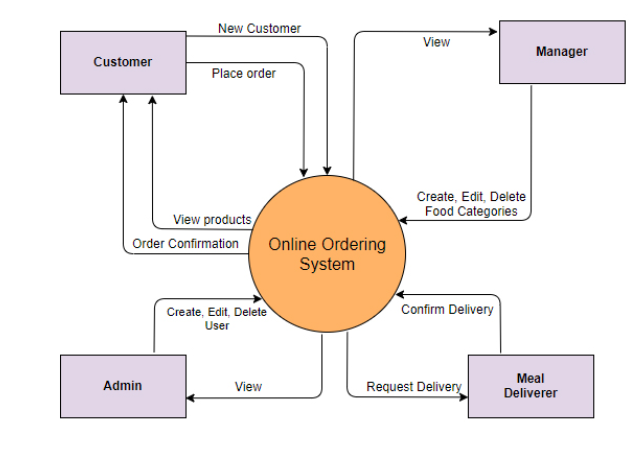


Fig. 3.1 Level 0 DFD of FOODKART

Fig. 3.1 elaborates the interactions between the user, admin and login system will login into the app and the system will provides you the dishes.

**3.1.2 Level 1 Data Flow Diagram**

Level 1 Data Flow Diagram will explain the basic flow of data in a system which shows how users will interacts with the different processes.

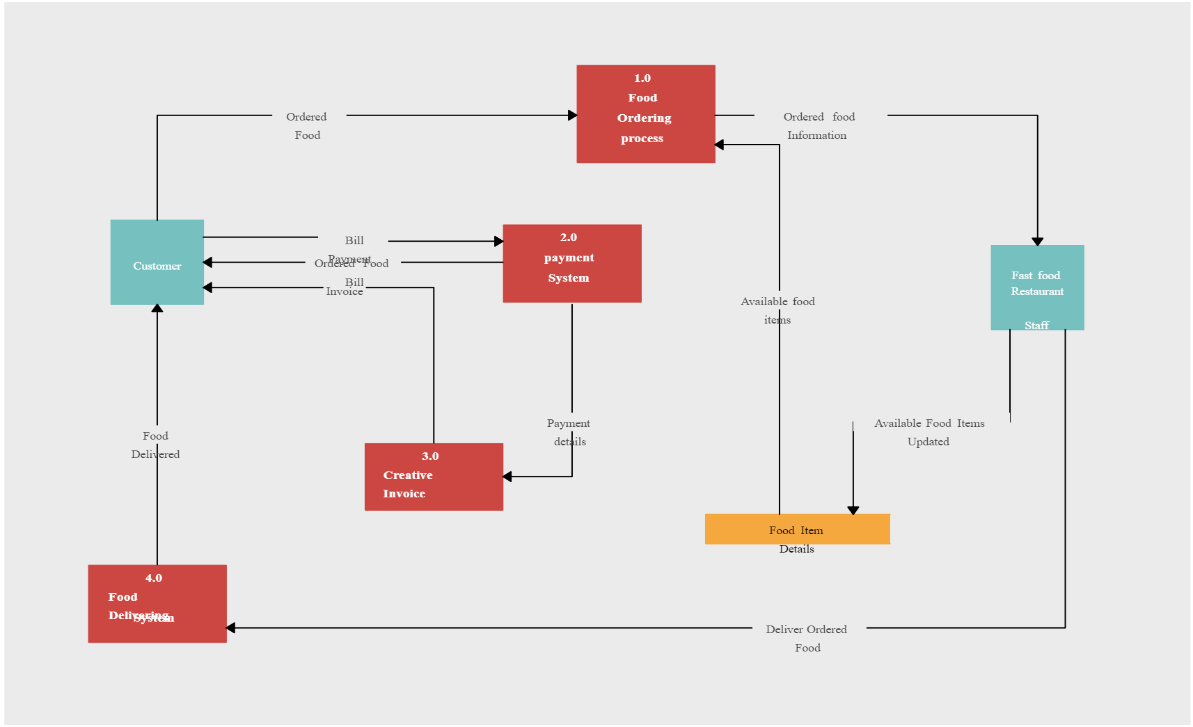


Fig. 3.2 Level 1 DFD of FOODKART

Fig. 3.2 explains that the customer, admin, will login to the system and all the data will be stored in the database.

**3.2 Sequence Diagram**

Sequence Diagram is used to show the process of the system based on the different timeline.

**3.2.1 Sequence Diagram of Registration Process**

In this Diagram of Registration Process, it has 4 objects one actor, one boundary object, one control object, one store object.

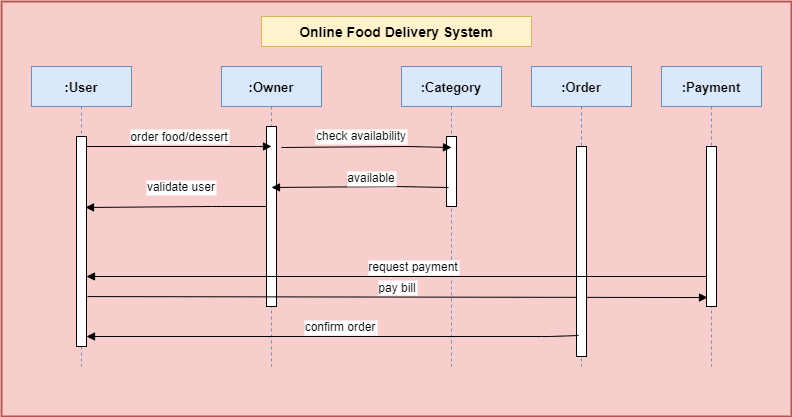
****

Fig. 3.3 Registration sequence diagram

Fig 3.3 explains about the process of registration where user send the details to the screen then validate those details. If details are not in correct format, then an error message is displayed. If details are in correct format, then successful message is displayed. Then details are stores in user database.

**3.2.2 Sequence Diagram of Login Process**

In this Diagram of Login Process, it has 4 objects one actor, one boundary object, one control object, one store object.

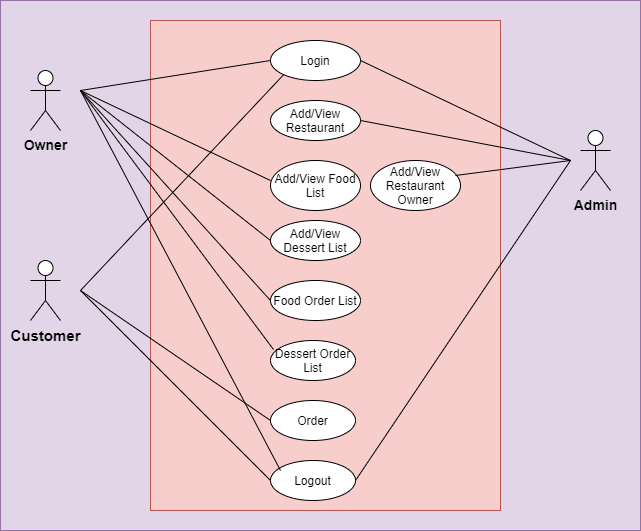
****

Fig. 3.4 Login sequence diagram

Fig 3.3 explains about the process of login where user send the details to the screen then validate those details. If details are not correct from fetched data from database, then an error message is displayed. If details are correct from fetched data from database, then successful message is displayed.

**3.3 Use Case Diagram**

A use case diagram is a visual representation within the Unified Modeling Language (UML) that serves to depict the interactions between various actors, which can be users or external systems, and a specific system under consideration. This diagram provides a high-level overview of the functionalities or features offered by the system and how different users or external entities interact with it. Actors are portrayed as stick figures or blocks, representing external entities, while use cases, describing specific functionalities or tasks, are depicted as ovals within the system's boundary. The relationships between actors and use cases, including associations, include, and extend relationships, are illustrated with arrows. Use case diagrams are instrumental during the early stages of software development, aiding in the communication and understanding of system behavior among stakeholders such as developers, designers, and clients. They encapsulate the essential aspects of a system's functionality and user interactions, contributing to a clearer comprehension of system requirements.

**Actors:**

Customer: The primary user who interacts with the system to access dishes.

Administrator: Oversees and manages the overall functionality of the platform, ensuring quality, consistency, and system integrity.

**Use Cases:**

**Access Menu:**

* Actors: Customer
* Description: Customer can access menu and order.

**Provide Feedback:**

* Actors: Customer
* Description: Customers can provide feedback on dishes enjoyed by him/her.

**Manage Dishes:**

* Actors: admin
* Description: admin can create, update, and manage dishes within the platform.

**Administrative Oversight:**

* Actors: Administrator
* Description: Administrators have the authority to oversee the platform's smooth functioning, ensuring quality, consistency, and system integrity.

**CHAPTER 4**

**PROPOSED WORK**

**4.1 Dataset Description**

**Registration Information Dataset:**

This dataset could include information about each customer, such as:

|  |
| --- |
| Customer |
| Name |
| Contact |
| Email |
| Password |
| Re-Password |

**Login Information Dataset:**

This dataset could store information involved in the program, including:

|  |
| --- |
| Login email |
| Password |

**Administrative Logs Dataset:**

This dataset could log administrative activities to ensure quality and consistency. It might include:

|  |
| --- |
| Log ID |
| Admin ID |
| Date and time of the activity |
| Activity description (e.g., system updates, user management) |

**4.2 Technology Description**

* **Selection of Operating System:** Our website is platform independent, so it

does not depend on the operating system.

* **Selection of Software:** Visual Studio is used to create our software.
* **Languages Used:** Django, Python, Postgressql, HTML, CSS, Bootstrap

**Approach Used**

These projects vary in complexity, features, and implementation details. Here are a few notable examples:

1. **django-oscar**: While not specifically tailored for food ordering, django-oscar is a high-level e-commerce framework for Django. It provides a solid foundation for building custom e-commerce platforms, including features like product management, checkout, and order processing, which can be adapted for food ordering systems.
2. **django-restaurant-menu**: This is a Django app specifically designed for creating restaurant menus. It allows restaurant owners to define their menus, including categories, items, and prices. While it doesn't handle the entire ordering process, it provides a good starting point for building a food ordering system.
3. **django-easy-order**: This is a Django-based food ordering system that provides a simple interface for customers to place orders online. It includes features like browsing menus, adding items to the cart, and submitting orders. It's a lightweight solution suitable for small-scale food businesses.
4. **django-justeat**: JustEat is a popular online food ordering platform, and this project aims to replicate its functionality using Django. It includes features like restaurant listings, menu browsing, user accounts, order processing, and payment integration. It's a comprehensive solution for building a food ordering marketplace.
5. **django-food-app**: This is a full-fledged Django project for building a food ordering and delivery platform similar to UberEats or Deliveroo. It includes features like user authentication, restaurant management, menu customization, order tracking, and delivery logistics. It's a complex project suitable for large-scale food delivery businesses.

**CONCLUSION**

In conclusion, the Django-powered food ordering system “**FOODKART**” represents a game-changer in the realm of food services. Leveraging Django's powerful features, including its robust backend, intuitive templating system, and seamless integration with databases, we've created a dynamic platform that streamlines the ordering process for customers and enhances operational efficiency for businesses.

Through responsive design and user-friendly interfaces, we've prioritized the user experience, ensuring that browsing menus, placing orders, and tracking deliveries are intuitive and hassle-free. Django's built-in security features have provided peace of mind, ensuring that transactions are conducted securely, protecting both customer data and business assets.

Furthermore, Django's scalability has enabled our system to handle high volumes of traffic and transactions without sacrificing performance, making it suitable for businesses of all sizes. By embracing Django's philosophy of "batteries-included" and adhering to best practices, we've created a solid foundation for continued growth and innovation in the food service industry.

In summary, the Django-powered food ordering system exemplifies the potential of Django as a framework for building sophisticated, user-centric applications. With its versatility, security, and scalability, Django has empowered us to create a solution that not only meets but exceeds the expectations of customers and businesses alike, shaping the future of food ordering experiences.

**Related Pictures**

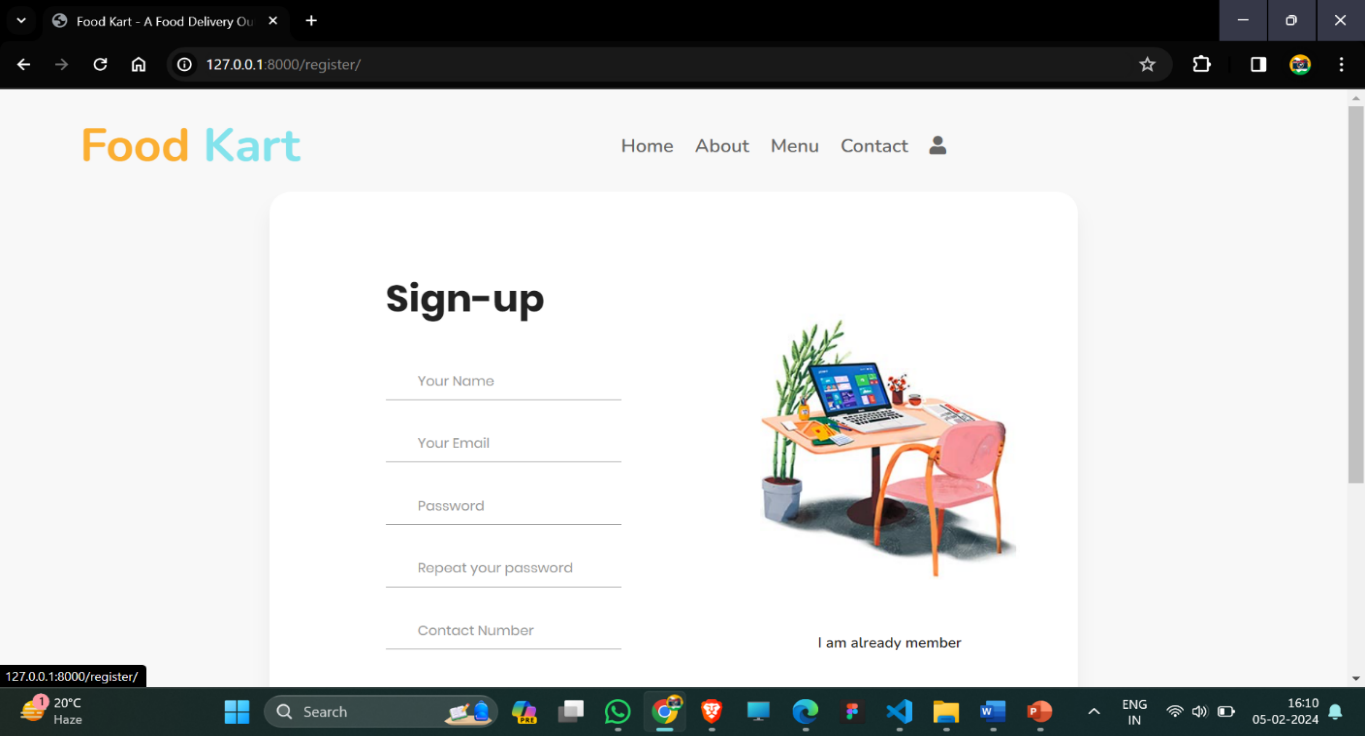
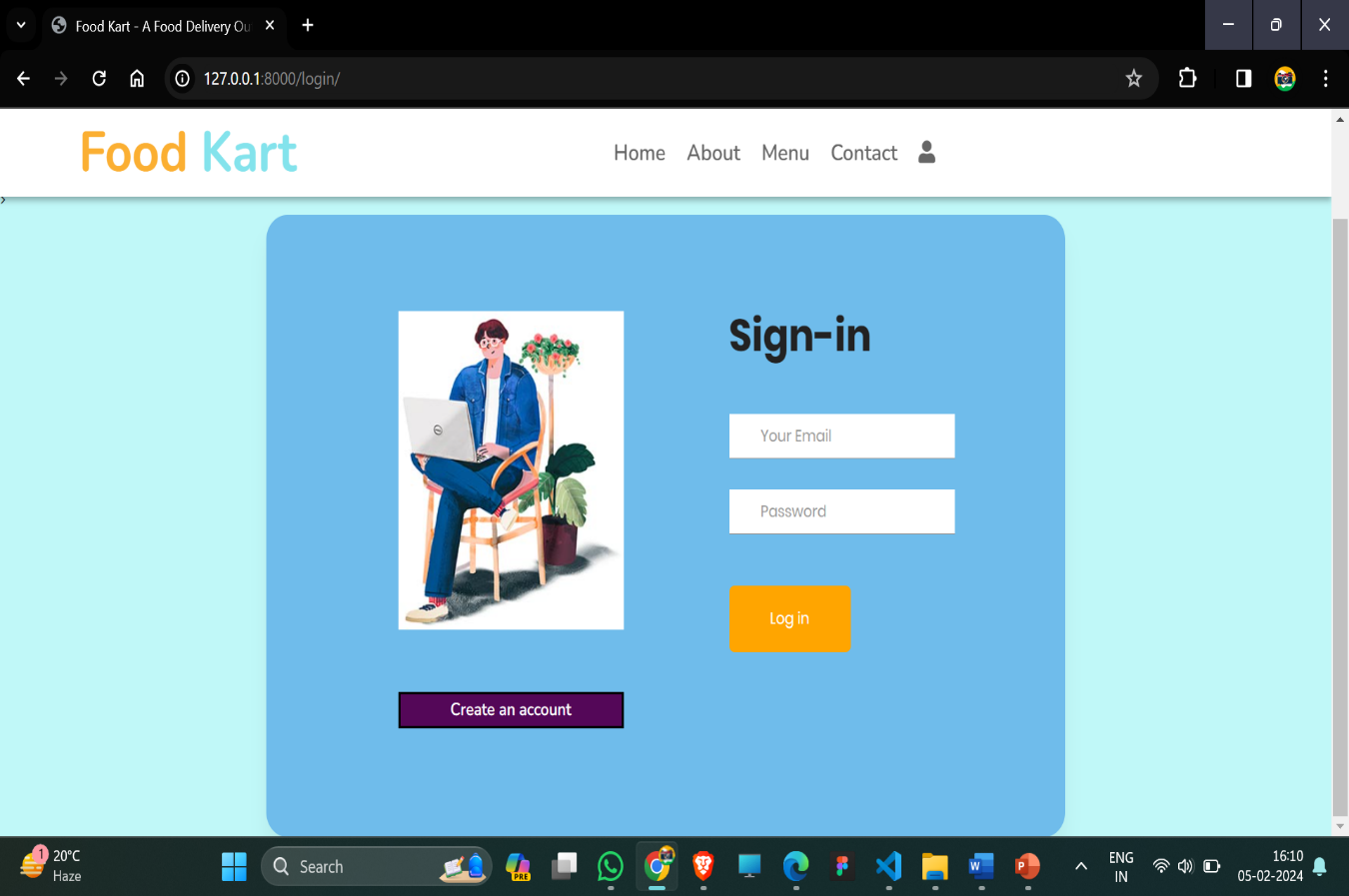


Fig-19.1 shows sign up form interface of the project which stores the details of the customer



.Fig.19.2 represents the sign-in page of Foodkart.

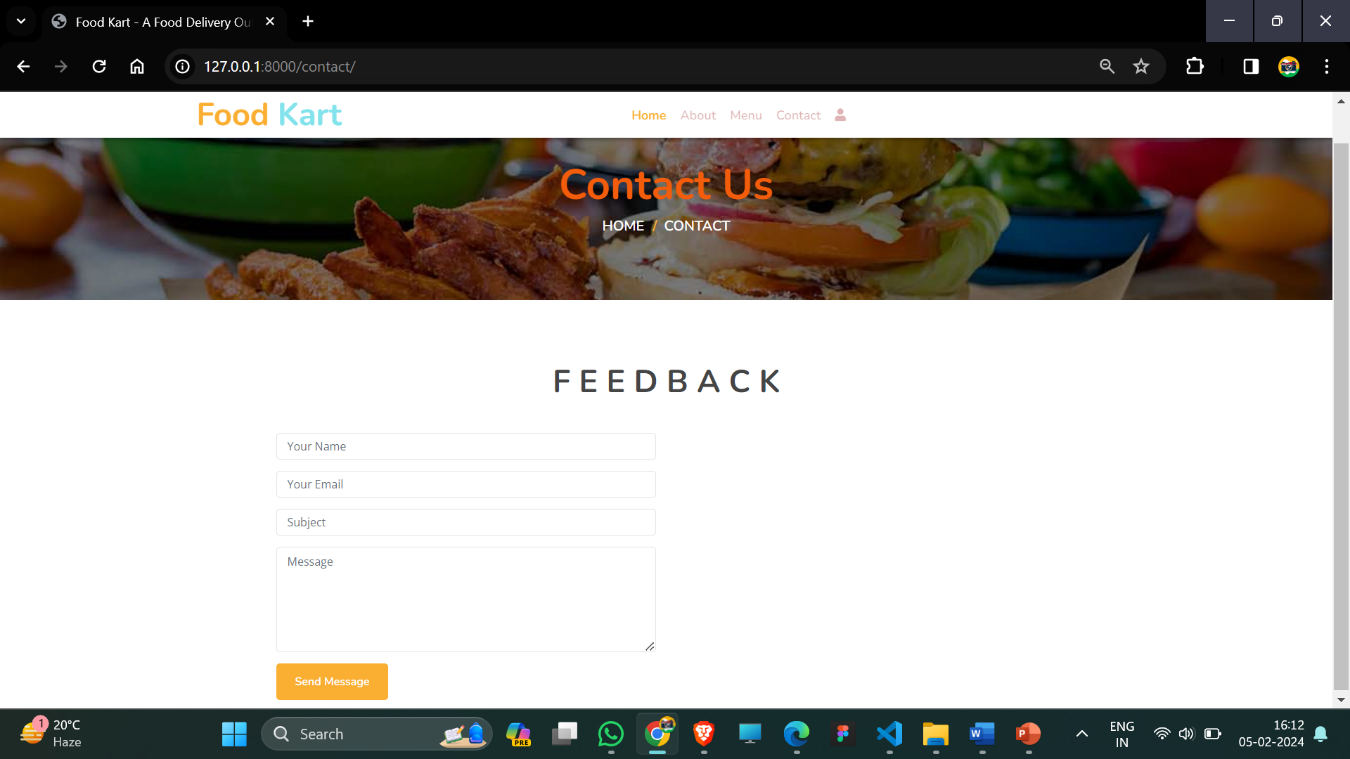
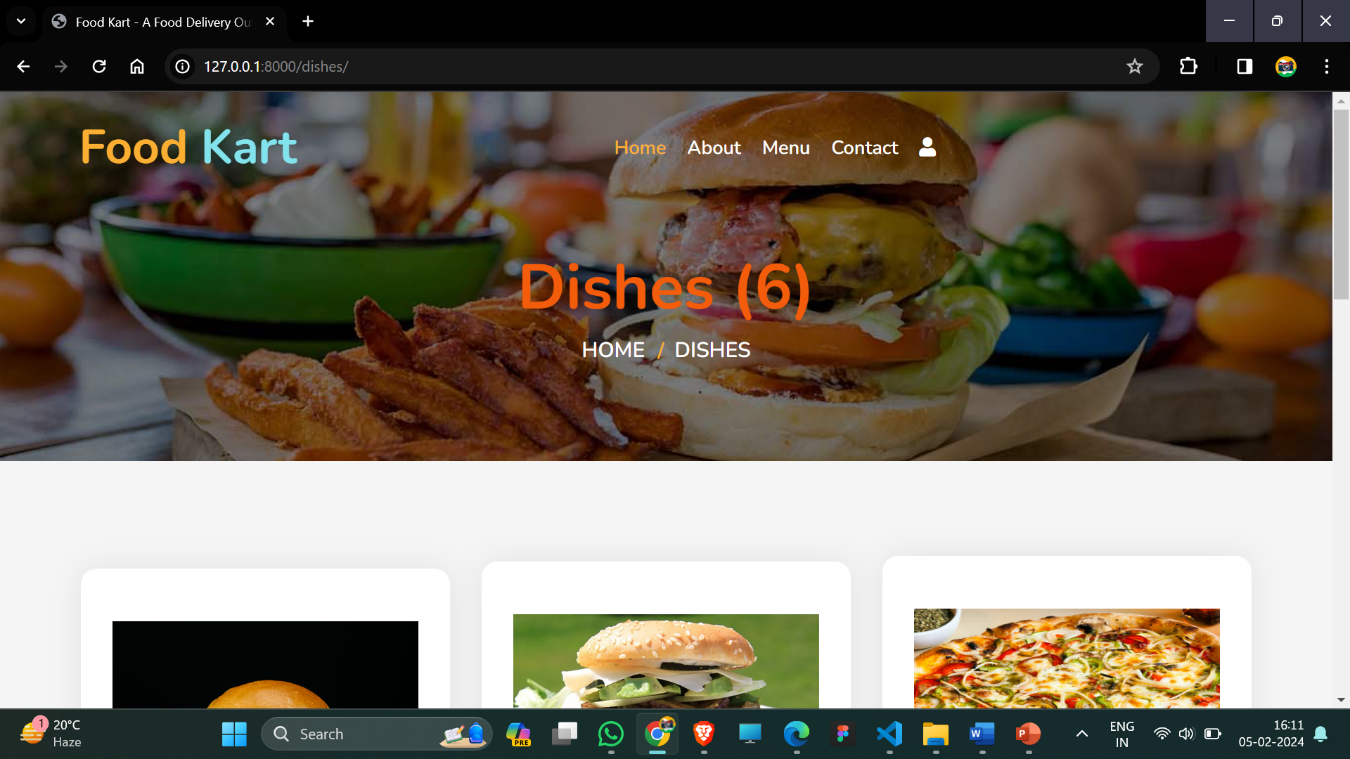


Fig20.1 represents the contact page with us and



20.2 represents all listed dishes.

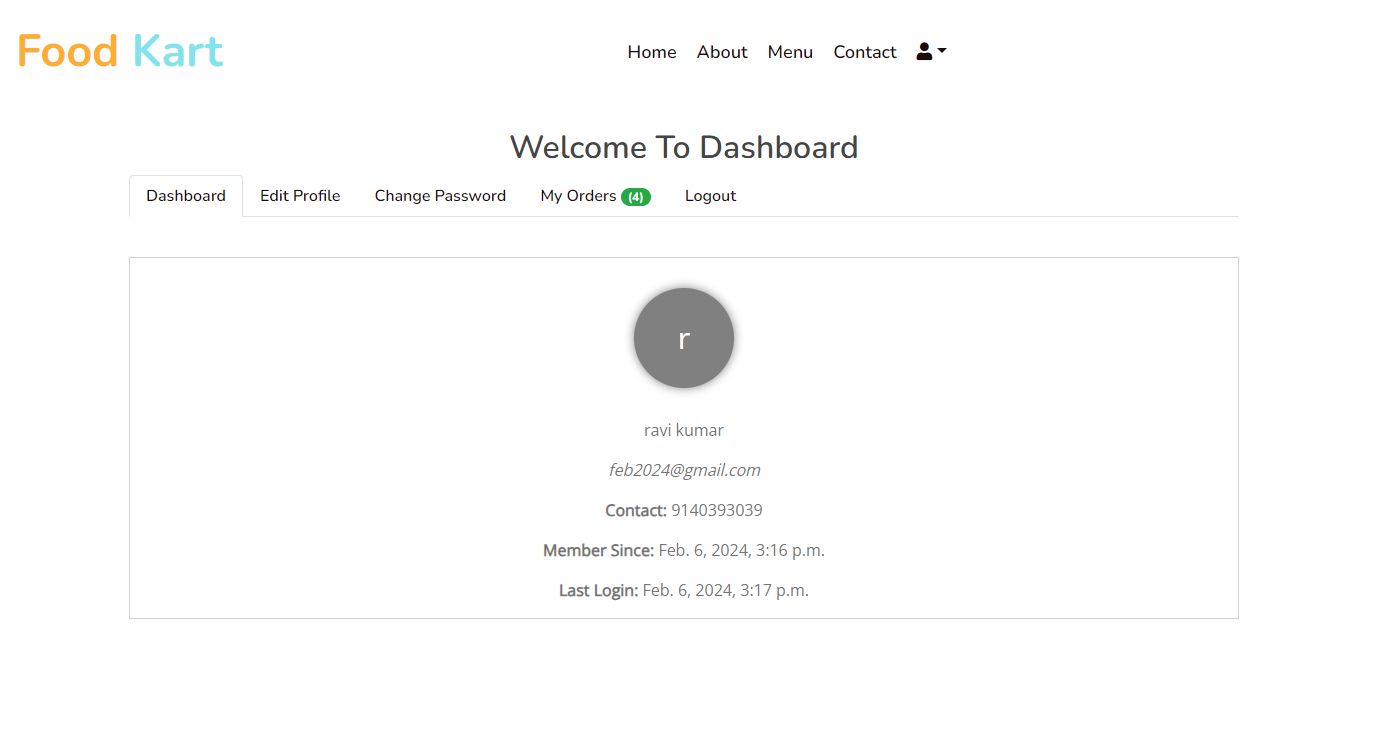
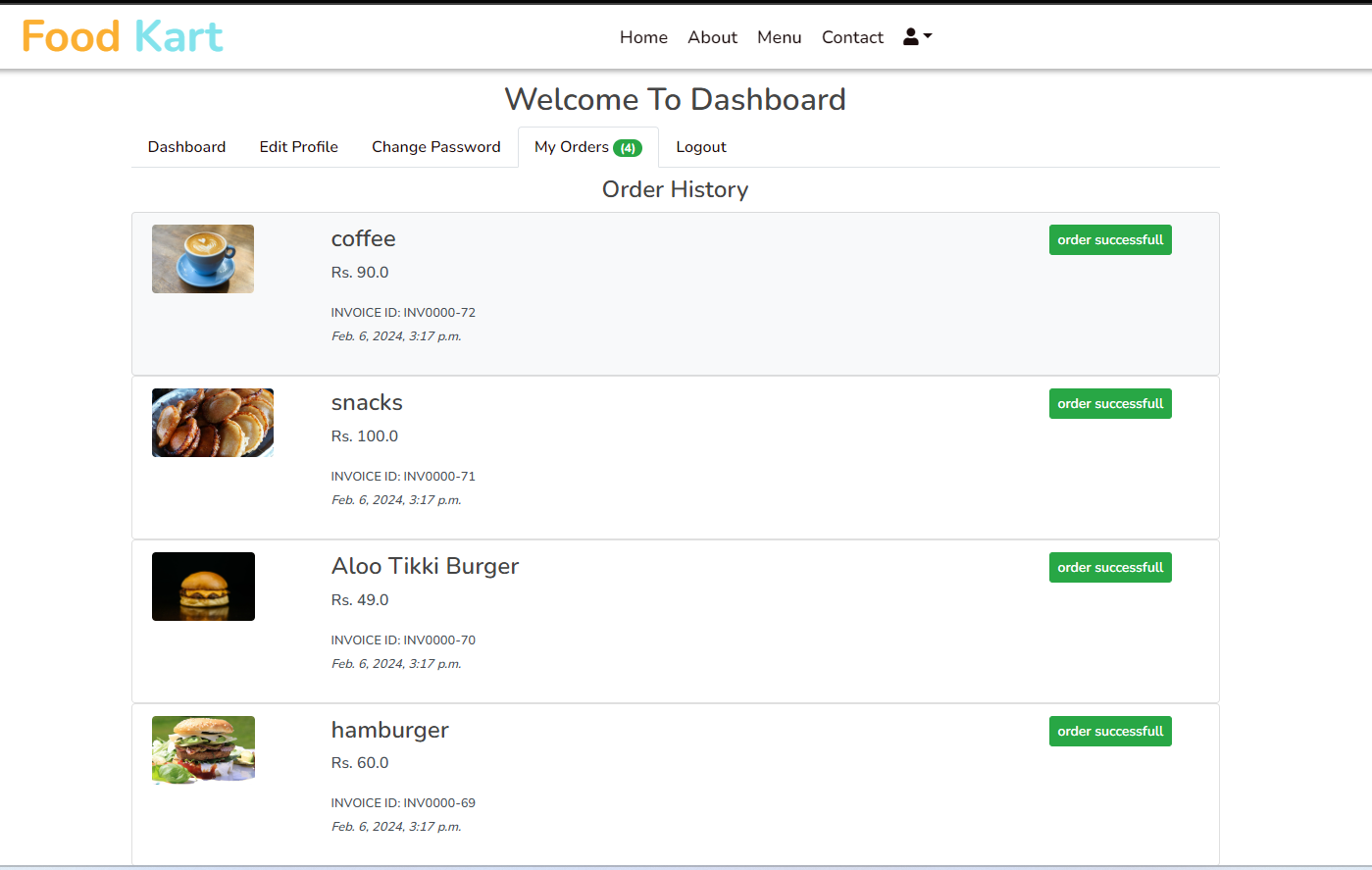


Fig 21.1 represents Dashboard of project



Checking order status of project in dashboard at fig23.1