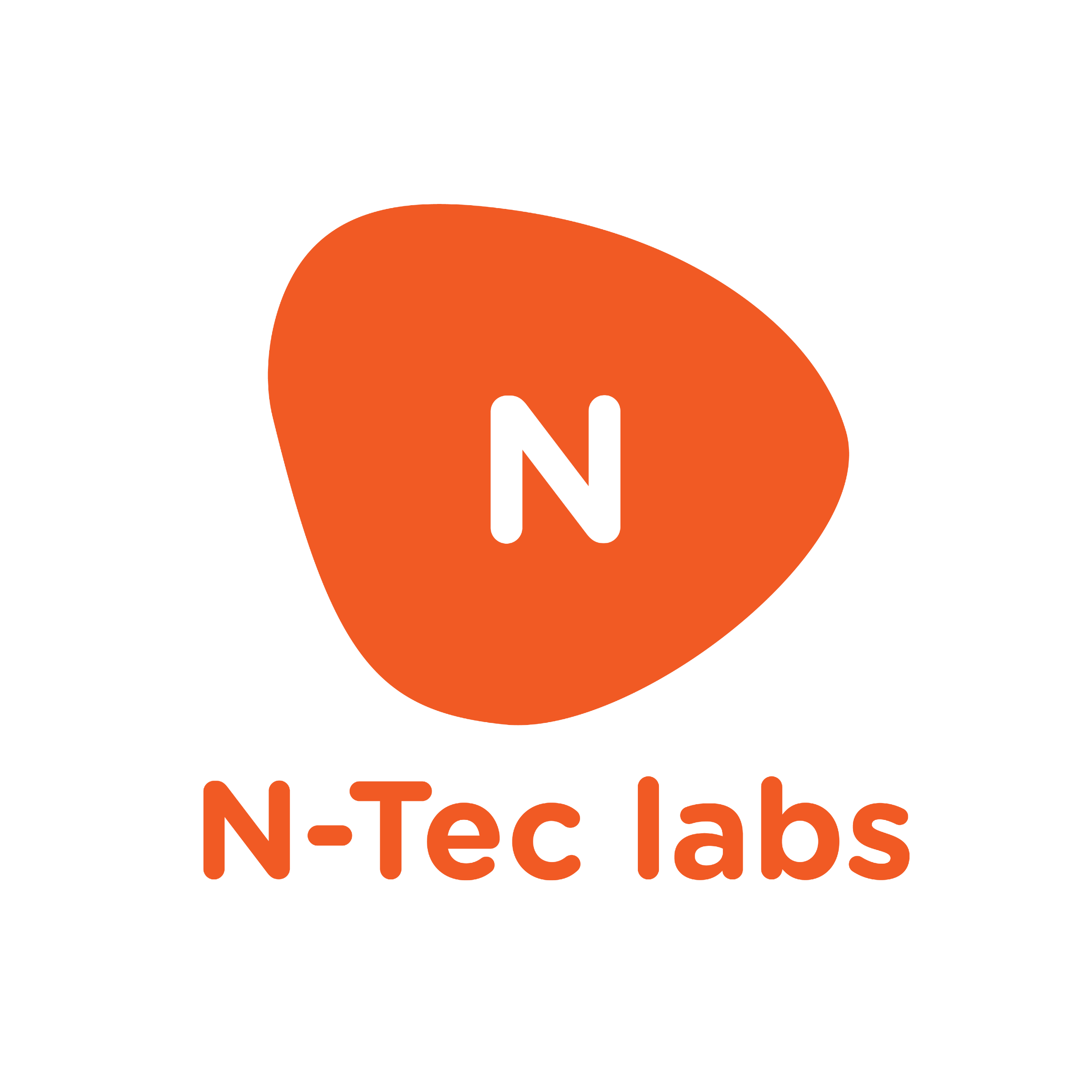
**REPUBLIQUE DU CAMEROUN**

Paix-Travail-Patrie

**REPUBLIC OF CAMEROON**

Peace-Work-Fatherland



A logo with a map and a globe

Description automatically generated

**N-TEC LABS**

PO BOX: Grand Chefferie Simbock, Immeuble DC

Tel: (+237)651834545

Website: www.nteclabs.com

Email: contact@nteclabs.com

**AFRICAN INSTITUTE OF COMPUTER SCIENCES FOR PAUL BIYA TECHNOLOGICAL CENTER OF EXCELLENCE CAMEROON**

PO Box: 13719 Yaoundé

Tel: (+237)242729957 / (+237)242729958

Website: [www.iaicameroun.com](http://www.iaicameroun.com)

Email: contact@iaicameroon.com

Email: contact@iaicameroun.com

INTERNSHIP REPORT

**RESTAURANT PRE-ORDER AND ORDER SYSTEM**

**CASE STUDY: SNAP BURGER CRADAT YAOUNDE**

Internship Carried out from the 01st July to 30th September 2023

In view of obtaining a **Higher Technician Diploma (HTD)** in computer sciences option

**Software Engineering**

Submitted by:

**NDIP LUCY-DIANE BANYI**

**Level 2 Student at AICS Cameroon**

Supervisors:

**Academic Supervisor**

**Mr. NGUH PRINCE**

**Lecturer at AICS-Cameroon**

**Professional Supervisor**

**Miss NFORMI VICTORY**

**Developer at N-tech labs**

**ACADEMIC YEAR 2023-2024**

**DEDICATION**

**THANKS TO MY PARENTS AND FAMILY**

FOR THEIR LOVE, ADVICE AND UNCONDITIONAL SUPPORT

**CONTENT**

**DEDICATION ........................................................................................................................... i**

**ACKNOWLEDGEMENT ....................................................................................................... ii**

**CONTENT ............................................................................................................................... iii**

**LIST OF FIGURES .................................................................................................................. v**

**LIST OF TABLES ................................................................................................................. viii**

**GLOSSARY ............................................................................................................................. ix**

**ABSTRACT .............................................................................................................................. x**

**RESUME .................................................................................................................................. xi**

**GENERAL INTRODUCTION ............................................................................................... 1**

**PART I ....................................................................................................................................... 2**

**INSERTION PHASE ............................................................................................................... 2**

**PART II ................................................................................................................................... 13**

**EXISTING SYSTEM ............................................................................................................. 13**

**PART III .................................................................................................................................. 23**

**SPECIFICATION BOOK ..................................................................................................... 23**

**PART IV .................................................................................................................................. 40**

**ANALYSIS BOOK ................................................................................................................. 40**

**PART V .................................................................................................................................... 76**

**CONCEPTION PHASE ........................................................................................................ 76**

**PART VI .................................................................................................................................. 96**

**REALIZATION PHASE ....................................................................................................... 96**

**PART VII............................................................................................................................... 107**

**TEST OF FUNCTIONALITIES ......................................................................................... 107**

**PART VIII ............................................................................................................................. 115**

**INSTALLATION GUIDE AND USER GUIDE ................................................................ 115**

**GENERAL CONCLUSION ................................................................................................ 135**

**ANNEXES ............................................................................................................................... xii**

**BIBLIOGRAPHY ................................................................................................................. xiii**

**WEBOGRAPHY ................................................................................................................... xiv**

**TABLE OF CONTENT ......................................................................................................... xv**

**ACKNOWLEDGEMENTS**

This work would not have been accomplished without the help, encouragements, and advice, teachings and precious time of the following people:

* We give thanks to God who made me able to this project.
* The Resident Representative of African Institute of Computer Sciences Yaounde Cameroon, **Mr. ARMAND CLAUDE ABANDA.**
* **Mr. NGUH PRINCE** our academic supervisor and lecturer at AICS-Cameroon for his availability, advice and moral support throughout the internship period which helped in the realization of this project.
* The CEO of N-Tech Labs Mr. **NGWANYI JOEL** for having welcomed us and validate our application.
* **Miss FONGWA BLESSING** for having welcomed us, encouragements, availability, support, supervision, counselling and motivation throughout the internship period.
* **Miss** **NFORMI VICTORY** our professional supervisor, for her warm welcome, encouragements, availability, support, supervision, counselling and motivation throughout the internship period.
* To all the teachers of the African Institute of Computer Sciences Yaounde Cameroon for their help and support throughout our training.
* To my parents and family for their guidance and support and for have done the required sacrifice to have made me integrate in this prestigious and talented institution
* To all friends and classmates at AICS for their knowledge, ideas, collaboration and most importantly the unforgettable moments spent together.

**LIST OF ABBREVIATIONS**

* **2TUP:** Two Track Unified Process
* **AICS:** African Institute of Computer Sciences
* **API:** Application Programming Interface
* **CSS:** Cascading Style Sheet
* **DBMS:** Database Management System
* **ER:** Entity Relationship
* **ERD:** Entity Relationship Diagram
* **HTML:** Hyper Text Markup Language
* **HTTP:** Hypertext Transfer Protocol
* **PDM:** Physical Data Model
* **SQL:** Structured Query Language
* **Mr.:** Mister
* **Mrs.:** Miss

**LIST OF FIGURES**

**LIST OF TABLES**

**ABSTRACT**

Restaurants play a vital role in society by contributing to culture, the economy, and social interactions. To enhance the efficiency of their operations and customer experience, it's essential for restaurants to adopt **modern technology solutions**. Many restaurants still rely on manual processes for order-taking and service management, which leads to common challenges such as order errors, delays, slow service, cash handling issues, and difficulties in tracking sales trends and customer preferences.

In response to these challenges, we propose the **design and development of a Pre-Order and Order Restaurant System**. This system enables customers to pre-order and order meals for dine-in, significantly reducing wait times and ensuring smoother service flow. To build this system, we conducted extensive studies and interviews with key stakeholders (customers, servers, kitchen staff, and managers) to understand their needs.

The application development follows the **2TUP methodology**, ensuring an organized approach to design and implement. Using **UML (Unified Modelling Language)**, we formalized data interaction, ensuring a clear representation of the system's workflow. Data storage and management are handled through the **MySQL Relational Database Management System (RDBMS)**, allowing for secure and efficient data handling. The back-end implementation using **PHP** guarantees a flexible, secure, and user-friendly platform for enhancing smooth streamline ordering.

By adopting this **Pre-Order and Order Restaurant System**, restaurants can **streamline their operations, improve customer service speed, reduce order errors**, and provide an enhanced customer experience. The system also offers valuable data insights, enabling restaurants to better understand customer preferences and make data-driven decisions that foster growth and profitability.

**RESUME**

Les restaurants jouent un rôle essentiel dans la société en contribuant à la culture, à l'économie et aux interactions sociales. Pour améliorer l'efficacité de leurs opérations et l'expérience client, il est crucial pour les restaurants d'adopter des solutions technologiques modernes. De nombreux restaurants dépendent encore de processus manuels pour la prise de commandes et la gestion des services, ce qui entraîne des défis courants tels que des erreurs de commande, des retards, un service lent, des problèmes de gestion des espèces et des difficultés à suivre les tendances de vente et les préférences des clients.

En réponse à ces défis, nous proposons la conception et le développement d'un **Système de Précommande pour Restaurants**. Ce système permet aux clients de précommander leurs repas pour une collecte ou une consommation sur place, réduisant ainsi considérablement les temps d'attente et assurant un flux de service plus fluide. Pour concevoir ce système, nous avons mené des études approfondies et des entretiens avec les parties prenantes clés (clients, serveurs, personnel de cuisine et gestionnaires) afin de comprendre leurs besoins.

Le développement de l'application suit la méthodologie **2TUP**, garantissant une approche structurée de la conception et de la mise en œuvre. En utilisant **UML (Unified Modelling Language)**, nous avons formalisé les interactions de données, assurant une représentation claire du flux de travail du système. La gestion et le stockage des données sont pris en charge par le **Système de Gestion de Base de Données Relationnelles MySQL (RDBMS)**, permettant une gestion sécurisée et efficace des données. L'implémentation du back-end en **PHP** assure une plateforme flexible, sécurisée et conviviale pour la gestion des services du restaurant.

En adoptant ce **Système de Précommande**, les restaurants peuvent optimiser leurs opérations, améliorer la rapidité du service, réduire les erreurs de commande et offrir une meilleure expérience client. Le système fournit également des informations précieuses sur les données, permettant aux restaurants de mieux comprendre les préférences des clients et de prendre des décisions basées sur les données pour favoriser la croissance et la rentabilité.

**GENERAL INTRODUCTION**

Technology is advancing at an incredible rate, impacting various sectors, from business to agriculture and beyond. In this fast-paced, increasingly digitalized society, life is becoming more convenient. In Cameroon, many businesses are striving to leverage the digital economy to foster growth and boost the national economy. To achieve this, enterprises require qualified personnel with skills in computer sciences and related fields. As part of our academic curriculum, second-year students in the **AICS Cameroon** are required to undertake a three-month internship to apply the knowledge acquired in school to real-world professional settings.

For this reason, we applied for an internship at **N-TEC LABS**, a company specializing in IT solutions. During our internship, we were assigned the theme **“DEVELOPMENT OF A PRE-ORDER AND ORDER RESTAURANT SYSTEM.”** This report is structured into eight parts:

1. **Insertion Document**: This section introduces the company where we completed our internship and describes the integration process for interns.
2. **Existing System**: Here, we will outline the current ordering system in place within the restaurant industry, detailing its strengths and weaknesses.
3. **Specification Book**: In this section, we will define the user needs for the pre-order system, considering project timelines and costs.
4. **Analysis Document**: This part presents the chosen analysis methods, along with all relevant diagrams used to analyze the project.
5. **Conception Phase**: This section details both the generic and specific design of the pre-order system to reflect real-world requirements.
6. **Realization Phase**: This phase visualizes the implementation process of the proposed solution, highlighting key milestones and challenges.
7. **Test of Functionalities**: In this section, we will demonstrate the various functionalities of our application and how they operate to meet user needs.
8. **User Guide**: This part provides comprehensive instructions on how to use the pre-order system and outlines the necessary conditions for its effective operation.

# **PART I:**

# **INSERTION PHASE**

Preamble

The insertion phase is a part of t he internship report where we will present the detailed structure and characteristics of our enterprise where our internship was carried out.

Content

INTRODUCTION

1. WELCOME AND INTEGRATION INTO N-TEC LABS
2. GENERAL PRESENTATION OF N-TEC LABS
3. ORGANISATION OF THE COMPANY
4. HARDWARE AND SOFTWARE RESOURCES OF THE COMPANY
5. BRIEF PRESENTATION OF THE PROJECT THEME

CONCLUSION

**INTRODUCTION**

The insertion phase is a period (generally of 02 weeks) reserved for the different interns to discover and to familiarize themselves with the working environment. Here, we got to know about the staff, the different hardware and software resources used, the different departments which constitute the enterprise, how the company functions both internally and externally and we were introduced to our work space. During this period, we were also attributed an internship master often called professional supervisor and a theme. We also had a time to discuss amongst us interns on topics like what we love doing most, what we dislike, our beliefs and experiences.

1. **WELCOME AND INTEGRATRION**
2. **Welcome**

We arrived at N-TEC LABS on Monday July 2024 at 8:00am. we were welcomed, by **Mme Fongwa Blessing** the enterprise’s main Secretary, who introduced us to our work space and her collaborators. She then introduced us to the boss **Mr. Ngwanyi Joel** and our supervisor **Mme Nformi Victory.** Later on, we were logged onto the Intern portal and we were given a task to test our competence

1. **Integration**

A working day at N-TEC LABS start from 08:00am to 06:00pm. Our tasks in the company generally follow a predefined routine. Every day we were assigned tasks to carry out. There was also an opportunity to present our difficulties and challenges to everyone and receive directives.

1. **General Presentation of the company**
2. **Geographical location**

*Figure1: Location of N-tec Labs*



1. Company’s Identification Form

*Table 1: Company Identification form*

|  |  |
| --- | --- |
| IDENTIFICATION FORM | |
| Company name | N-TEC LABS |
| Type of Enterprise | Institution, Digital Agency |
| President of the Company | Mr. NWANYI Joel |
| Office Telephone | (+237)651834545 |
| Website | nteclabs.com |
| Language | English and French |
| Logo | A logo with a map and a globe  Description automatically generated |

1. **History**

N-Tec Labs is a Cameroon based non-governmental tech start-up institute founded by **Mr. Ngwanyi Joel in** 2022 which proposes IT solutions and empowers the use of new technologies and also trains various people in Various IT domains in Cameroon.

1. **Mission**

The mission of N-Tec Labs mainly relates to the empowerment and the perpetuation of technology use. These missions include:

* Design and realization of mobile-oriented software for companies and individuals;
* Offering training in computer technologies and information technology
* Taking an active part in the sustainable development of the world through innovative solutions and virtual reality.

1. **Vision**

At N-Tec Labs, we believe that the true potential of every organization and individual can be unlocked through the harmonious fusion of analysis, development, realization, and data analytics. Our vision propels us forward, guiding us to make a lasting impact on the digital landscape and shape a future where innovation knows no bounds.

1. **Activities**

The activities of N-Tec Labs range from computer sciences, engineering, and training. We can outline the following:

* Computer Training: Hardware, Web Mastery, Secretaryship
* Software development and maintenance;
* Training in Software related fields;
* CCTV Installations
* Wired / Wireless Networking
* IT support

1. **ORGANISATION OF THE ENTERPRISE**
2. **Administrative Organisation of N-Tec Labs**

N-Tec Labs is administratively organized as follows;

1. **The General Management**

This is the decision-making department of the company. They perform some of the following functions

* Ensure the office runs properly
* Evaluate the realization of projects by the company

1. **The Teaching Department**

This department is in charge of the school activities of the enterprise.

1. **The Development Department**

This is the heart of the company because they carry out the company’s projects. They perform some of the following functions:

* Re enforce the application standards of the company
* Write training content
* Maintenance
* Innovations

1. *Functional organization of N-Tec Labs*

The functional branch of N-Tec Labs is organised as follows:

*Figure 2: Organisation Chart of N-Tec Labs*

PRESIDENT OF N-TEC LABS

SECRETARY

DEVELOPMENT TEAM

SUPERVISOR

INTERNS

STUDENTS

1. **HARDWARE AND SOFTWARE RESOURCES OF THE COMPANY**
2. **Hardware Resources**

***Table 2****: Hardware resources of N-tec labs*

|  |  |  |
| --- | --- | --- |
| Designation | Quantity | Characteristics |
| Mac Desktop | 4 | APPLE |
| Hp Desktop | 1 | HP |
| Star link | 1 | / |
| Printer | 2 | HP |
| Training equipment and office furniture | 22 | / |

### **Software Resources**

As an enterprise specialised in IT, N-Tec also comprises of software resources such as;

***Table 3****: Software resources of N-Tec Labs*

|  |  |
| --- | --- |
| Designation | Software |
| Operating system | Windows 11, Mac OS |
| Design tools | Photoshop |
| Integrated development environment (IDE) | Vs Codium |
| Text editor | Sublime text, Notepad++ |
| Database management system (DBMS) | MongoDB, PostgreSQL, MySQL |
| Web browser | Google Chrome, Microsoft Edge, Safari |
| Document editor | Microsoft Office Word |
| Presentation | Microsoft Office PowerPoint |

1. **BRIEF PRESENTATION OF THE PROJECT**

During our insertion phase at N-Tec Labs, we were asked to look for themes which solve a specific problem and something innovative. Our professional supervisor took some time to look into what we proposed as themes and he also proposed some theme ideas. Finally, we were assigned the theme **“RESTAURANT PRE-ORDER SYSTEM”**.

**CONCLUSION**

To end, our insertion phase into N-tec labs was a serene, warm and convenient experience with the company’s personnels. One of the things we learned during this phase was collaboration and discipline which is essential for a successful career.

# **PART II:**

**EXISTING SYSTEM**

Preamble

The existing system is a document that provides a view of the system currently in place, that is how it carries out its different activities, also it provides a deep understanding of this system associated to the various limitations, the problems that result from these and the solution we propose

Content

INTRODUCTION

1. PRESENTATION OF THE THEME
2. DESCRIPTION OF THE EXISTING SYSTEM
3. LIMITS OF THE EXISTING SYSTEM
4. PROBLEMATICS
5. DELIMITATION OF THE FIELD OF STUDY
6. PROPOSED SOLUTION

CONCLUSION

**INTRODUCTION**

As engineers, our primary role is to solve problems and enhance everyday life. To develop effective solutions, we must first understand the root causes of existing issues and how they are currently addressed. Analysing this system is essential for identifying its shortcomings and areas for improvement. By evaluating current operations, we can ensure our proposed solution effectively addresses these challenges and enhances the overall dining experience. Thus, understanding the existing system is crucial before implementing a more efficient and user-friendly application.

1. **PRESENTATION OF THE THEME**

Our theme is the “DEVELOPMENT OF A PRE-ORDER AND ORDER RESTAURANT SYSTEM.” This application aims to revolutionize the dining experience by enabling customers to conveniently pre-order their meals before arriving at the restaurant. In today’s fast-paced world, where efficiency and convenience are paramount, the need for a streamlined ordering process is essential, especially during peak hours when restaurants are often overcrowded. Long wait times can lead to customer dissatisfaction and a decline in repeat business. This system is designed to address these challenges by allowing customers to browse a comprehensive menu, customize their meal selections. By pre-ordering, customers can bypass the inconvenience of waiting, ensuring their meals are freshly prepared and ready upon arrival. Furthermore, the system includes features for order tracking, enabling customers to receive real-time updates about the status of their meals, thus minimizing uncertainty and enhancing satisfaction. The Pre-Order and Order Restaurant System not only streamlines the ordering process but also offers a modern, technology-driven dining experience that aligns with customer expectations for speed and convenience. By embracing this innovative solution, restaurants can gain a competitive edge in a rapidly evolving market while providing customers with an enjoyable and efficient dining experience.

1. **DESCRIPTION OF THE EXISTING SYSTEM**

The existing system refers to the framework already in place, which has led to the decision to develop a more efficient solution. The current system presents several challenges that disrupt smooth operations, making it crucial to fully understand these issues before moving forward. Through thorough research and investigation, we have identified key problems that are affecting the user experience and operational efficiency. This phase of analysis serves as foundational research to outline the framework for our proposed solution. After all possible studies carried out on the existing system we identified the following consequences of the various problems. in addition to these consequences, we provided some proposed solutions:

1. **LIMITATIONS OF THE EXISTING SYSTEM**

*Table 4: Limitations Of the Existing System*

|  |  |  |
| --- | --- | --- |
| Criticism | Consequences | Proposed Solutions |
| Slow ordering process via waiter | Customers experience longer wait times, leading to frustration and dissatisfaction. | Implement an app-based ordering system allowing customers to place orders directly through their phones. |
| Limited menu visibility | Customers may not be aware of special offers or detailed dish descriptions, limiting their choices. | Provide a digital menu within the app that includes item details, pictures, and real-time specials. |
| Human error in order taking | Miscommunication between waiters and the kitchen can lead to incorrect orders or missed items. | The app automates order transmission directly to the kitchen, reducing the chances of human error. |
| Difficulty tracking customer preferences | Waiters have difficulty remembering individual customer preferences for future visits. | Use the app to store customer preferences and order history, enabling personalized recommendations. |
| Waiter availability issues | Waiters may be busy or unavailable when customers need assistance, causing delays in service. | Allow customers to request assistance or modifications through the app, ensuring quicker responses. |
| Inaccurate billing due to manual calculations | Errors in manually calculating bills can cause discrepancies and customer dissatisfaction. | Automate bill generation through the app to ensure accuracy and transparency in billing. |
| Difficulty handling large groups efficiently | Large parties may face delays in placing and receiving orders, impacting their overall dining experience. | The app can offer group ordering features, allowing multiple guests to place their orders simultaneously. |

1. **PROBLEMATIC**

With the limitations of the current restaurant system clearly defined, we can pinpoint the core issue: “**HOW CAN WE OPTIMIZE AND DIGITIZE THE ORDERING PROCESS IN RESTAURANTS?”**  This question serves as the foundation for developing the new System, which seeks to tackle many of the challenges identified, such as long wait times, inefficient order management, and limited meal customization options. By harnessing technology, the proposed system will streamline the ordering process, improve operational efficiency, minimize errors, and ultimately enhance the overall dining experience. This transformation aims to create a more efficient and enjoyable environment for both customers and restaurant staff.

1. **PROPOSED SOLUTION**

After analyzing the current challenges in the existing system, particularly long wait times and inefficiencies in order management, we propose to design a **Web application** that will allow both **restaurants and customers** to benefit from a more efficient dining experience.

**For Restaurants:**

* Manage orders through an intuitive dashboard that displays pre-orders based on scheduled pick-up or dine-in times.
* Streamline kitchen workflows by preparing meals in advance based on customer reservations.
* Track and update order statuses, notifying customers of order completion in real-time.
* Improve customer management by reducing wait times and enhancing overall service quality.

**For Customers:**

* Browse comprehensive, categorized menus to select and customize meals (e.g., add/remove ingredients).
* Pre-order meals in advance and specify desired pick-up or dine-in times, ensuring orders are ready upon arrival.
* Track order status in real-time, from preparation to completion, through notifications.
* Reorder previous meals easily from recent order history, speeding up the ordering process.

**CONCLUSION**

Having concluded this section, we conducted a thorough study of the existing system by surveying Customers. This survey helped us identify the system's limitations and the resulting challenges. Based on these findings, we proposed solutions to address these shortcomings. Ultimately, we recommended a web application as a comprehensive solution. This step was essential, as understanding how the current system operates allows us to effectively streamline and enhance the processes involved, ensuring a more efficient and user-friendly experience.

**PART III:**

**SPECIFICATION BOOK**

Preamble

The primary objective of the specification book is to outline the comprehensive specifications for our project, detailing functional and non-functional requirements, UI design, system limitation, and database structures serving as the development guide.

Content

INTRODUCTION

1. CONTEXT AND JUSTIFICATION OF STUDY
2. OBJECTIVES OF THE PROJECT
3. EXPRESSION OF USER NEEDS
4. PLANNING OF THE PROJECT
5. ESTIMATION OF COST OF THE PROJECT
6. CONSTRAINTS OF THE PROJECT
7. DELIVERABLES

CONCLUSION

INTRODUCTION

**INTRODUCTION**

The Specification Book lays the vital groundwork for this software project by mapping out the vision, requirements, designs and plan that will guide us from idea to completion. This thorough documentation serves as a roadmap and compass to smoothly navigate the development journey and ensure the end goal is achieved.

1. **CONTEXT AND JUSTIFICATION OF STUDY**
2. CONTEXT

The restaurant industry is facing growing challenges in meeting customer expectations for speed and convenience, particularly during peak hours when long wait times can lead to dissatisfaction. Traditional systems, where customers order upon arrival, create inefficiencies in kitchen operations and service delivery. This study addresses these issues by developing a **pre-order and order system** that allows customers to reserve their meals ahead of time and in restaurants.

1. JUSTIFICATION

The restaurant industry faces rising demands for faster service and improved customer experience, particularly during peak hours when long wait times often lead to dissatisfaction. Traditional models, where customers order upon arrival, result in unpredictable delays, especially during busy periods.

The Pre-Order Restaurant System addresses this by allowing customers to reserve meals in advance and specify a pick-up time, ensuring their food is ready when they arrive. This system streamlines operations, reduces wait times, and enhances customer satisfaction. For restaurants, it optimizes kitchen workflows, minimizes errors, and improves efficiency. By offering a modern, tech-driven dining experience, the system helps restaurants stay competitive in a rapidly evolving market focused on convenience.

1. **OBJECTIVES OF THE PROJECT**

**A. General Objective**  
The main objective of this project is to streamline the restaurant ordering process by allowing customers to pre-order their meals , have them prepared before arrival and also being able to track their orders through their phones in restaurants , improving both customer experience and restaurant efficiency.

**B. Specific Objectives**

* Enable customers to log into the platform, browse the menu, and reserve meals in advance by specifying a dine-in time.
* Ensure that customers receive real-time updates on the status of their orders, from preparation to ready, minimizing wait times.
* Allow restaurants to manage kitchen workflows more effectively by distributing order preparation based on pre-order reservation times, reducing pressure during peak hours.
* Provide restaurant staff with an easy-to-use interface to view, manage, and update the status of incoming orders.
* Improve overall operational efficiency by reducing delays, minimizing errors in order handling, and enhancing resource management.
* Facilitate a modern, convenient, and tech-driven dining experience that aligns with customers’ expectations for speed and convenience.

**III. EXPRESSION OF USER NEEDS**

This section outlines the essential requirements for the **Pre-Order Restaurant System**, focusing on the system's goals from the user's perspective. These requirements are divided into two categories: functional requirements, which detail the system’s features, and non-functional requirements, which specify the quality and performance characteristics that enhance user experience.

* 1. FUNCTIONAL NEEDS

These are the key features that define what the system is designed to do:

* **Sign Up & Login for Customers and Staff**: Both customers and restaurant staff must be able to create accounts and securely log in to access personalized features.
* **Menu Browsing & Meal Selection**: Customers can browse a categorized menu (e.g., appetizers, main courses, desserts, beverages) to easily find and select their desired meals.
* **Meal Customization**: The system allows customers to customize their meal orders by specifying preferences (e.g., add extra ingredients, remove certain items, select portion sizes).
* **Pre-order:** Customers can place orders in advance and order status updated ensuring their meals are ready upon arrival.
* **Order Tracking** Customers receive real-time updates on the status of their orders, including preparation, ready, and any potential delays.
* **Reorder from Previous Orders**: The system allows customers to quickly reorder meals from their order history, streamlining the process for frequent users.
* **Order Management for Staff**: Restaurant staff can view, update, and manage incoming orders in real-time, helping them to plan preparation based on the pick-up or reservation times.
  1. NON-FUNCTIONAL NEEDS

Non-functional needs refer to the quality attributes that dictate how the **Pre-Order and Order Restaurant System** should operate to provide an optimal user experience. These features ensure the platform's performance, security, and reliability over time, contributing to long-term user satisfaction.

* **Security**: Personal information, including customer details, staff credentials, and order history, must be securely stored and encrypted to protect user privacy. Secure payment gateways should be implemented to safeguard financial transactions.
* **Performance**: The system must be fast and responsive, even during high traffic periods. Pages should load quickly, orders must be processed efficiently, and downtime should be minimal to avoid any interruptions that could frustrate users.
* **Usability**: The platform should have an intuitive, user-friendly interface that allows customers to easily browse menus, customize orders, and track their meals without confusion. A simple, clean design is crucial for both first-time users and returning customers.
* **Reliability**: The system must function consistently without crashes or errors, ensuring orders are accurately processed and delivered as expected. Reliability is key to building trust with users and ensuring the platform meets operational and market expectations.

1. **PLANNING OF THE PROJECT**
   1. ESTMATION OF RESOURCES NEEDED
2. Software Resources

*Table 5: Software resources needed project development*

|  |  |  |  |
| --- | --- | --- | --- |
| SOFTWARE | USAGE | Logo | PRICE(FCFA) |
| Microsoft 365 | Used for the creation our report and PowerPoint |  | 500,500 |
| Visual studio code | Open and free software for the coding part of the project |  | Free |
| PowerAMC | For doing the design of the diagrams to represent the structure and the interaction between the different actors |  | 150,000 |
| Google | For doing the research on the internet and testing of the application |  | Free |
| Windows 11, version 23H2 | Operating system |  | Free |
| Gantt Project | For building a gantt chart for the planning of the work |  | Free |
| XAMPP | For creating database |  | Free |
| Visual Paradigm Enterprise version | For doing the design of the diagrams to represent the structure and the interaction between the different actors |  | 500,000 |
| TOTAL | **1,150,500** | | |

1. Hardware Resources

*Table 6: Hardware resources needed project development*

|  |  |  |  |
| --- | --- | --- | --- |
| Material | USAGE | QUANTITY | PRICE (FCFA) |
| LAPTOP (LENOVO i5 8th Gen, SSD) | Used to build the system, type the report, create the power point, make research, etc. | 01 | 425,000 |
| USB flash disk 64gb | Used for storage and to print the report | 01 | 16,000 |
| Internet Modem | Used for internet connection | 01 | 65,000 |
| Samsung Tablet | Used for carrying research and storage | 01 | 150,000 |
| Printer HP Officejet 4630 | Used to print the scientific document | 01 | 86,550 |
| TOTAL: | **742,550** | | |

1. Human Resources

*Table 7: Human resources needed project development*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Role | Duration (Week) | Quantity | Unit Price (FCFA) | Total Price (FCFA) |
| Project manager | 08 | 01 | 250,000 | 2,000,000 |
| DevOps | 06 | 01 | 205,000 | 1,230,000 |
| Analyst | 02 | 01 | 200,000 | 4,00,000 |
| Designer | 02 | 01 | 150,000 | 300,000 |
| Developer | 04 | 01 | 100,0000 | 400,000 |
| Tester | 02 | 02 | 50,000 | 50,000 |
| TOTAL | **4,680,000** | | | |

1. Overall Resources

Table 8: Overall resources needed for the project development

|  |  |
| --- | --- |
| Designation | Cost (FCFA) |
| Software Resources | 1,150,500 |
| Hardware Resources | 742,550 |
| Human Resources | 4,680,000 |
| Unforeseen | 500,000 |
| Total in figures | **7,073,050** |
| Total in Words | **Seven million seventy-three thousand fifty.** |

* 1. PROJECT PLANNING

*Table 9: Estimation of time required for project*

|  |  |  |  |
| --- | --- | --- | --- |
| Phase | Task / Objective | Output | Duration (WEEKS) |
| Insertion Document | Collection of Information concerning the Enterprise | Insertion Report | 02 |
| Specification Book | Specification of the user’s need | Specification Book | 01 |
| Analysis | Study of the existing system, modelling user interactions | Analysis Report | 02 |
| Conception | Preliminary Conception  Detailed Conception | Conception Report | 01 |
| Realization | Implementation, Unitary tests, Integration test, development | Realization document | 01 |
| User Guide | User guide for solution | User guide | 02 |
| Presentation | Design slides showing how work was realised | PowerPoint | 01 |

* 1. GANTT DIAGRAM

This diagram in planning, scheduling and monitoring the project. It was realised using the software **GANTT PROJECT.**

Figure3: Gantt Diagram

1. **PROJECT CONSTRAINTS**
2. Criterions Of Acceptability

The delivered product is judged acceptable if it respects the different functionalities that have been presented.

1. Time Constraint

The project will be realized in 13 weeks (3 months) starting from the beginning date, July 3rd, 2023.

1. Cost constraint

The realization of our project will require expenditures in human resources, material and software a total cost of **Seven million seventy-three thousand fifty FCFA.**

1. **LIST OF PARTICIPANTS AND DELIVERABLES**
   1. LIST OF PARTICIPANTS

*Table 10: List of Participants*

|  |  |  |
| --- | --- | --- |
| NAME | FUNCTION | ROLE |
| Miss NFORMI Victory | Follows up interns at the  company level | Professional supervisor |
| MR NGUH Prince | Follows up student at the  academic level | Academic Supervisor |
| BONG SHALOM Ngwi | AICS Student | Student at AICS |

DELIVERABLES

Regarding this project, below lies its deliverables:

* A report composed of the following:
  + The analysis documents
  + The conception documents
  + The realisation document
  + The user guides

**CONCLUSION**

In conclusion, this section represents a crucial milestone, demonstrating the application's market value, its benefits for Cameroonians, and providing a clear timeline with cost estimates for successful project completion.

# **PART IV:**

# **ANALYSIS BOOK**

Preamble

The main goal of the analysis phase is to thoroughly capture the users' needs, define the project’s boundaries, and obtain a clear understanding of the restaurant system being developed. To achieve this, we will utilize UML (Unified Modelling Language) combined with the 2TUP (2-Track Unified Process) methodology, which is specifically applied to analyse the system’s functionality. Content

INTRODUCTION

1. PRESENTATION OF THE MODELING APPROACH
2. CHOICE OF THE ANALYSIS METHOD
3. MODELING OF THE PROPOSED SOLUTION

CONCLUSION

**INTRODUCTION**

Building a restaurant pre- order and order app is like embarking on a creative journey, where the first step is to deeply understand the existing system and set clear goals for the future. This phase goes beyond just identifying where we are but it’s about pinpointing key challenges and providing targeted solutions that enhance the dining experience. We apply innovative methodologies and advanced modelling techniques, not only to address the app's requirements but to resolve existing pain points within the system. Whether it's streamlining orders, improving customer engagement, or optimizing restaurant operations, every step of the analysis is aimed at delivering a solution that is both forward-thinking and meticulously crafted. The ultimate goal is to transform identified challenges into opportunities for creating a refined, efficient, and user-friendly app experience.

* 1. **PRESENTATION OF THE MODELING APPROACH**
     1. SOME MODELLING TECHNIQUES

In analysing the software project, it's essential to select a model that effectively designs and visualizes the project's structure based on its specific needs. Here, we explore various modelling techniques and discuss the chosen approach for the project's development.

1. MERISE

**MERISE** (Méthode d'Étude et de Réalisation Informatique pour les Systèmes d'Entreprise) is a structured methodology used in the development and management of information systems. Developed in France in the late 1970s, it primarily focuses on the design and modeling of data and processes within an organization. MERISE is based on the separation of data and processes, allowing for a more modular and flexible approach to system development. It separates system design into three levels: conceptual (high-level understanding of data), logical (structured data and processes), and physical (implementation). MERISE emphasizes data modeling using the Entity-Relationship model and is particularly useful for complex, data-driven projects. It’s widely used in French-speaking regions for systematic and modular software development.

1. SCRUM

Scrum is an Agile framework introduced in the early 1990s, designed to manage complex product development through collaboration, adaptability, and continuous feedback. It breaks projects into short cycles called **Sprints**, typically lasting 2-4 weeks, during which a cross-functional team focuses on delivering specific, prioritized tasks. Key roles include the **Scrum Master** (ensuring smooth process), **Product Owner** (defining and prioritizing requirements), and the **Development Team** (executing the work). Scrum also emphasizes practices like **Daily Stand-ups**, **Sprint Planning**, and **Retrospectives** to promote communication and continuous improvement. Unlike traditional methods, Scrum delivers small, testable product increments throughout development, allowing for early validation and quick adaptation to feedback.

1. UP

UP Stands for “Unified Process” it is sometimes associated with the **Rational Unified Process (RUP)** due to a particular commercial variant, is an object-oriented and use-case-driven approach to software development. These iterative and incremental framework structure the development phase into four sequential phases: **Inception (**initial planning and requirements gathering), **Elaboration (**refining the architecture and resolving high-risk elements), **Construction (**developing the bulk of the software), and **Transition (**deploying the solution to users). Throughout, UP employs the **Unified Modeling Language (UML)** as a standard notation to visualize and design system architecture. With its disciplined approach to assignments and roles, UP integrates various best practices from software engineering.

1. DYNAMIC SYSTEMS DEVELOPMENT METHOD (DSDM)

**DSDM (Dynamic Systems Development Method)** is one of the earliest Agile methodologies, introduced in the 1990s as part of the Agile Alliance. It is designed to help organizations deliver high-quality software solutions quickly and efficiently, while ensuring flexibility and collaboration throughout the development process. The core focus of DSDM is on meeting business objectives through a balanced approach that emphasizes **collaboration**, **timely delivery**, and **adaptability.**

* + 1. **PRESENTATION OF UML, 2TUP AND ERD**
       1. UML

**UML (Unified Modelling Language)** is a standardized, versatile modelling language used extensively in software engineering and system design. Introduced in the 1990s by pioneers Grady Booch, Ivar Jacobson, and James Rumbaugh, UML was created to offer a consistent framework for visualizing, designing, and documenting software systems.

Since its inception, UML has evolved into an indispensable tool for software architects, developers, business analysts, system engineers, and stakeholders. It facilitates clear communication and precise documentation across various aspects of software development.

UML 2.O comprises of 13 diagrams categorized into two main groups

1. **Structural Diagrams**: These diagrams represent the static aspects of a system, illustrating its components and their relationships. Diagrams in this category include:
   * + - Class Diagram
       - Component Diagram
       - Composite Structure Diagram
       - Deployment Diagram
       - Package Diagram
       - Object Diagram
2. **Behavioural Diagrams**: These diagrams depict the dynamic aspects, showcasing how the system behaves and interacts over time. Diagrams in this category include:
   * **Use Case Diagram**
   * **Sequence Diagram**
   * **State Machine Diagram**
   * **Activity Diagram**
   * **Communication Diagram**.
   * **Interaction Overview Diagram**.
   * **Timing Diagram**

With its comprehensive set of diagram types, UML provides a robust foundation for understanding and managing complex software systems, ensuring clarity and coherence throughout the development process. However, it is important to note that UML is not a modelling methodology on its own. To effectively apply UML, it needs to be associated with a structured approach, such as the Unified Process (UP). While there are several variations of the Unified Process, our modelling approach for this project will be the 2TUP (Two-Track Unified Process). This methodology will guide our conceptualization and ensure a systematic approach throughout the project.

* + - 1. COMPARISON BETWEEN UML AND MERISE

*Table 11: Comparison of MERISE and UML*

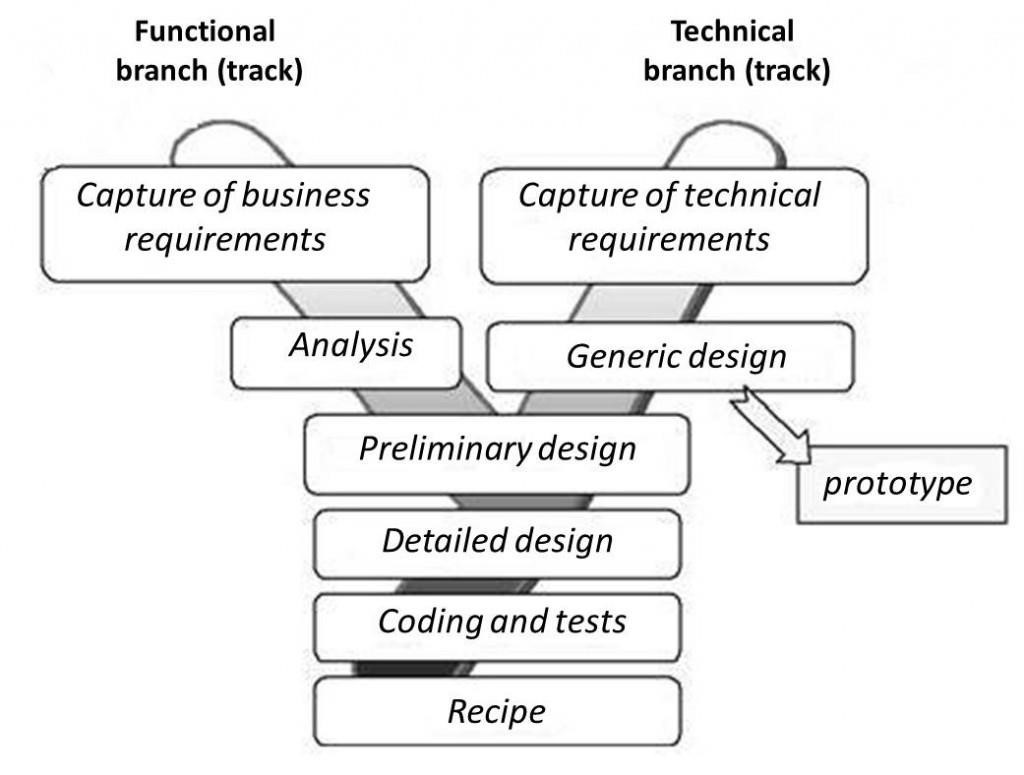
|  |  |  |
| --- | --- | --- |
| Aspect | UML (Unified Modelling Language) | MERISE |
| Primary Focus | General-purpose modelling language for software engineering. | Methodology for designing and managing information systems, with a focus on data and processes. |
| Diagram Types | 14 diagram types, including Class, Use Case, Sequence, and Activity diagrams. | 6 main diagrams, including Context Diagram, Data Flow Diagram, and Entity-Relationship Diagram. |
| Modelling Paradigms | Object-oriented, emphasizing objects and their interactions. | Entity-relationship and process-oriented, focusing on data modelling and process flow. |
| Level of Abstraction | Provides various levels of abstraction from high-level overviews to detailed designs. | Focuses more on data and process modelling with a less diverse set of diagrams. |
| Use Case Focus | Strong emphasis on use cases and user interactions. | Less emphasis on use cases, more focus on data and process modelling. |
| Methodological Approach | Iterative and incremental approach to software development. | More structured and sequential approach, particularly for data analysis and system design. |

This comparison highlights the differences in focus, methodology, and application between UML and MERISE. UML is a broader, internationally recognized modelling language with a focus on object-oriented design, while MERISE is more specialized for data and process modelling, with strong roots in French methodologies.

* + - 1. TWO TRACK UNIFIED PROCESS (2TUP)

In software development, both the process model and the tools for capturing and communicating requirements are crucial for project success. The **2TUP (Two-Track Unified Process)** model provides a strategic approach by integrating conceptual and technical aspects into a structured, Y-shaped flow. This model divides the development process into two parallel tracks: the **Functional Branch** focusing on defining what the system should do, and the **Technical Branch** concentrating on how it will be implemented. These branches converge in the **Realization Branch**, where both tracks are integrated to achieve the final system. The figure below gives us more details on how software development follows the three branches of 2TUP.

*Figure 4: 2TUP diagram (source: https://www.mysciencework.com/omniscience/pervasive-mobile-healthcare-system-based on-cloud-computing).*



This diagram illustrates a software development process involving two main branches: the **Functional Branch** and the **Technical Branch.** Both of these branches work in parallel, and the process converges towards the **Realization Branch** to complete the project.

* + **Functional Branch**
* **Capture of business requirements**: This step involves gathering the business needs and goals. It focuses on understanding the features, functionality, and objectives of the application from a business perspective. Techniques like **use case diagrams** might visualize complex business interactions.
* **Analysis**: After gathering the business requirements, an analysis is performed to evaluate the needs, align them with technical feasibility, and understand how the system will fulfill these requirements.
  + **Technical Branch**
* **Capture of technical needs**: In this step, the technical requirements are identified. It includes the technical specifications necessary to build the system, such as hardware, software, and technical constraints.
* **Generic Design**: The technical requirements are translated into a generic design, which outlines the system's overall architecture, defining how different components will work together.
  + **Realization Branch**

The two branches merge at the realization stage, where the functional and technical designs are implemented in a step-by-step manner:

* **Preliminary Design**: A rough system design that incorporates both business and technical requirements.
* **Detailed Design**: A more refined and detailed version of the system design that specifies every element of the system and how it will be built.
* **Coding and Testing**: The actual coding of the application takes place, followed by testing to ensure it meets both functional and technical requirements.
* **Recipe**: This step likely represents the finalization of the system and could involve preparing the application for deployment or outlining the procedures for implementation.
  1. **CHOICE OF THE ANALYSIS METHOD**

1. JUSTIFICATION AND MOTIVATION FOR THE CHOSEN APPROACH

Our choice for the UML modelling language and the 2TUP development process are based on the following criteria:

* **Justification:**

The Two-Track Unified Process (2TUP), when integrated with Unified Modelling Language (UML) diagrams, provides a clear and structured way to visualize a software system’s functional requirements, technical design, and development plan.

UML provides a common standard, ensuring clear communication among stakeholders and reducing miscommunication risks. Its flexibility, combined with 2TUP's iterative approach, enables ongoing refinement of diagrams as requirements and designs evolve. UML offers a variety of diagram types, such as: use case diagrams (capturing business needs), sequence diagrams (for system interactions), and deployment diagrams (for technical implementation). These diagrams effectively represent each stage of the 2TUP methodology.

* **Motivation:**

1. **Improved understanding:** UML diagrams help teams better comprehend system requirements and designs, leading to a more efficient development process.
2. **Enhanced collaboration:** UML serves as a common language between stakeholders and developers, promoting clear communication and shared understanding.
3. **Early issue identification:** Visualizing the system early on, combined with the iterative approach, helps teams spot and resolve potential issues before they become major problems, reducing risks.

By combining UML with the 2TUP methodology, teams can integrate strategic planning with visualization, improving clarity, collaboration, and efficiency throughout the software development lifecycle.

* 1. **MODELING OF THE PROPOSED SOLUTION**
     1. FUNCTIONAL BRANCH

Here we are focusing on modelling business requirements and system functionality, ensuring clear communication among stakeholders.

* + - * 1. USE CASE DIAGRAM

**Definition**

A use case diagram visually depicts the interactions between users (actors) and a system. It focuses on how actors engage with the system to accomplish specific tasks. This diagram helps outline the system's functional requirements from the user's perspective.

**Formalism**

A diagram of a system

Description automatically generated

Figure 5: Use case Diagram Formalism

**Components Of the Use case Diagram**

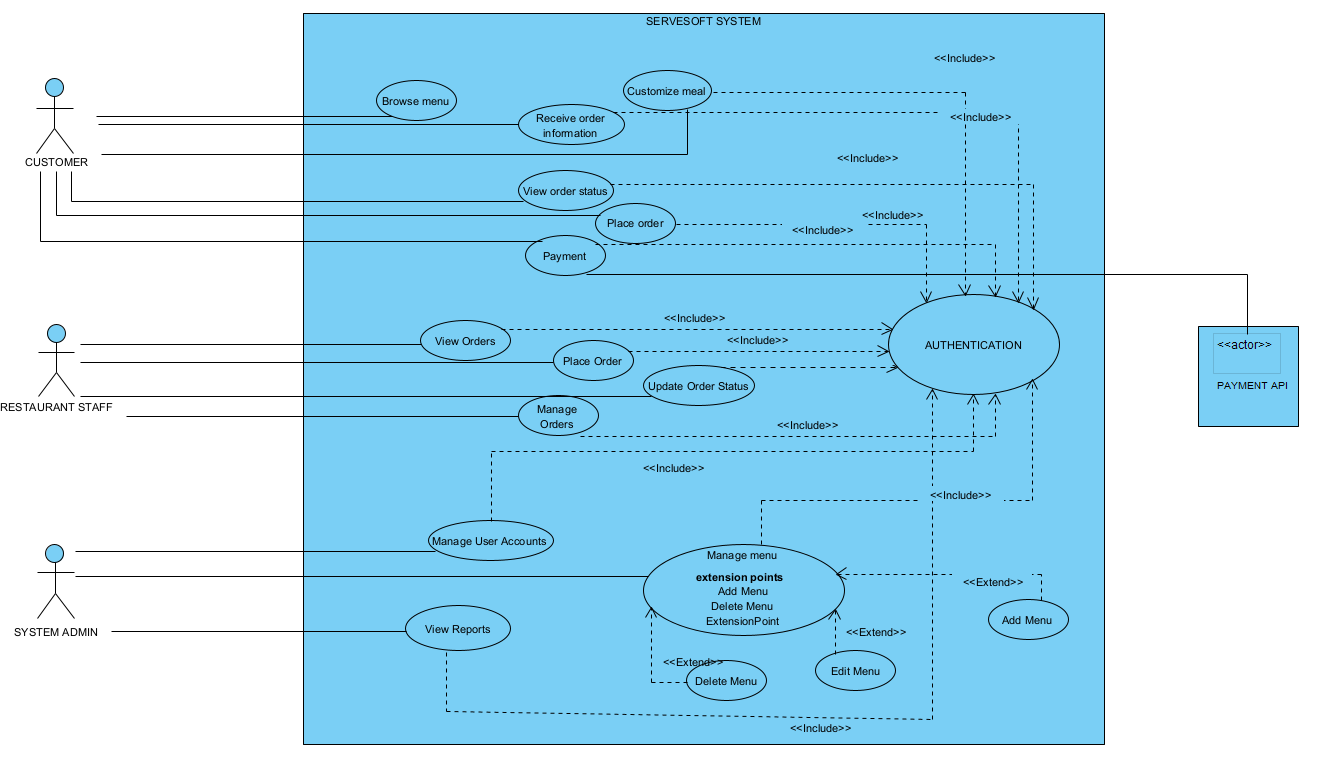
*Table 12: Components Of the Use case Diagram*

|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTION | DIAGRAMMATIC REPRESENTATION |
| Actor | An actor in a use case diagram represents an external entity, such as a user, system, or device, that interacts with the system. |  |
| Use case | A use case represents a functionality or features of the system that interact with actor of the system. |  |
| Communication Link | The participation of an actor and a use case is shown by connecting an actor to a use case and this is done with using a solid link to represent it |  |
| Boundary of System | This is the overall system having in its all the functionalities (use case) inside the system and their interaction |  |
| Extends | Extends represents a conditional relationship where one use case (the extending use case) adds additional behaviour to another use case (the base use case) under specific conditions**.** |  |
| Include | Include represents a relationship where one use case (the base use case) always incorporates the behaviour of another use case (the included use case) as part of its process. |  |
| Generalization | Generalization represents an inheritance relationship where a child use case or actor inherits the behaviour and characteristics of a parent use case or actor. |  |

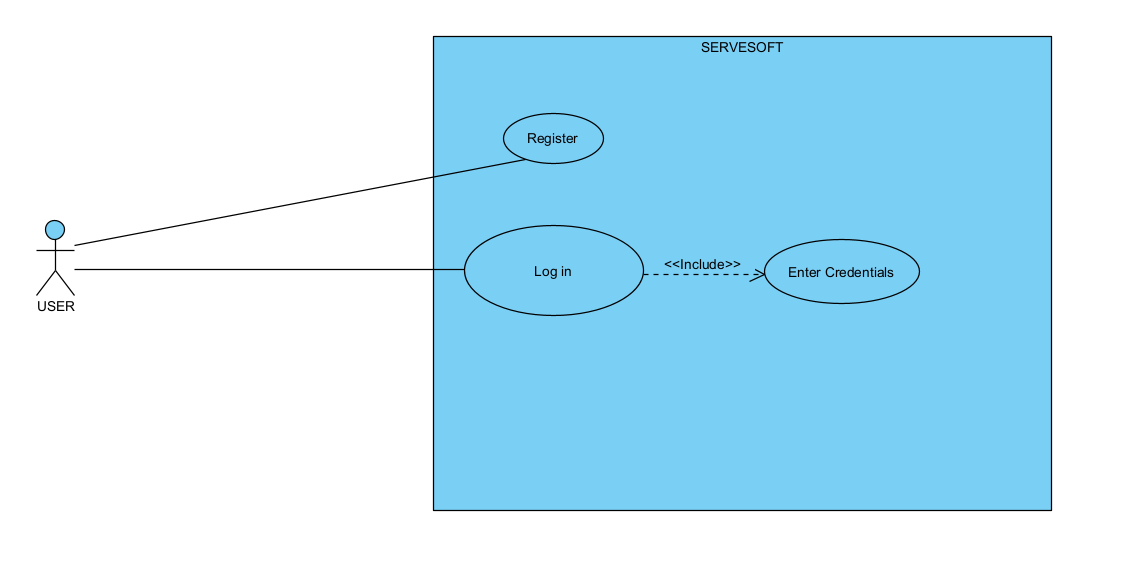
|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTION | DIAGRAMMATIC REPRESENTATION |
| Association | Association represents a relationship between an actor and a use case, showing how the actor interacts with the system. |  |
| Inheritance | Inheritance refers to a relationship where a child actor or use case inherits the properties and behaviours of a parent actor or use case**.** |  |

* LIST OF ACTORS
  + **Administrator:** Responsible for the system including updating the menu, handling orders, overseeing customer service, and monitoring restaurant operations, administration of users.
  + **Customer:** Interacts with the app to browse the menu, place orders.
  + **Restaurant Staff:** interacts with the app, updates order status place orders for people who want to have traditional restaurant services.

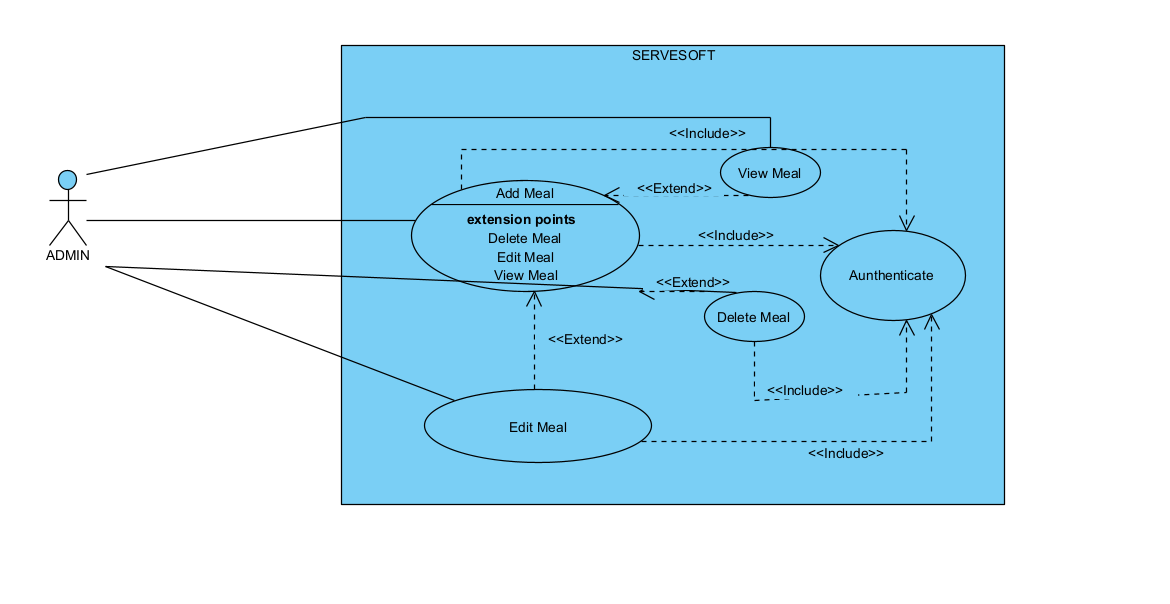
*Figure 6: Use case Diagram for Servesoft*



*Figure 7: Specific Use case Diagram for Authentication*



*Figure 8: Specific Use case Diagram for Meal Management*



**TEXTUAL DESCRIPTION**

**Definition**

A textual description of a use case is a detailed written explanation of how a system functions. It provides a comprehensive understanding of the system by outlining the interactions between various actors and the system, under specific conditions.

**Formalism**

|  |  |
| --- | --- |
| COMPONENT | EXPLANATION |
| Actors | Identify the primary actor (the entity initiating the use case) and any secondary actors (participants or systems that support or interact with the primary actor). |
| Description | Provide a brief, concise summary of the use case’s goal or main objective, outlining what the system is supposed to accomplish from the actor's perspective. |
| Preconditions | Outlines the necessary conditions that must be true or fulfilled before the use case can begin. This might include system states or actor prerequisites. |
| Postconditions | Define the expected state of the system once the use case has been successfully completed. This ensures that the intended goal has been achieved. |
| Nominal Scenario | Describe the typical sequence of events, step-by-step, when the use case is executed under normal conditions and everything proceeds as expected. |
| Alternative Scenario | Provide descriptions of possible deviations from the nominal scenario. This includes variations based on exceptions, errors, or alternate decisions by the actor. |
| Exceptions | Highlight specific conditions or errors that may prevent the use case from reaching its goal. Include how the system should respond to these exceptions. |
| Special Requirements | Detail any non-functional requirements (such as performance, security, or regulatory needs) or other special conditions necessary for this use case to function properly. |

*Table 13: Formalism of Textual Description*

**TEXTUAL DESCRIPTION FOR AUTHENTICATION**

|  |  |
| --- | --- |
| COMPONENT | EXPLANATION |
| Actors | The User who interacts with the system to perform activities such as logging in or signing up. |
| Description | This use case diagram shows how a User interacts with the system for authentication (signing up, logging in) and handling errors such as Invalid Credentials and Account Locked scenarios. |
| Preconditions | The system is operational.  The user must have proper credentials or needs to create an account. |
| Postconditions | If login is successful, the user gains access to the system.  If invalid credentials are entered, an error message is displayed. |
| Nominal Scenario | * + 1. Signup: A new user can create an account in the system.     2. Login: An existing user enters their credentials to access the system. |
| Alternative Scenario | InvalidCredentials: If the user enters incorrect login information, the system shows an error and prompts them to re-enter the correct details. |
| Exceptions | InvalidCredentials: This is the exception when the entered username or password is incorrect. |
| Special Requirements | Security: The login process must ensure proper encryption and data protection to safeguard staff information.  Performance: The authentication process (login, account locking) must be quick to ensure smooth operations. |

*Table 14: Textual Description for authentication*

**TEXTUAL DESCRIPTION FOR MEAL MANAGEMENT**

*Table 14: Textual Description for meal Management*

|  |  |
| --- | --- |
| COMPONENT | EXPLANATION |
| Actors | The admin is the main user of this system, initiating and controlling the various functionalities. They can add, edit, view, and delete meals, as well as authenticate users into the system. |
| Description | This use case diagram illustrates the interactions between the admin and the system's primary functionalities. The admin is responsible for managing meals within the system, which includes adding, editing, viewing, and deleting meals. The authentication process ensures that only authorized users (admins) can perform these actions. |
| Preconditions | The admin must be authenticated before they can access or manage meals in the system. This is depicted by the "Authenticate" use case, which is included in all primary actions (Add, Edit, View, and Delete Meal). |
| Postconditions | After any action is successfully completed (e.g., adding, editing, or deleting a meal), the system will update its meal records accordingly. If an Admin successfully views a meal, they will be presented with the requested information. |
| Nominal Scenario | * + 1. The admin logs into the system (Authenticate).     2. The admin adds a new meal to the system (Add Meal).     3. If the admin wants to modify the meal, they can use the Edit Meal feature (which includes the Authenticate and View Meal processes).     4. The admin can also delete an unwanted meal (Delete Meal), after viewing it to confirm.     5. Viewing a meal allows the admin to see the meal details (View Meal). |
| Alternative Scenario | If the admin forgets to log in, the system will prompt them to authenticate before they can manage any meal-related tasks.  The admin may choose to view meals first before deciding to edit or delete them. |
| Exceptions | Authentication Failure: If the Admin fails to authenticate, they cannot access any system features.  Meal Not Found: If a meal being edited or deleted does not exist, the system will alert the Admin. |
| Special Requirements | Security**:** Only authenticated Admin users can manage meals, ensuring restricted access.  UserInterface**:** The system should have a user-friendly interface allowing the admin to easily navigate between adding, editing, viewing, and deleting meals. |

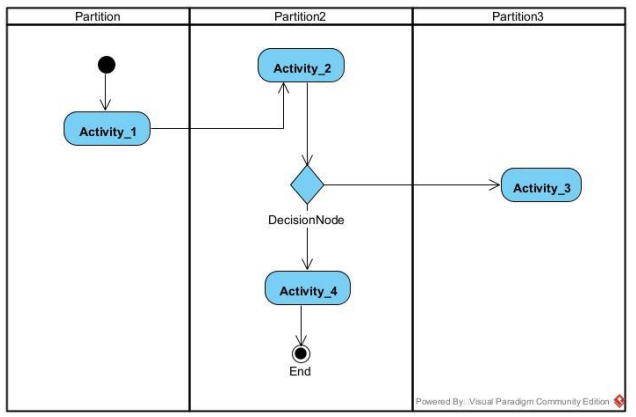
* + - * 1. ACTIVITY DIAGRAMS

**Definition**

These are visual representations of workflows or processes that outline the sequence of activities involved in completing a task or achieving an objective. They illustrate how different actions are performed and how decisions or conditions influence the flow from one activity to the next.

**Formalism**

*Figure 9: Formalism of the activity diagram*

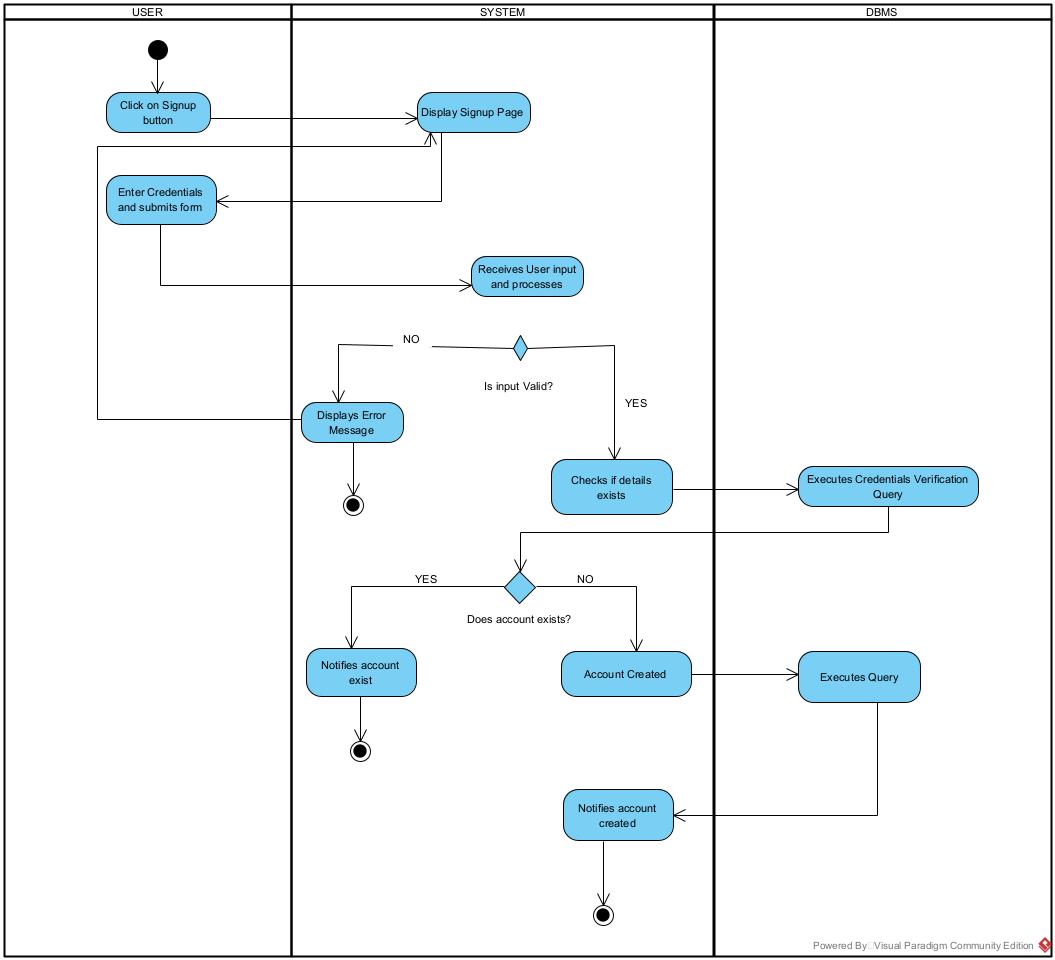


**Components of the Activity Diagram**

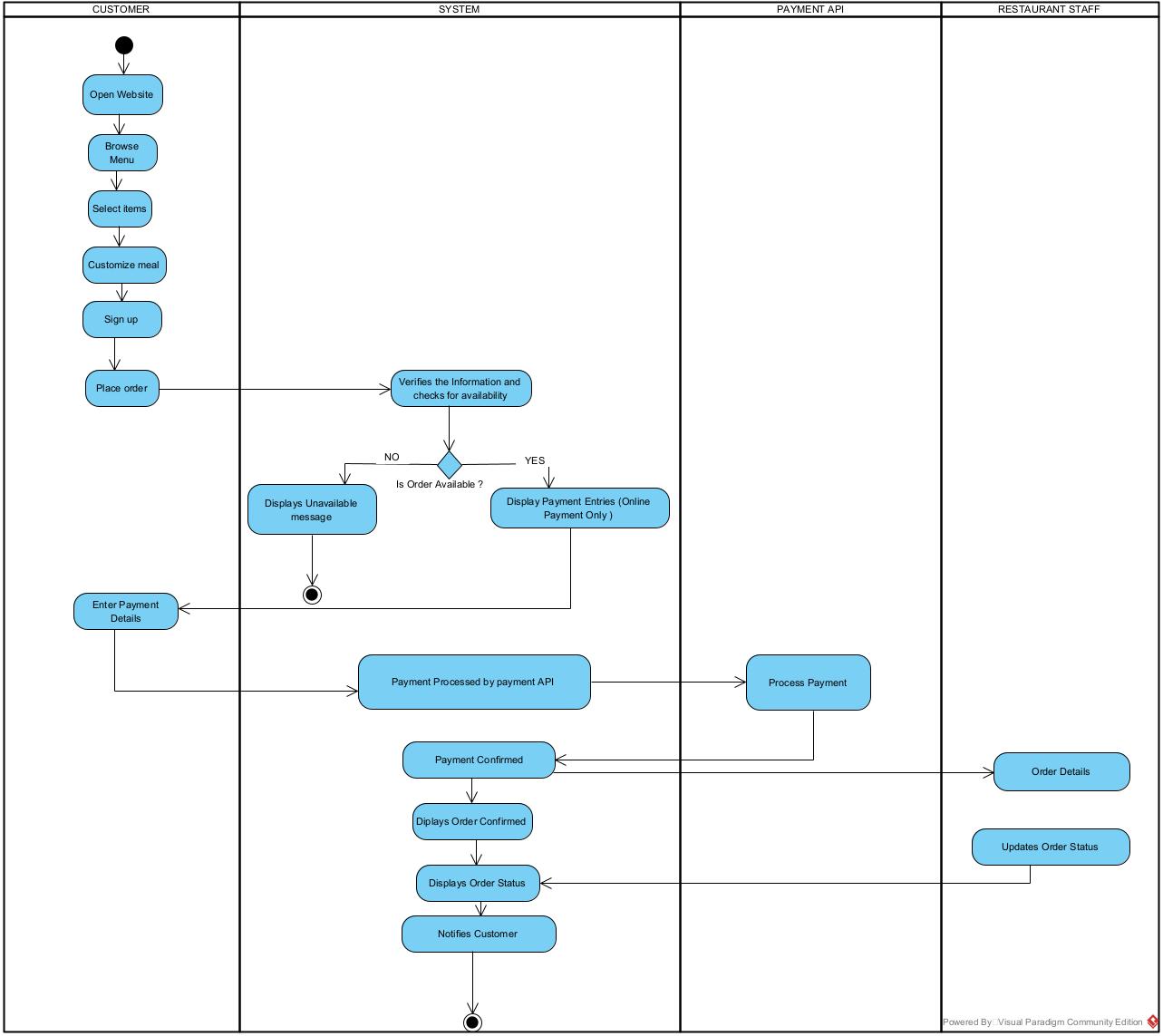
*Table 16: Components of the Activity Diagram*

|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTION | NOTATION |
| Initial node | Represents the starting point of a set of actions or activities within an activity diagram. |  |
| Activity | Denotes a set or sequence of actions to be performed, typically used to encapsulate larger processes or behaviors within the system. |  |
| Action | Refers to a specific operation or task that is to be executed, forming part of an activity. |  |
| Control flow | Illustrates the order in which actions or activities are carried out within the process. |  |
| Object flow | Shows the path of an object as it moves from one activity to another within the process. |  |
| Object node | Represents an object in the activity diagram, often connected to object flows, indicating the role of objects in the workflow. |  |
| Decision node | Depicts a branching point where a condition is tested to determine which path the control or object flow should follow. |  |
| Merge node | Combines different paths that were previously separated by decision nodes, consolidating them into a single flow. |  |
| Fork node | Splits a single flow into multiple concurrent or parallel flows of activities, enabling actions to occur simultaneously. |  |
| Join node | Reunites multiple concurrent or parallel flows back into a single flow, ensuring synchronization of activities or actions. |  |
| partition | Organizes activities by actor or role, or separates activities into threads, highlighting who performs which actions or in which thread they occur. |  |
| Activity final node | Terminates all ongoing control or object flows, marking the completion of an activity or process within the diagram. |  |

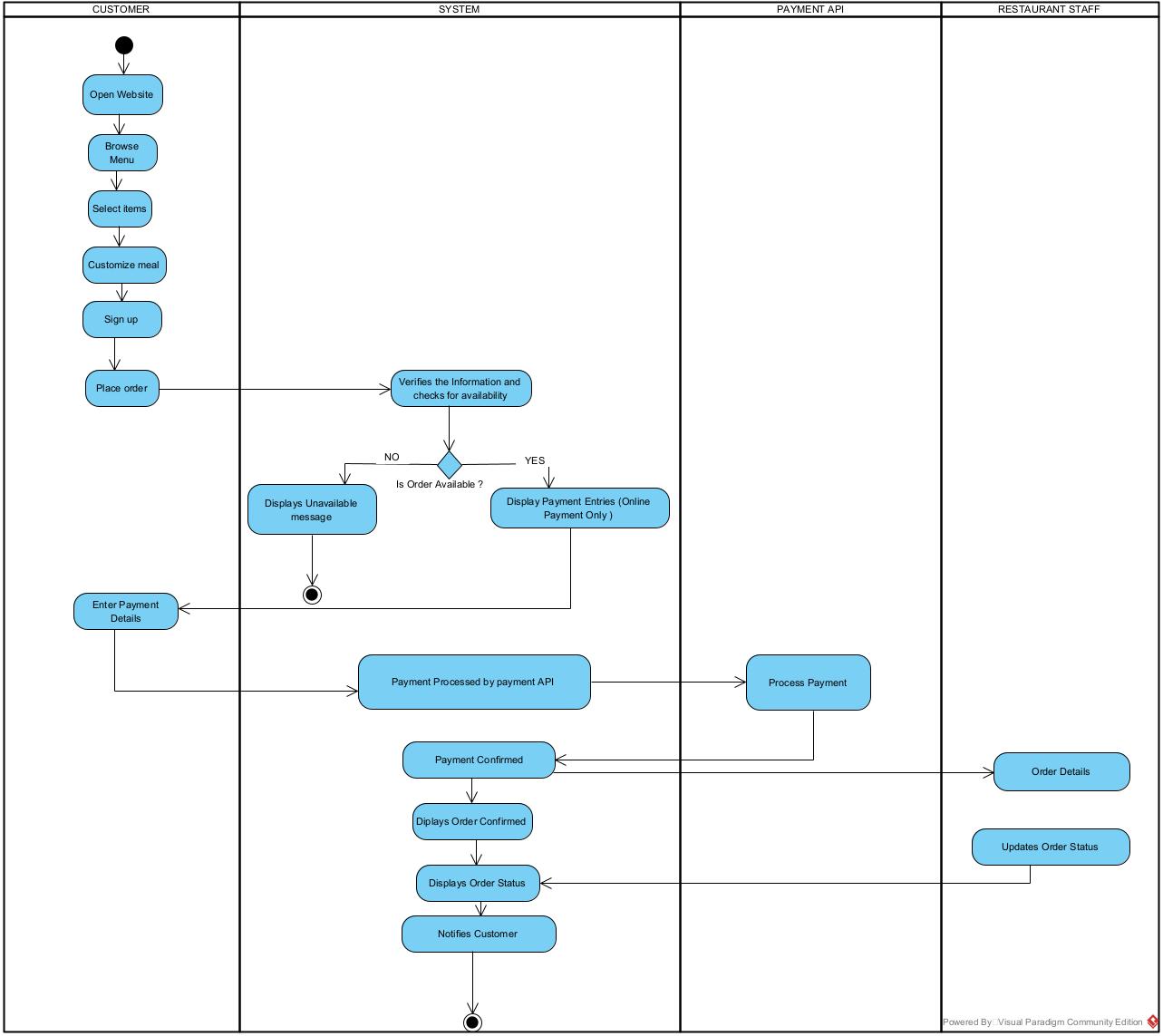
*Figure 10: Activity diagra*m for Authentication



*Figure 11: Activity diagra*m for Pre-ordering



*Figure 12: Activity diagra*m for Ordering



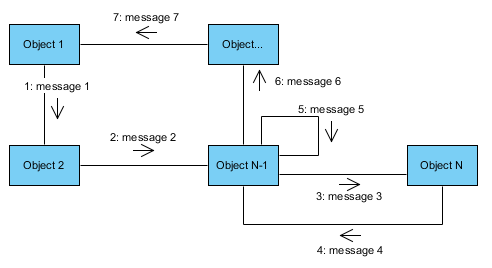
* + - * 1. COMMUNICATION DIAGRAMS

**Definition**

A communication diagram in UML visualizes the interactions between objects, emphasizing their relationships and the flow of messages exchanged to perform tasks. Unlike sequence diagrams, which focus on the timing of interactions, communication diagrams highlight the structure and connections between objects, providing a clear view of how components collaborate within a system.

**Formalism**

*Figure 13: Formalism of communication diagram*

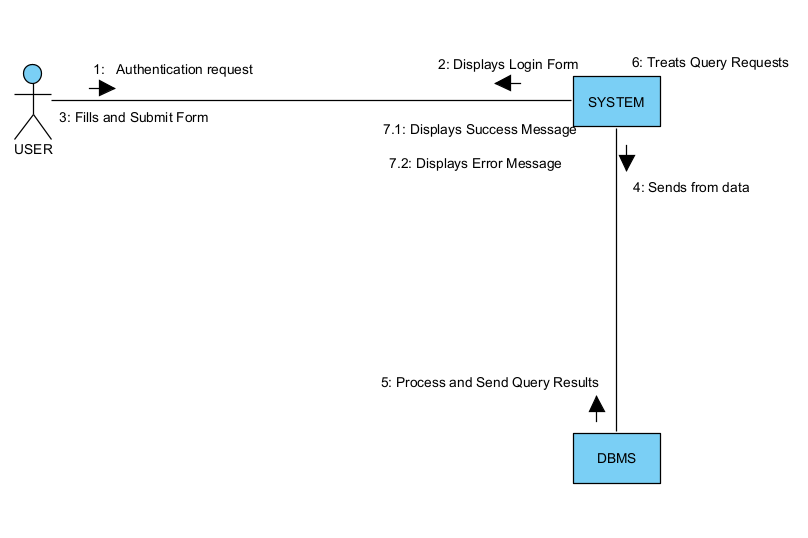


**Components of the communication diagram**

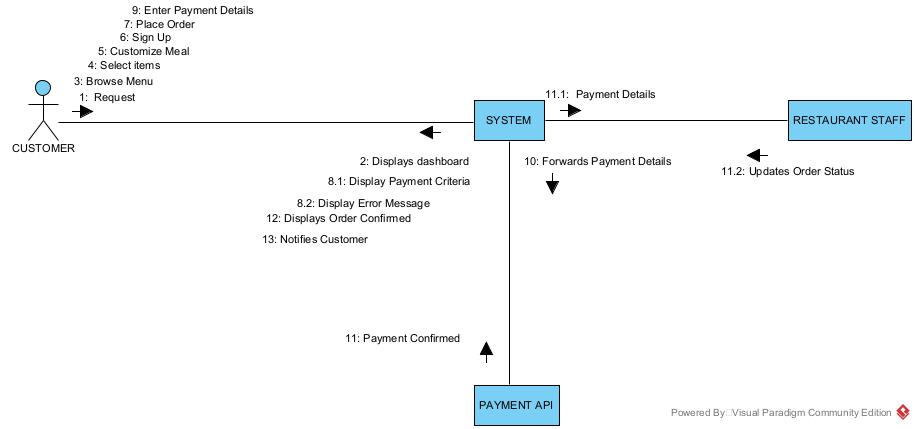
*Table 17: Components of the communication diagram*

|  |  |  |
| --- | --- | --- |
| ELEMENT | DESCRIPTION | NOTATION |
| Frame | The overall use case or sequence is represented within a rectangular frame. The name of the diagram is placed in a separate compartment at the upper-left corner for identification. | A white rectangular object with black text  Description automatically generated |
| Object | Represents an entity involved in the communication process. Objects participate by sending or receiving messages and are typically labeled with their name and role (e.g., User). |  |
| Message | An arrow between two objects that illustrates the flow of communication, specifying the direction and sequence of the interaction. Messages are labeled with a number and a description to indicate the order and type of communication. |  |
| Link | A straight line connecting two objects, symbolizing an established relationship or connection between them. This indicates that the objects can communicate with each other. |  |

*Figure 14: Communication diagram for Authentication*



*Figure 15: Communication diagram for Pre-ordering*



* + - * 1. SEQUENCE DIAGRAM

**Definition**

A sequence diagram visually represents how system components interact over time, showing the order of events and communications between them. With vertical lines depicting object lifetimes and horizontal arrows for interactions, it provides a clear, chronological view of processes, helping to understand system functionality and flow.

**Formalism**

*Figure 16: Formalism of Sequence Diagram*

**A diagram of a diagram of a lifeline

Description automatically generated**

**Components of the sequence Diagram**

*Table 17:* Components of the sequence Diagram

|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTIONS | NOTATION |
| Actor | Represents an external entity, typically a user, that interacts with the system. Illustrated as a stick figure, the actor initiates actions or receives outcomes from the system. |  |
| Lifeline | A vertical dashed line extending from an object or actor, symbolizing the entity’s presence over time within the sequence diagram. It represents the duration for which the object or actor is active. |  |
| Message | A horizontal arrow illustrating communication between objects or actors. The direction of the arrow indicates the sender and receiver of the message, marking the path of interaction. |  |
| Return Message | A dashed arrow indicating the response sent back to the original object or actor after a message. It typically follows an action or request and is used to show feedback or result. |  |
| Activation Bar | A thin vertical rectangle placed on the lifeline, representing the duration in which an object or actor is actively processing a message or task. It indicates the time frame of execution. |  |
| Destroy | A large "X" at the end of a lifeline, showing the point at which an object is terminated or ceases to exist. It marks the end of the object’s involvement in the process. |  |
| Combined Fragment | A rectangular area used to represent conditions, loops, or control structures in the diagram. It often includes labels such as ‘alt’ for alternative paths or ‘loop’ for repetitive actions, and groups related lifelines. |  |
| Note | A comment or annotation box attached to elements in the diagram to provide additional context or clarification. It serves as a means of offering supplementary information about specific parts of the sequence. |  |

**CONCLUSION**

As we conclude the analysis phase for the Restaurant Pre-order and Order App, we've outlined the core functionalities and interactions using UML diagrams. This solid blueprint will guide us into the design phase, ensuring a user-friendly system and a smooth development process.

**PART IV:**

**CONCEPTUAL PHASE**

Preamble

The conception phase, or design phase, translates requirements into clear blueprints, defining architecture, database design, and user interfaces. It ensures the product is user-friendly, secure, and scalable, providing developers with a structured roadmap for a smooth development process.

Content

INTRODUCTION

1. TECHNICAL BRANCH

* Generic Design

1. IMPLEMENTATION BRANCH

* Preliminary Design
* Detailed Design

CONCLUSION

**INTRODUCTION**

The conception phase is a key transition in software development, moving from abstract requirements to detailed design. Using the 2TUP methodology, this phase splits into two branches: technical, which outlines the system architecture, and implementation, which focuses on detailed execution. Together, they form a clear roadmap for smooth development.

* 1. **TECHNICAL BRANCH**
* Generic Design  
  This stage of the technical branch focuses on creating a high-level design blueprint of the software without getting into detailed implementation. It outlines the core structure and overall architecture of the application.
  + 1. Physical Architecture

The physical architecture defines the system's components and how they interact. Our app follows a **3-tier architecture**, with the layers described below:

* **Presentation Tier**: This is the front-end of the application, where users interact via web browsers. In our case, it's built with **HTML, Bootstrap, and CSS**, offering a responsive and user-friendly interface.
* **Application Tier**: This tier contains the core logic, processes requests, and applies business rules. Our app’s back-end is developed in **PHP**, managing requests between the user interface and the database.
* **Data Tier**: This layer handles data storage and retrieval. We use a **MySQL database** for storing restaurant details, user information, orders, and more.
  + 1. Logical Architecture

The logical architecture focuses on internal software design and the patterns used within the code structure. Our app uses the **MVC (Model-View-Controller)** design pattern, breaking the system into three primary components:

* **Model**: Represents the data structure, interacting with the database to manage data operations. In our app, this is where the business logic is applied, linking to the **MySQL database**.
* **View**: This is where data is presented to the user. Built using **HTML, Bootstrap, and CSS**, the view layer displays the restaurant menu, orders, and other details to the user.
* **Controller**: The controller manages user input, handling requests and coordinating between the model and the view. In our case, **PHP** serves as the controller, processing user inputs like orders and managing the flow of information.
  1. **IMPLEMENTATION BRANCH**

The Implementation Branch begins with the Preliminary Design, where UML diagrams like Class Diagrams and Component Diagrams are used to visualize the system’s structure and high-level components. These models help guide the early stages of development by providing a blueprint for implementation.

* PRELIMINARY DESIGN

This phase serves as the foundation for refining system requirements and guiding subsequent development efforts.

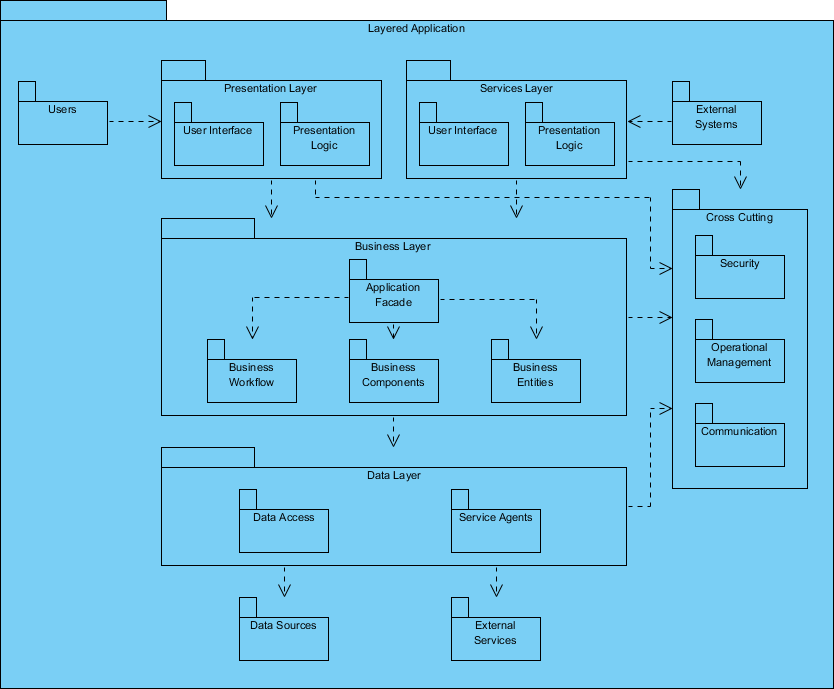
* + 1. PACKAGE DIAGRAM

**Definition**

A Package Diagram in UML showcases how system components or classes are organized into groups (packages) and their dependencies. It offers a high-level modular view of the system's architecture, helping to identify shared components, minimize redundancies, and clarify the overall structure.

**Formalism**

*Figure 16: Formalism of the Package Diagram*

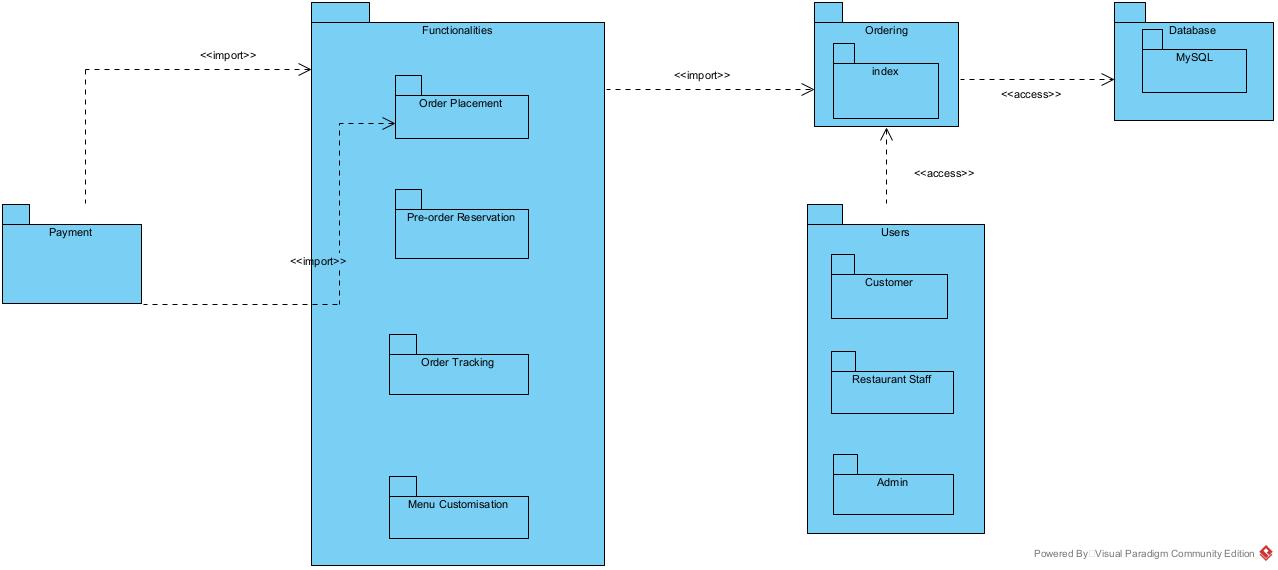


**Components of the Package Diagram**

*Table 18: Components of the Package Diagram*

|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTION | NOTATION |
| Package | A namespace used to group semantically related elements, similar to a folder. Packages can contain other packages, creating a hierarchical structure. |  |
| Dependency | A relationship where one package depends on another. Represented by a dashed arrow, it indicates that changes in the target package may affect the dependent package. |  |
| Package import | Indicates that all publicly visible members of a package are accessible in another package. |  |
| Generalization | Represents inheritance between packages, allowing one package to inherit characteristics or behaviour from another. |  |
| Stereotypes | |  | | --- | |  |  |  | | --- | | Labels or tags added to packages to provide additional semantic meaning. Common examples include <<utility>>, <<system>>, or <<subsystem>>. | | // |

*Figure 17: Package Diagram of The System*



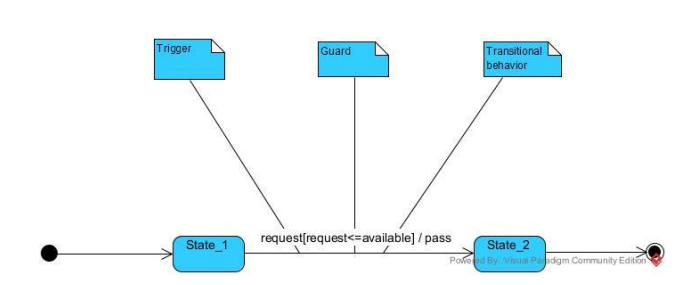
* + 1. STATE MACHINE DIAGRAM

**Definition**

A state machine diagram serves as a visual representation of a system's behavior. It identifies the different states or conditions that the system can enter and illustrates how transitions occur from one state to another based on specific triggers. This diagram is designed to enhance understanding of complex systems. By mapping out the system's behaviors, designers, developers, and users can anticipate its responses, facilitating more effective design, troubleshooting, and interaction with the system.

**Formalism**

*Figure 18: Formalism of the state machine diagram*

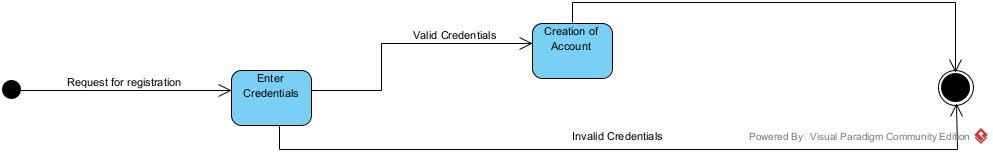


|  |  |  |
| --- | --- | --- |
| ELEMENT | DESCRIPTION | NOTATION |
| Initial State | It defines the initial state (beginning) of a system, and it is represented by a black filled circle |  |
| State-box | It depicts the conditions or circumstances of a particular object of a class at a specific point of time. A rectangle with round corners is used to represent the state box | A blue rectangle with black text  Description automatically generated |
| Decision-box | It is of diamond shape that represents the decisions to be made based on an evaluated guard. | A blue diamond with black outline  Description automatically generated |
| Transition | A change of control from one state to another due to the occurrence of some event is termed as a transition. It is represented by an arrow labeled with an event due to which the change has ensued. |  |
| Final state | It represents the final state (end) of a system. It is denoted by a filled circle present within a circle |  |

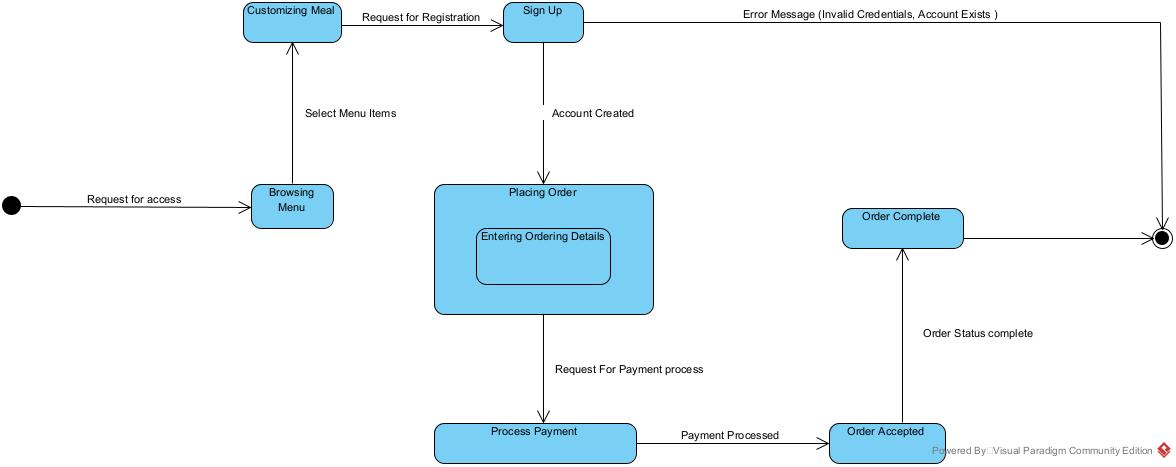
**Components of the State Machine Diagram**

*Table 19: Components of the State Machine Diagram*

*Figure 19: State Machine Diagram for Register*



*Figure 20: State Machine Diagram for Ordering*



* DETAILED DESIGN

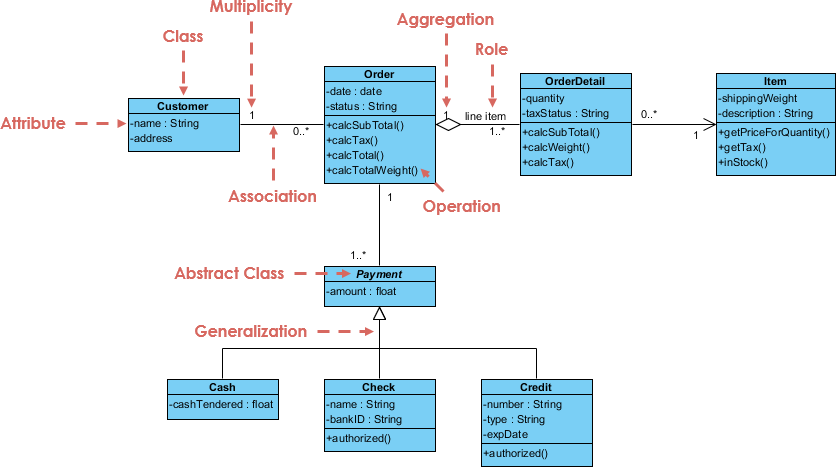
1. CLASS DIAGRAM

**Definition**

A class diagram is a static structure diagram within the Unified Modelling Language (UML) that depicts the structural patterns of a system. It illustrates the relationships between classes and how they interact with one another. Essentially, it serves as a blueprint for the system's architecture.

**Formalism**

*Figure 21: Formalism of the Class diagram*

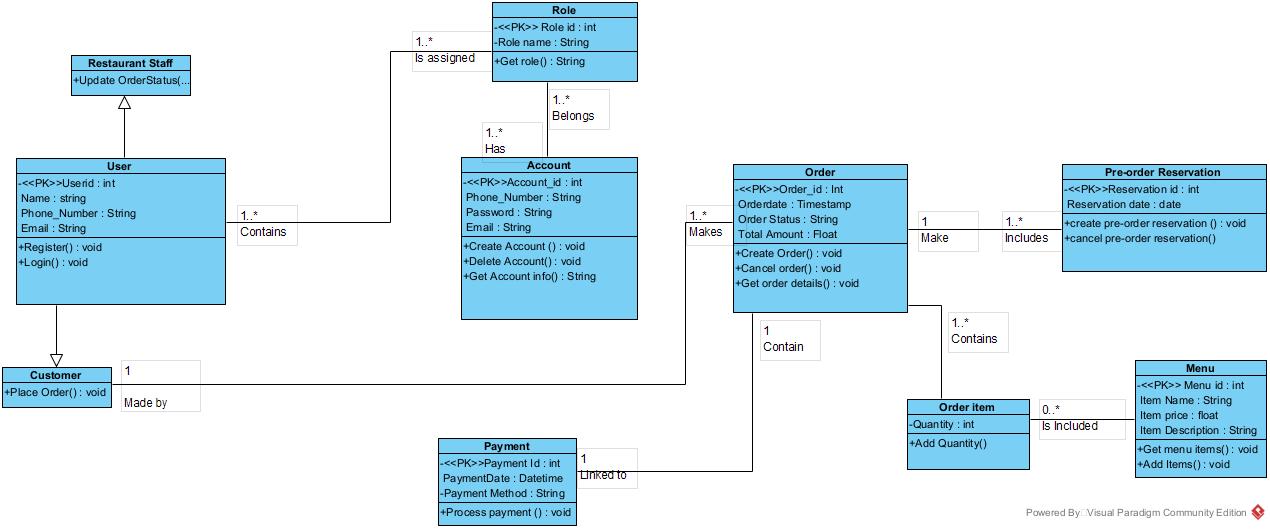


**Components of the Class Diagram**

*Table 20: Components of the Class Diagram*

|  |  |  |
| --- | --- | --- |
| Elements | Description | Notation |
| Class | Symbolized by a rectangle divided into three sections: the top section for the Name, the middle for Attributes, and the bottom for Operations/Methods. Represents entities or objects along with their characteristics. | A blue box with black text  Description automatically generated |
| Association | Represented by a solid line connecting two classes, indicating a relationship between them. |  |
| Generalization (Inheritance) | Depicted by a hollow triangle pointing to the parent class, indicating an "is-a" relationship between the subclass (child class) and the superclass (parent class). | A diagram of a diagram  Description automatically generated |
| Aggregation | Represented by a hollow diamond on the side of the whole, indicating a "whole-part" relationship where the part can exist independently of the whole. | A diagram of a diagram  Description automatically generated |
| Composition | Shown with a filled diamond on the side of the whole, indicating a stronger "whole-part" relationship where the part cannot exist separately from the whole. | A diagram of a diagram  Description automatically generated |
| Dependency | Depicted by a dashed line with an arrow, indicating that one class relies on another. For example, in the diagram, Class1 depends on Class2. | A diagram of a connection between two squares  Description automatically generated |
| Visibility | Indicated by symbols (+, -, #, ~) representing public, private, protected, and package-level visibility, respectively. |  |
| Multiplicity | Represents the number of instances of one class that can be associated with a single instance of another class, expressed as cardinal numbers (e.g., 1, 2) or ranges (e.g., 0..1, 1..\*), typically found at the ends of associations. |  |

*Figure 22: Class diagram of the System*



**CONCLUSION**

In the conception phase, we created the architectural framework and structural blueprint for our application, ensuring a seamless integration of user experience and backend functionalities. Utilizing the 2TUP model and UML diagrams, we achieved a clear understanding of the system's behaviours and state transitions through class, state machine, and package diagrams. With this solid design framework in place, we are now prepared to enter the realization phase, where we will transform these designs into functional modules for our innovative card management application.

**PART VI**

**REALIZATION PHASE**

Preamble

In the software development journey, each step is crucial in shaping the final product. After navigating the complexities of conceptualization and design, we are now on the brink of the realization phase, where our ideas transform into tangible components. This phase reveals the interplay of software modules with hardware, laying the groundwork for a seamless user experience. As we move forward, we will dissect our application, demonstrating how each element contributes to the cohesive whole of our digital solution.

Content

INTRODUCTION

1. COMPONENT DIAGRAM
2. DEPLOYMENT DIAGRAM
3. PRESENTATION OF THE TOOL USED
4. TECHNOLOGY STACK
5. ARCHITECTURE OF THE APPLICATION

CONCLUSION

**INTRODUCTION**

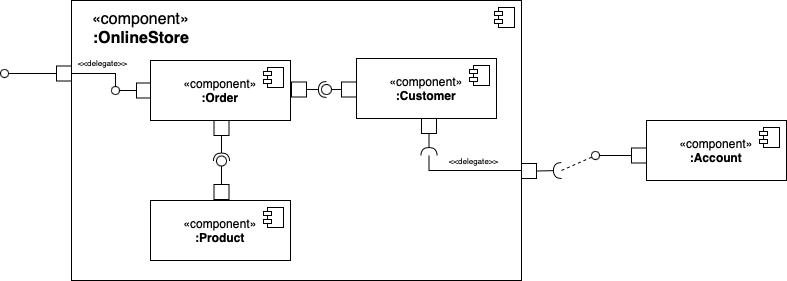
In this phase, we are focusing on two key tools: the Component Diagram and the Deployment Diagram. Think of the Component Diagram as our application's family tree, illustrating each part and their connections. Meanwhile, the Deployment Diagram serves as a launch day game plan, detailing where everything will be positioned and how it will all connect. By the end of this phase, we will have a clear strategy that sets the stage for the upcoming development. Let’s dive in and bring our vision to life!

* 1. **COMPONENT DIAGRAM**

**Definition**

A Component Diagram is a type of structural diagram in the Unified Modelling Language (UML) that visualizes the organization and dependencies among software components. These components may include classes, interfaces, or entire software packages and modules. The diagram offers a high-level overview of a system's architecture, illustrating how software components interact and integrate within the overall system.

**Formalism**



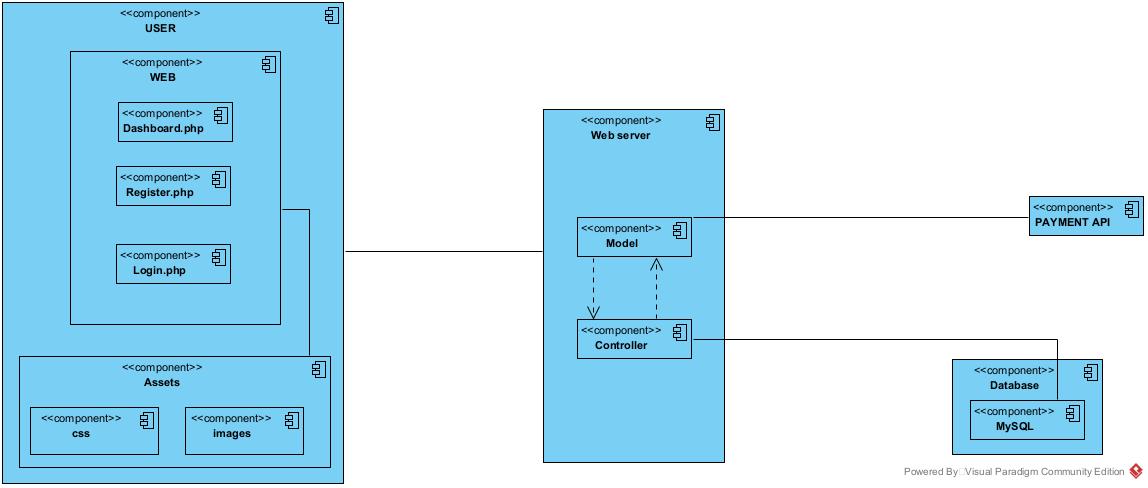
*Figure 23: Formalism of the Component Diagram*

**Components of the Component Diagram**

|  |  |  |
| --- | --- | --- |
| ELEMENTS | DESCRIPTION | NOTATION |
| Component | Represented as a rectangle with the component's name inside, often with an icon of two smaller rectangles on the left. Symbolizes a modular part of the system that encapsulates specific functionality. |  |
| Interface | Depicted as a circle (or "lollipop" symbol), sometimes as a semi-circle attached to a component. Defines a set of operations that outline a component's responsibilities. |  |
| Dependency | Illustrated with a dashed arrow, indicating that one component relies on another for proper functioning. |  |
| Port | Shown as a small square on the edge of a component. Represents an entry or exit point for data or control flow. |  |
| Connector | Depicted as a solid line between two components or ports, representing a communication path between them. |  |

*Table 21: Components of the Component Diagram*

*Figure 24: Component Diagram of the System*

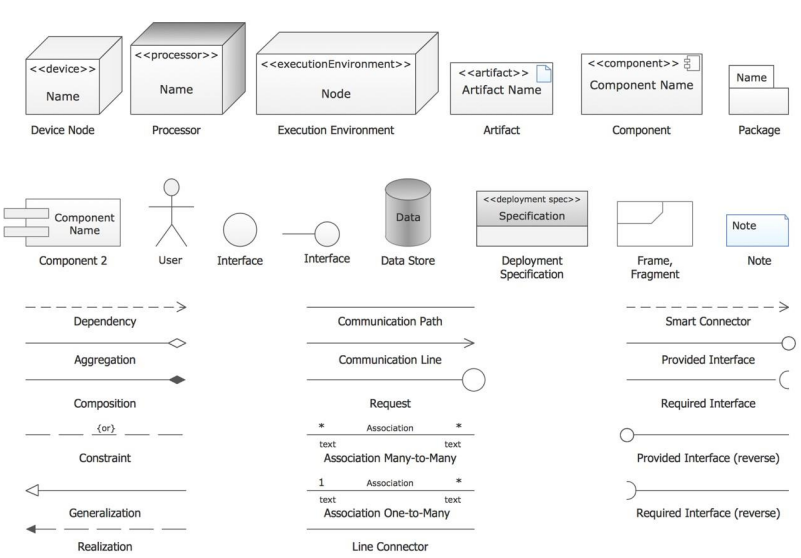


* 1. **DEPLOYMENT DIAGRAM**

**Definition**

A Deployment Diagram is a specialized type of UML diagram that focuses on a system’s physical nodes. It visually represents the deployment of artifacts and nodes, detailing the hardware components and how software components installed on them interact. Essentially, it illustrates how the system's software runs on the hardware and how these components communicate with one another.

**Formalism, Components of the Deployment Diagram**



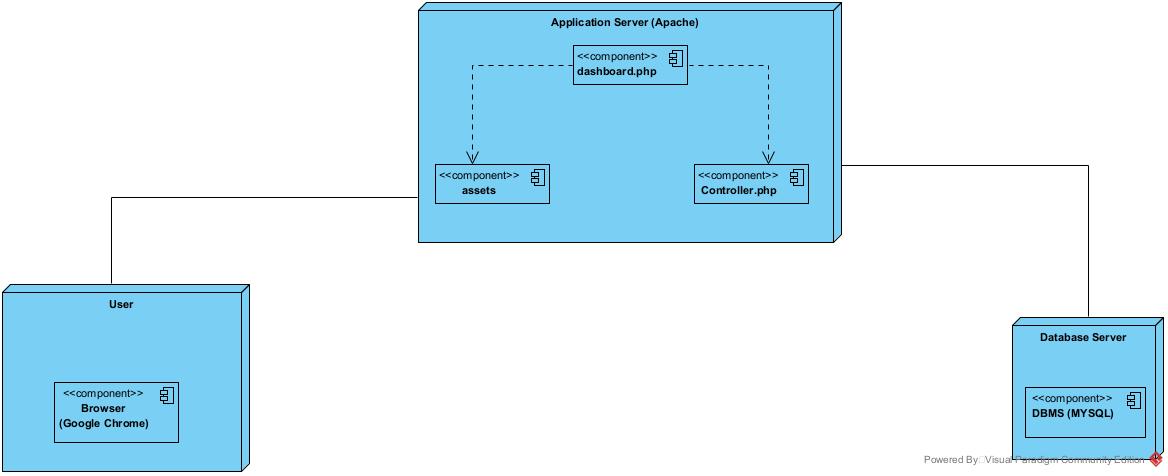
*Figure 25: Formalism, Components of the Component Diagram*

**Description of some components of the Deployment Diagram**

*Table 22: Description of some components of the Deployment Diagram*

|  |  |
| --- | --- |
| ELEMENTS | DESCRIPTION |
| Node | Represents physical hardware where software runs, depicted as a three-dimensional box. |
| Artifact | Represents a tangible piece of software residing on a node, such as executables, libraries, archives, database tables, or configuration files. |
| Communication path | Illustrates the flow of information between nodes, showing how they interact within the system. |
| Dependency | Depicted as a reliance of one element (node or artifact) on another for functionality or deployment. |
| Association | Represents a relationship between two nodes, indicating communication or connectivity between them. |
| Component | Although not exclusive to deployment diagrams, components represent functional parts of a software system. In deployment diagrams, they show high-level software deployed on nodes. |
| Package | A logical grouping of related elements, such as nodes or artifacts, to provide better clarity within the deployment diagram. |

*Figure 26: Deploment Diagram of the System*



* 1. **PRESENTATION OF TOOL USED**

|  |  |  |
| --- | --- | --- |
| SOFTWARE | USAGE | LOGO |
| Microsoft 365 | Used for the creation our report and PowerPoint |  |
| Visual studio code | Open and free software for the coding part of the project |  |
| PowerAMC | For doing the design of the diagrams to represent the structure and the interaction between the different actors |  |
| Google | For doing the research on the internet and testing of the application |  |
| Windows 11, version 23H2 | Operating system |  |
| Gantt Project | For building a gantt chart for the planning of the work |  |
| XAMPP | For creating database |  |
| Visual Paradigm Enterprise version | For doing the design of the diagrams to represent the structure and the interaction between the different actors |  |

1. SOFTWARE RESOURCES

*Table 23: Software Resources Used*

1. HARDWARE RESOURCES

*Table 24: Software Resources Used*

|  |  |
| --- | --- |
| MATERIAL | USAGE |
| LAPTOP (LENOVO i5 8th Gen, SSD) | Used to build the system, type the report, create the power point, make research, etc. |
| Internet Modem | Used for internet connection |
| Samsung Tablet | Used for carrying research and storage |
| Printer HP Officejet 4630 | Used to print the scientific document |

* 1. **TECHNOLOGY STACK**

*Table 24: Technology stack*

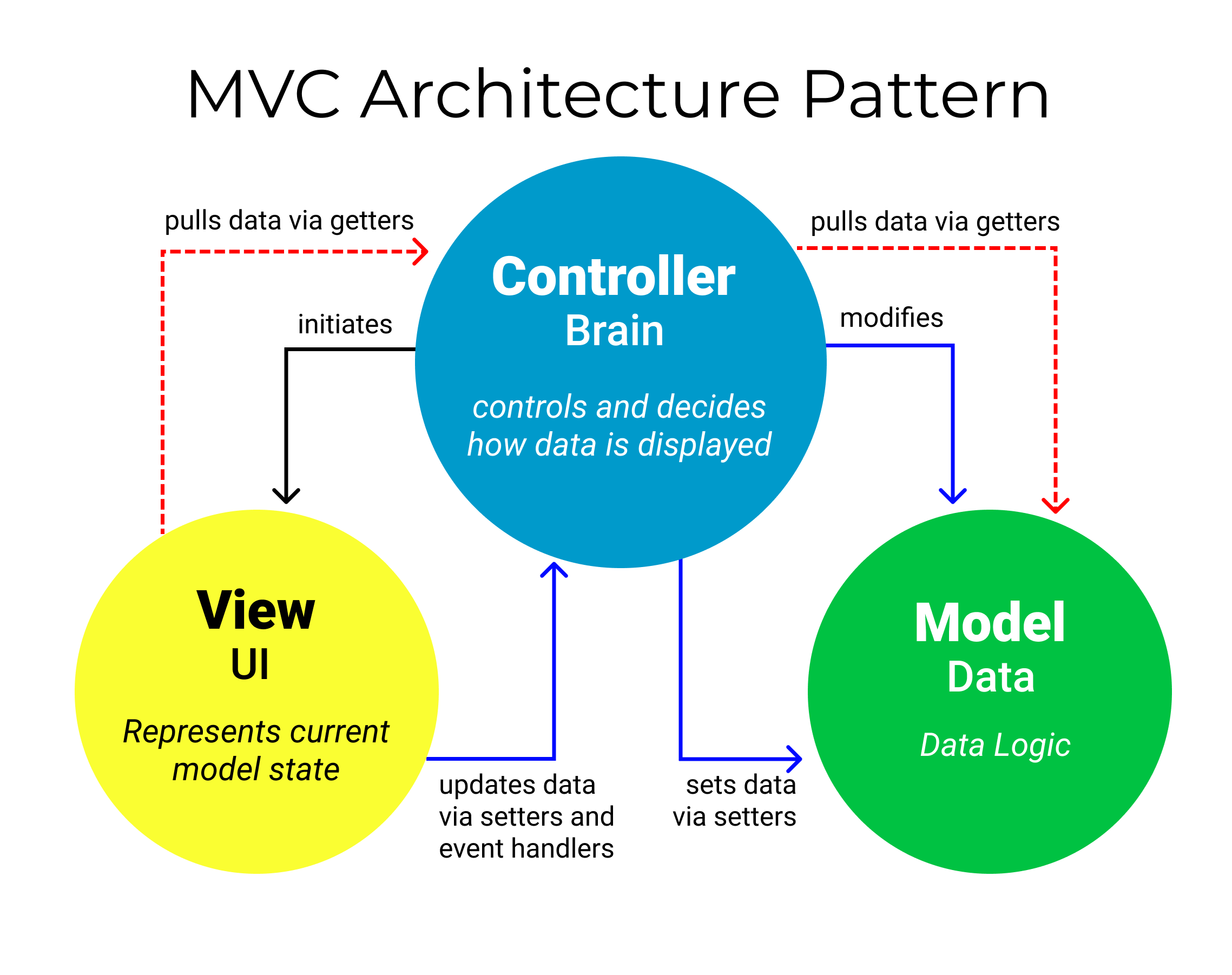
|  |  |
| --- | --- |
| Software | Roles |
| HTML/CSS (Bootstrap) | These are used for building and styling the front-end of the restaurant app, ensuring a responsive and user-friendly interface. **Bootstrap** provides pre-designed UI components for faster front-end development. |
| PHP | PHP is used as the server-side scripting language for processing user inputs, handling orders, and managing the communication between the front-end and the database. |
| MySQL | A relational database management system used to store and retrieve restaurant data, such as menus, user profiles, orders, and transaction records. |
| UML | Unified Modeling Language (UML) is employed to model and visualize system diagrams during the analysis and design phases of the restaurant app development. |
| PostgreSQL | The DBMS of my application to store and retriev e |
| |  | | --- | | **JavaScript** |  |  | | --- | |  | | Enhances interactivity on the front end, like form validation, dynamic content updates, and handling asynchronous requests. |
| Ajax | Used to make asynchronous calls between the client (front-end) and server (back-end) to update parts of the webpage without refreshing the entire page. |
| Apache | A web server that hosts the restaurant app, handling HTTP requests and serving web pages to users. |

* 1. **ARCHITECCTURE OF THE APPLICATION** 
     1. LOGICAL ARCHITECTURE

The logical architecture defines the overall structure of the software system, including the design pattern used and how different components of the system interact.

**Design Pattern: MVC (Model-View-Controller)**

* **Model:**  
  This layer handles the database and data-related logic. It communicates with the database to fetch, save, or modify restaurant data (such as menu items, user profiles, and order history).
* **View:**  
  The view layer is responsible for the presentation and user interface. The front-end will be developed using **HTML, CSS (Bootstrap)**, and **JavaScript** to create responsive layouts that display the restaurant's menu, handle ordering, and display user profiles.
* **Controller:**  
  The controller acts as the middleman between the Model and View. It processes incoming requests (like making an order or updating a user profile) and sends the appropriate data to the Model. It also selects the correct View to display the data to the user.



* + 1. PHYSICAL DATA MODEL

A **Physical Data Model (PDM)** represents the implementation of a data model on a physical storage medium like a database. It includes all the details necessary for implementing a system, such as table structures, column data types, indexes, relationships, and constraints. The **PDM** focuses on how data is stored, optimized for performance, and secured. It transforms the logical design (ERD or class diagrams) into a schema that can be implemented in a relational database.

**Elements of a Physical Data Model:**

1. **Tables**: Physical realization of entities.
2. **Columns**: Attributes in tables with data types.
3. **Primary Keys (PK)**: Unique identifiers for each record.
4. **Foreign Keys (FK)**: Fields that link to primary keys in other tables to maintain relationships.
5. **Indexes**: Optimized data retrieval mechanism.
6. **Constraints**: Rules to enforce data integrity (e.g., NOT NULL, UNIQUE, CHECK).
7. **Data Types**: Defines the type of data (e.g., INT, VARCHAR, DATE).
8. **Triggers and Stored Procedures**: Automated actions on data changes.
9. **Relationships**: Links between tables, typically handled via primary and foreign keys.

**CONCLUSION**

In this phase, we completed the UML diagrams and introduced the architecture on which our application is built. We also outlined the tools and programming language used in its development. Now, we are moving forward to conduct functionality testing for the application.

**PART VII:**

**USER GUIDE**

Preamble

In today’s fast-paced environment, convenience and efficiency are paramount. Our restaurant app is designed to enhance your dining experience by offering seamless access to essential services, from browsing menus to placing orders. Whether you're a first-time user exploring our platform or a returning customer looking for specific features, this guide will walk you through every step. With clear, user-friendly instructions, our aim is to ensure that your interaction with the app is smooth and hassle-free, helping you fully leverage its features to make dining easier and more enjoyable.

Content

INTRODUCTION

1. DEPLOYMENT OF THE APPLICATION
2. SHOW CASE

CONCLUSION

**INTRODUCTION**

In this final phase of our report, we will provide a detailed walkthrough of the system’s requirements, installation process, and how to access and use its features. Each section will be presented in a step-by-step manner, designed to simplify the setup for first-time users and ensure a smooth onboarding experience with the platform.

* + 1. **DEPLOYMENT OF THE APPLICATION**

After developing the application, we have to deploy the application for usage. The deployment phase of the software is a phase in which the application is hosted on a server so that it can be available on the clients working area. The deployment of the software will need the following files and environments.

1. Necessary Environments
   1. Google Chrome (or any other web browser) must be installed
   2. One must have an internet Connection in order to install libraries.
2. Installation of Working Environments
   * DATABASE SERVER INSTALLATION

To deploy a software, we need to install it database server, in our case XAMPP will be installed in our system which is windows.

*Figure 27: Xampp Icon*



STEP 1:

* + - 1. Open your browser and go to the official XAMPP website:  
         https://www.apachefriends.org/index.html
      2. On the website, you will find different versions of XAMPP for Windows, Linux, and macOS. Click the **"Download"** button for your operating system.

STEP 2:

1. Once the download is complete, **navigate** to your Downloads folder or the location where you saved the installer.
2. Double-click the **installer file** (e.g., xampp-win32-x.x.x-VC15-installer.exe for Windows).
3. If you get a **User Account Control (UAC)** prompt, click **Yes** to allow the installer to run.

STEP 3:

1. The XAMPP Setup Wizard will open. Click **Next** to continue.
2. You will be asked to choose which components you want to install. By default, all components (Apache, MySQL/MariaDB, PHP, phpMyAdmin, etc.) are selected.

* If you don't need specific components, you can uncheck them (e.g., if you don't need Perl, you can uncheck it).
* **For beginners**, it's best to leave everything checked.

1. After choosing components, click **Next**.

STEP 4:

1. The wizard will prompt you to choose the folder where you want to install XAMPP. The default installation directory is usually C:\xampp.
2. You can choose a different directory if you prefer, but the default directory works well for most users.
3. Click **Next** to continue.
4. Now, you are ready to install XAMPP. Click **Next** and the installation process will begin.
5. Wait for the installation to complete. It may take a few minutes, depending on your system.
6. Once the installation is complete, click **Finish**.
7. XAMPP will ask if you want to start the **XAMPP Control Panel**. Keep the checkbox checked and click **Finish**.

* WEB APPLICATION INSTALLATION

To be able to use Servesoft we application the link would be shared.

* 1. **SHOWCASES**

GENERAL CONCLUSION

As we conclude our project and internship at N-tec labs, we reflect on a transformative journey filled with learning and growth. Our restaurant pre-order and order app project was driven by the goal to enhance the dining experience through a seamless digital platform. By leveraging modern technologies such as PHP, HTML, Bootstrap, and CSS, we’ve streamlined processes from ordering to management, ensuring both users and restaurant staff benefit from an efficient and intuitive system. This project has deepened our understanding of teamwork, innovation, and user-centric design, and we remain committed to continuously improving its impact.

BIBLOGRAPHY

* Mr. MESSIO UML course AICS-Cameroon 2022-2023
* OMG, OMG Unified Modelling Language 2.5, Object Management Group, March 2015, 794 pages.
* Mr. AGBOR PHP Course AICS-Cameroon 2023-2024
* Mr. AGBOR HTML course AICS-Cameroon 2022-2023
* Mr. MESSIO MySQL course AICS-Cameroon 2022-2023

WEBOGRAPHY

Contents

**DEDICATION ........................................................................................................................... i**

**ACKNOWLEDGEMENT ....................................................................................................... ii**

**CONTENT ............................................................................................................................... iii**

**LIST OF FIGURES .................................................................................................................. v**

**LIST OF TABLES ................................................................................................................. viii**

**GLOSSARY ............................................................................................................................. ix**

**ABSTRACT .............................................................................................................................. x**

**RESUME .................................................................................................................................. xi**

**GENERAL INTRODUCTION ............................................................................................... 1**

**PART I ....................................................................................................................................... 2**

**INSERTION PHASE ............................................................................................................... 2**

**PART II ................................................................................................................................... 13**

**EXISTING SYSTEM ............................................................................................................. 13**

**PART III .................................................................................................................................. 23**

**SPECIFICATION BOOK ..................................................................................................... 23**

**PART IV .................................................................................................................................. 40**

**ANALYSIS BOOK ................................................................................................................. 40**

**PART V .................................................................................................................................... 76**

**CONCEPTION PHASE ........................................................................................................ 76**

**PART VI .................................................................................................................................. 96**

**REALIZATION PHASE ....................................................................................................... 96**

**PART VII............................................................................................................................... 107**

**TEST OF FUNCTIONALITIES ......................................................................................... 107**

**PART VIII ............................................................................................................................. 115**

**INSTALLATION GUIDE AND USER GUIDE ................................................................ 115**

**GENERAL CONCLUSION ................................................................................................ 135**

**ANNEXES ............................................................................................................................... xii**

**BIBLIOGRAPHY ................................................................................................................. xiii**

**WEBOGRAPHY ................................................................................................................... xiv**

**TABLE OF CONTENT ......................................................................................................... xv**