

Table Of Contents

[**Dedication** 3](#_Toc136975824)

[**Acknowledgments** 4](#_Toc136975825)

[**Introduction** 5](#_Toc136975826)

[**Company presentation** 6](#_Toc136975827)

[**Chapter One: The Conceptual aspect.** 7](#_Toc136975828)

[**I.** **Specifications document:** 8](#_Toc136975829)

[ **Problematic:** 8](#_Toc136975830)

[ **Objective:** 8](#_Toc136975831)

[ **What is ecommerce?** 8](#_Toc136975832)

[ **Functional Needs:** 9](#_Toc136975833)

[ **Client Needs:** 9](#_Toc136975834)

[ **Non-functional Needs:** 10](#_Toc136975835)

[ **Technical Requirements** 10](#_Toc136975836)

[**II.** **Project management methodology:** 11](#_Toc136975837)

[ **Introduction:** 11](#_Toc136975838)

[ **Choosing the project management method:** 11](#_Toc136975839)

[ **Principle of SCRUM:** 12](#_Toc136975840)

[ **Product Backlog:** 12](#_Toc136975841)

[ **Sprint Planning:** 13](#_Toc136975842)

[ **Sprint Execution:** 13](#_Toc136975843)

[ **Daily Scrum:** 13](#_Toc136975844)

[ **Presentation of Tools:** 14](#_Toc136975845)

[ **User stories:** 15](#_Toc136975846)

[ **Sprints/Tasks:** 17](#_Toc136975847)

[**III.** **Gantt Chart:** 18](#_Toc136975848)

[**IV.** **UML Diagrams:** 21](#_Toc136975849)

[ **Introduction:** 21](#_Toc136975850)

[ **Class Diagram:** 22](#_Toc136975851)

[ **Sequence Diagram:** 23](#_Toc136975852)

[ **Use Case Diagram:** 24](#_Toc136975853)

[ **Data dictionary:** 25](#_Toc136975854)

[**Chapter Two: Implementation** 27](#_Toc136975855)

[**I.** **Technologies used:** 28](#_Toc136975856)

[ **Front end:** 28](#_Toc136975857)

[ **Back end :** 29](#_Toc136975858)

[**II.** **Benchmarking:** 31](#_Toc136975859)

[ **Why use react js and express js:** 31](#_Toc136975860)

[ **MongodB VS MySQL:** 33](#_Toc136975861)

[ **Braintree vs stripe:** 34](#_Toc136975862)

[**Chapter Three: User Manuel** 36](#_Toc136975863)

[**I.** **WIREFRAMES:** 37](#_Toc136975864)

[**Home Page:** 37](#_Toc136975865)

[**User Dashboard:** 38](#_Toc136975866)

[**Cart Page:** 38](#_Toc136975867)

[II. **Navigation diagram:** 39](#_Toc136975868)

[**III.** **Final Pages:** 40](#_Toc136975869)

[**Chapter Four: Technical specifications for the administrator** 44](#_Toc136975870)

[**I.** **Runtime Environment:** 45](#_Toc136975871)

[ **the specific requirements of the runtime environment:** 45](#_Toc136975872)

[**Conclusion** 47](#_Toc136975873)

[**Bibliography** 48](#_Toc136975874)

[**Abbreviations** 49](#_Toc136975875)

# **Dedication**

To my dear parents

Nothing can express the degree of love and affection I feel for you.

You have never stopped supporting and encouraging me during all the years of my studies, you have always supported me.

May God give you health, happiness, and a long life.

To my dear friends and colleagues,

especially our friend Salma Achach,

Thank you for the good times we spent together,

I am incredibly grateful to have a second family.

Testifies to my great love,

My sincere thanks and infinite gratitude.

# **Acknowledgments**

At the conclusion of this internship, we would like to express our sincere gratitude to ISMO Tetouan for providing us with the opportunity to undertake an exceptional internship experience.

We would also like to extend our heartfelt appreciation to the management of HIBARIJ s.a.r.l for welcoming us into their company and for their invaluable support throughout our internship.

We are particularly grateful to Mr. Hamid Choclate, our manager at the company, for his unwavering encouragement and assistance, despite his numerous responsibilities and commitments. His guidance has been instrumental in ensuring the successful execution of our internship.

We would also like to extend our deepest thanks to Mr. Rahmouni Oussama for his continuous support during the entire duration of our internship. His dynamic approach and expertise have contributed to the accomplishment of our work.

Lastly, we would like to express our appreciation to all the teachers at ISMO for the invaluable training and knowledge they imparted to us during our formative years.

We are profoundly grateful for the opportunity to have been a part of this enriching internship experience, and we are confident that the skills and experiences gained during this period will contribute to our professional growth and development.

# **Introduction**

This internship report provides an overview of our experience working on an e-commerce application development project at HIBARIJ s.a.r.l. The objective of the project was to create a user-friendly and convenient platform for online shopping, focusing on the retail sale of telephone equipment, devices, and accessories. The report will delve into various aspects of the project, including analysis, implementation, a user manual, and technical specifications for the administrator.

In the analysis phase, we provided a solid foundation for the subsequent development process. The implementation phase was a critical stage where we put our coding skills into practice. Using the selected technologies and programming languages, we worked on developing different modules and integrating them into a cohesive e-commerce platform. Throughout this phase, we encountered technical challenges and applied problem-solving techniques to overcome them, gaining valuable insights into software development best practices.

Throughout the internship, we gained valuable insights into project management, teamwork, and the practical application of coding languages and technologies. The experience provided us with a deeper understanding of the e-commerce industry, software development processes, and the importance of user-centric design.

In the subsequent sections of this report, each part will be explored in detail, providing an in-depth analysis of the project, its implementation, the user manual, and the technical specifications for the administrator.

# **Company presentation**

HIBARIJ s.a.r.l, a small yet dedicated company, specializes in the retail sale of telephone equipment, devices, and accessories. Despite its modest size, HIBARIJ is passionate about providing customers with high-quality products and personalized service. With a strong focus on customer satisfaction, the company goes the extra mile to understand the unique needs and preferences of each individual. HIBARIJ takes pride in offering a carefully curated selection of telephone equipment, ensuring that every item meets their stringent quality standards. Their product range includes smartphones, landline phones, accessories, and essential peripherals, all sourced from trusted suppliers. By maintaining strong relationships with suppliers, HIBARIJ can offer competitive pricing without compromising on quality. With a small team of dedicated professionals, the company operates efficiently, delivering a personalized and attentive shopping experience to every customer. HIBARIJ's commitment to excellence and customer-centric approach sets them apart in the industry, earning them a loyal customer base and recognition for their reliable and trustworthy services.

|  |  |
| --- | --- |
| Address | Avenue Achorafaa, Residence Al Amane, No. 1, Tetouan. |
| Legal Form | Limited Liability Company |
| Capital | 10 000 MAD |
| Sales figures | 12,587 MAD |
| No. of employees at location | About 4 |

# 

# **Chapter One: The Conceptual aspect.**

## **Specifications document:**

This specification document outlines the key functionalities, performance, security, and technical requirements of the e-commerce mobile application. It will serve as a reference for the application's development, ensuring that user needs and expectations are considered in the design and implementation process.

### **Problematic:**

With the ongoing digitalization of various aspects of our lives, HIBARIJ recognized the need to adapt to the changing market dynamics and customer preferences. By embracing digitalization and developing the e-commerce application.

### **Objective:**

The objective is to develop an e-commerce application to provide users with a user-friendly and convenient platform for online shopping.

### **What is ecommerce?**

E-commerce (electronic commerce) is the activity of electronically buying or selling of products on online services or over the Internet. E-commerce draws on technologies such as mobile commerce, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. E-commerce is in turn driven by the technological advances of the semiconductor industry and is the largest sector of the electronics industry.

### **Functional Needs:**

* User Registration and Authentication: Users should be able to create accounts, log in, and manage their profiles securely.
* Product Catalog Management: The application should support the addition, removal, and modification of product listings, including categories, descriptions, prices, and availability.
* Shopping Cart Functionality: Users should be able to add products to a cart, view the cart, modify quantities, and proceed to checkout.
* Order Management: The application should handle order placement, order tracking, and notifications to keep users updated on the status of their orders.
* Payment Integration: Secure integration with popular online payment gateways to enable seamless and safe transactions.
* User Account Management: Users should be able to update their personal information, manage addresses, and view order history.

### **Client Needs:**

* Increase online sales and revenue by providing a seamless shopping experience.
* Attract and retain customers through a user-friendly and visually appealing interface.
* Ensure secure and reliable online transactions to build trust with customers.
* Streamline the order fulfillment and delivery process to enhance customer satisfaction.
* Collect and analyze customer data to personalize the shopping experience and improve marketing strategies.

### **Non-functional Needs:**

* Performance: The application should load quickly, respond promptly to user actions, and handle many concurrent users.
* Scalability: The application should be able to accommodate future growth and handle increased traffic and transactions without performance degradation.
* Security: Robust security measures should be implemented, including data encryption, secure payment processing, and protection against unauthorized access.
* Usability: The user interface should be intuitive, consistent, and accessible, ensuring a positive user experience for customers of all technical backgrounds.
* Reliability: The application should be reliable, with minimal downtime, and able to recover from failures gracefully.
* Compatibility: The application should be compatible with different screen sizes to reach a wide user base.

### **Technical Requirements**

* The application requires an active internet connection.
* Integration with secure online payment gateways (e.g., PayPal, Stripe).
* Storage of user and product data in a secure database.
* Integration with delivery services to track orders in real-time.

## **Project management methodology:**

### **Introduction:**

This chapter describes the project management approach. It first presents the choice of methodology, then the principle of this approach, the roles of SCRUM, and finally the project organization.

### **Choosing the project management method:**

SCRUM is the most widely used Agile methodology among other Agile methods. And in fact, it is the most proven one.

Moreover, SCRUM is an iterative and incremental process that represents an agile software development framework for managing product development.

It defines "a flexible approach, a holistic product development strategy, and enables development teams to organize themselves as a unit to achieve a common goal."

One of the distinctive features of SCRUM is that during product development, clients can change their minds about what they want and need (often referred to as requirements volatility).

### **Principle of SCRUM:**

SCRUM is an agile methodology dedicated to project management. This management method aims to improve the productivity of the team.

The SCRUM methodology involves progressing the project through the implementation of series of "sprints". At the start of each sprint, a planning meeting is held so that each team member can commit to the number of tasks they can execute, as well as the creation of the "sprint backlog," which is the comprehensive list of tasks to be accomplished during the sprint.

Every day during the sprint, all team members (along with the product owner and the SCRUM Master) must attend the daily SCRUM meeting. This meeting should not exceed 15 minutes and allows team members to share what they accomplished the previous day, what they are working on that day, and identify any issues that may impede the progress of the sprint. This meeting serves to synchronize all team members.

The end of a sprint is marked by a debriefing session to present the completed work to the product owner and share any information that may influence the next sprint.

### **Product Backlog:**

Users create a product backlog, which consists of all prioritized feature requests. In the product backlog, items are stacked from top to bottom, with the top representing the most requested and urgent features that need to be addressed first.

### **Sprint Planning:**

As shown in the diagram, we select the highest-priority product backlog items (PBIs), which are the most requested by users, for the sprint planning. In the sprint planning, we break down the PBIs into tasks and estimate their durations in hours.

### **Sprint Execution:**

Once the sprint planning is done, meaning we have determined all the tasks to be accomplished during this iteration (sprint), we move on to the sprint execution phase. This is where the project team, including the development team, works on delivering increments of functionality.

### **Daily Scrum:**

Every day, we have a brief event called the daily scrum, during which we review what was developed the previous day, what will be worked on the following day, and how we will approach it.

This short event takes a little time and serves as a quick check-in on what happened the day before, what needs to happen today, and any obstacles or issues that may arise.

### **Presentation of Tools:**



Wondershare EdrawMax logo

Wondershare EdrawMax is a comprehensive diagramming software that allows users to create a wide range of diagrams, charts, and visual representations.

It offers a user-friendly interface and a rich set of tools and templates, making it suitable for both beginners and professionals.

With EdrawMax, users can create various types of diagrams, including flowcharts, organizational charts, mind maps, network diagrams, floor plans, and more. The software provides a drag-and-drop interface, allowing users to easily add and connect shapes, symbols, and text to build their diagrams.

One of the key features of EdrawMax is its extensive library of pre-designed templates and symbols. Users can choose from a wide range of templates and customize them to suit their specific needs. The software also offers a vast collection of symbols and icons that can be used to enhance the visual appeal of the diagrams.

EdrawMax supports collaboration and sharing capabilities, enabling multiple users to work on the same diagram simultaneously. Users can easily share their diagrams with others through various methods, including exporting to different file formats, printing, or sharing online.

In addition to diagram creation, EdrawMax provides advanced features like data visualization, presentation mode, and integration with other applications. It supports cross-platform compatibility, allowing users to access and edit their diagrams on different devices and operating systems.

Overall, Wondershare EdrawMax is a powerful and versatile diagramming tool that offers a comprehensive set of features for creating professional-looking diagrams for various purposes. Whether it's for business presentations, educational materials, or project planning, EdrawMax provides the tools and flexibility to bring ideas and concepts to life through visual representations.

### **User stories:**

These user stories represent the desired functionality and features from the perspective of different user roles in our e-commerce app. They help prioritize and guide the development process, ensuring that the app meets the needs and expectations of its users.

* As a customer, I want to be able to search for products based on keywords so that I can find the items I'm interested in quickly and easily.
* As a customer, I want to be able to filter products by category, price range so that I can narrow down my search and find products that meet my specific criteria.
* As a customer, I want to be able to view detailed product information, including images and descriptions so that I can make informed purchasing decisions.
* As a customer, I want to be able to add products to a shopping cart and keep track of the items I intend to purchase.
* As a customer, I want to be able to securely proceed through the checkout process, enter my shipping and billing information, and select a preferred payment method to complete my purchase.
* As a customer, I want to see order updates, including order confirmations and shipping tracking information, so that I can stay informed about the status of my purchases.
* As a customer, I want to be able to create and manage a user account, including updating my personal information, viewing order history, and saving payment preferences, for a convenient and personalized shopping experience.
* As an administrator, I want to be able to manage product inventory, including adding new products, updating stock quantities, and marking products as out of stock.
* As an administrator, I want to be able to manage customer orders, including processing payments, updating order status, and generating invoices or shipping labels.
* As an administrator, I want to be able to generate reports on sales, customer activity, and product performance to gain insights and make data-driven decisions for the business.

### **Sprints/Tasks:**

we started the development in a segmented manner. In other words, we divided the implementation of the application into sprints with deliverables, so that we could have a complete project to deliver in the end. Therefore, we divided the project into five sprints:

1. User Registration and Login:

* Task 1: Implement user registration functionality.
* Task 2: Implement user login functionality.
* Task 3: Create basic front-end layouts for registration and product listing pages.

1. Product Listing and Search:

* Task 4: Develop the product listing page with basic search and filtering capabilities.
* Task 5: Set up a database to store user information and product data.

1. Shopping Cart and Checkout:

* Task 6: Build the shopping cart feature, allowing users to add products.
* Task 7: Implement basic validation and error handling in the cart and checkout flow.
* Task 8: Develop the checkout process, including shipping address and payment information.
* Task 9: Integrate with a payment gateway to manage secure transactions.

1. Order History and Tracking:

* Task 10: Enable admin to view user order history and change the status of their orders.
* Task 11: Enable users to view their order history and track the status of their orders.

1. Optimization and Refinement

* Task 12: Fix reported bugs and usability issues.
* Task 13: Improve app performance.
* Task 14: Make the page responsive on different devices.

## **Gantt Chart:**

The Gantt chart, commonly used in project management, is one of the most effective tools for visually representing the progress of different activities (tasks) that make up a project.

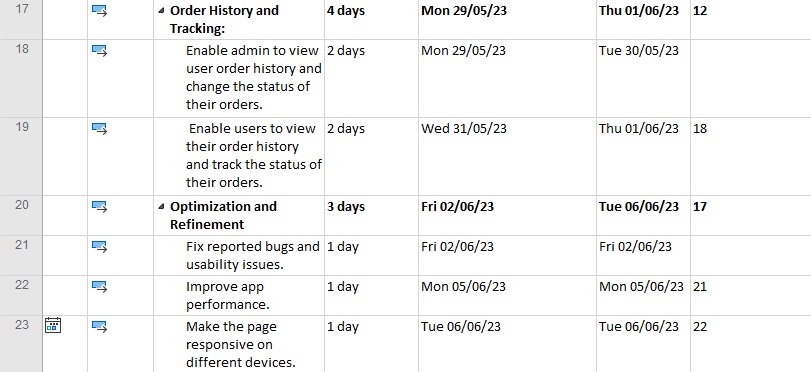
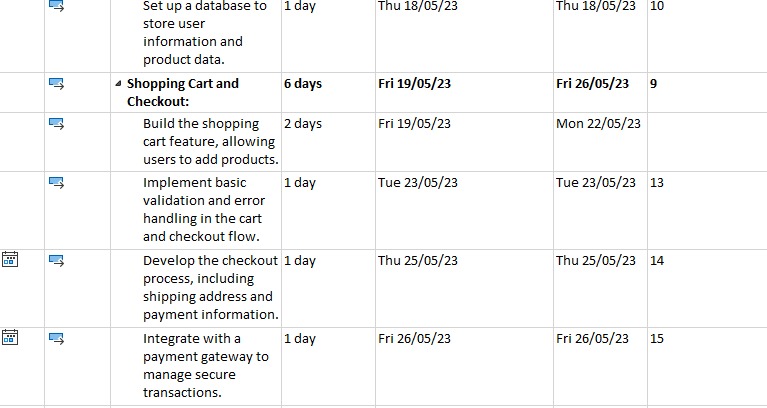
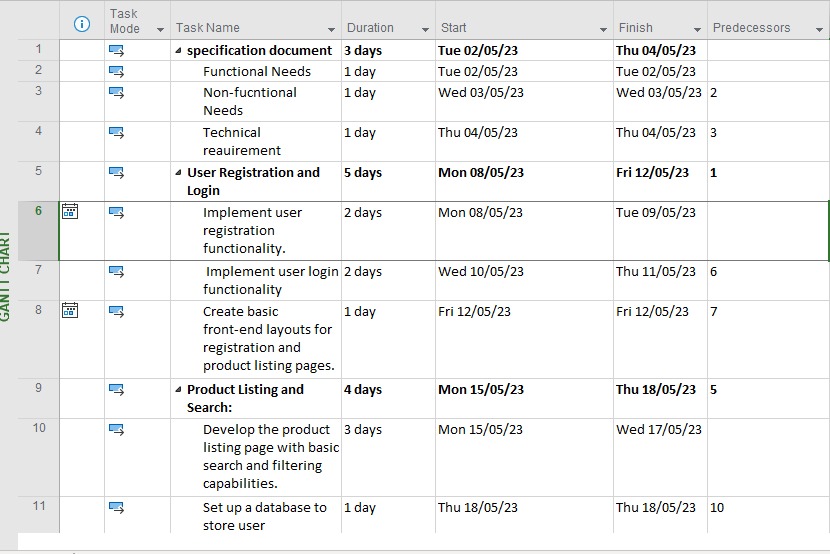
This chart allows for a quick visual overview of:

• The different tasks to be considered

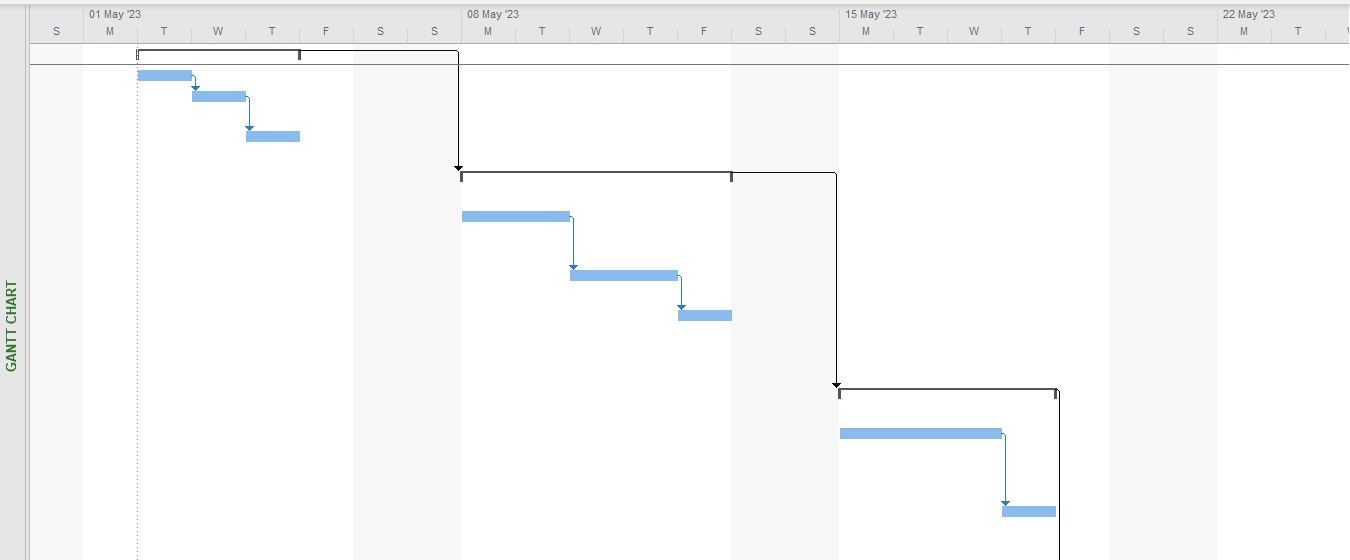
• The start date and end date of each task

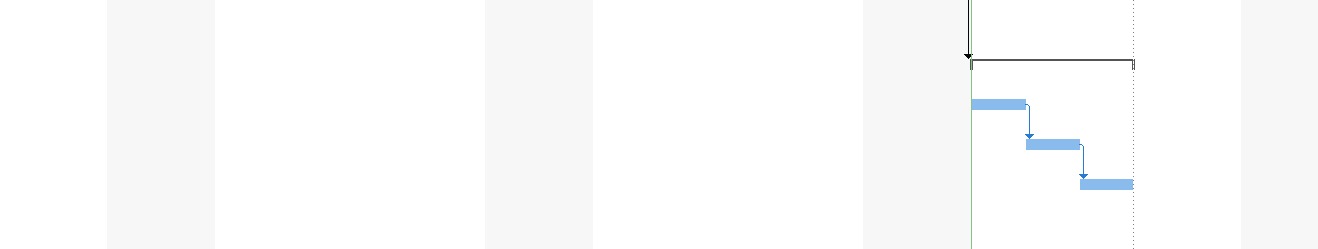
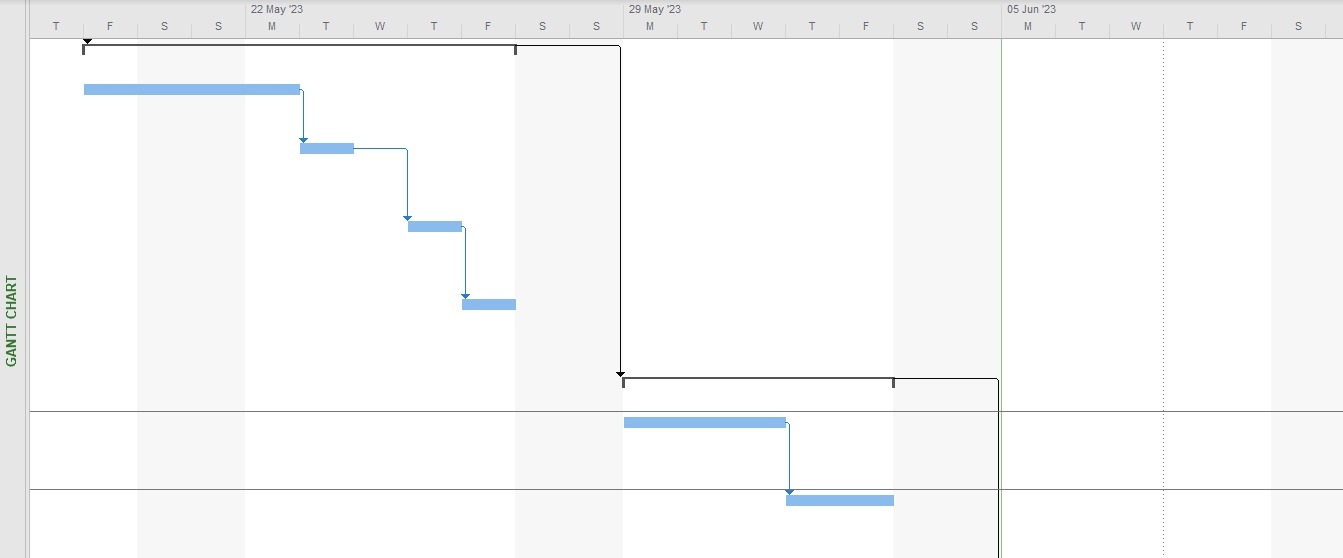
• The expected duration of each task

• Any potential overlap of tasks and the duration of that overlap

• The overall start date and end date of the project as a whole.

Gantt chart





Gantt chart

## **UML Diagrams:**

### **Introduction:**

In this section, we will analyze and model the client's requirements using the UML language.

The analysis and design activity aims to translate the functional requirements and constraints derived from the specifications and requirements specification document into a more professional and understandable language for all individuals involved in the development and use of the application.

1. Choice of UML:

UML stands for "Unified Modeling Language." It is a visual language consisting of a set of diagrams that provide different perspectives on the project at hand. UML provides us with diagrams to represent the software to be developed, including its functionality, deployment, and possible actions that can be performed by the software, among others.

1. Why Model?

Just as it is better to draw a house before building it, it is preferable to model a system before implementing it.

Modeling involves visually and graphically describing the needs, functional solutions, and technical aspects of the project.

We model in order to:

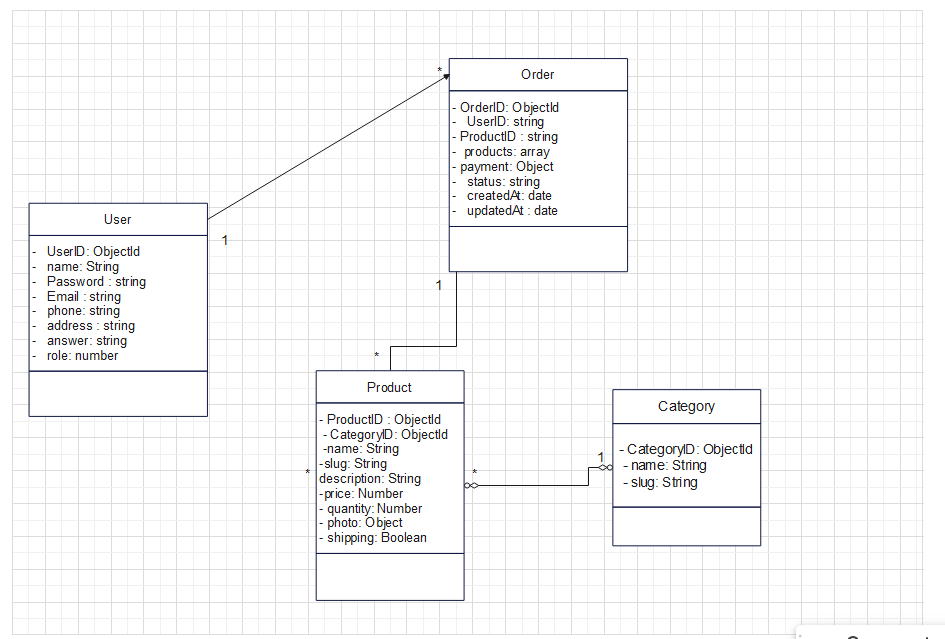
• Obtain a high-level modeling that is independent of specific programming languages and environments.

• Facilitate collaboration among participants from various backgrounds, working with a single synthesis document.

### **Class Diagram:**

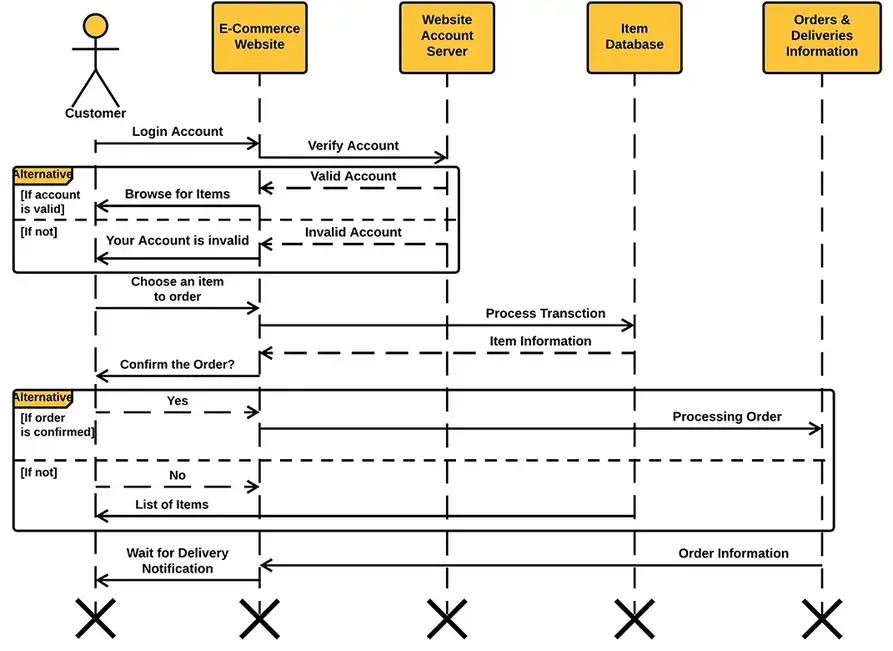
A class diagram is a type of UML diagram that represents the structure of a system by depicting the classes in the system, their attributes, methods, and relationships with other classes. It provides a static view of the system, emphasizing the organization and relationships among the classes.

In an ecommerce website context, a class diagram can be used to illustrate the various classes involved in the system, such as User, Product, Order, ShoppingCart, Payment, and more. Each class is represented as a box, showing its name, attributes, and methods. Relationships between classes, such as associations, aggregations, or inheritances, are depicted using lines and arrows connecting the relevant classes.

Class diagram

### **Sequence Diagram:**

A sequence diagram is a type of UML diagram that illustrates the interactions and sequence of messages exchanged between different objects or components in a system over a specific period. It represents the dynamic behavior of a system by showing the order of events and the flow of control between objects. In a sequence diagram, each object or component is represented as a vertical lifeline, and the messages exchanged between them are shown as arrows with the sequence number indicating the order of the messages. The diagram provides a visual representation of how objects collaborate and communicate to accomplish a specific task or scenario. Sequence diagrams are particularly useful for modeling and understanding the interactions between different parts of a system, identifying potential bottlenecks or inefficiencies, and validating the correctness of a system's behavior.

Sequence Diagram

### **Use Case Diagram:**

A use case diagram is a type of UML (Unified Modeling Language) diagram that represents the interactions between system actors (such as users or other systems) and the functionalities or use cases of a system. It provides a high-level view of the system's requirements and the actors' roles in interacting with the system.

In a use case diagram, use cases are represented as ovals, and actors are depicted as stick figures or other symbols. The use cases are connected to the actors through lines, indicating the interactions or relationships between them. The diagram helps in visualizing the overall functionality of the system and identifying the various possible scenarios or interactions between the actors and the system.



Use Case Diagram

### **Data dictionary:**

A data dictionary is a centralized repository or document that provides a comprehensive description and documentation of the data elements, attributes, and structures used in a database or information system. It serves as a reference guide for understanding the data within the system and helps ensure consistency, accuracy, and effective management of the data.

The purpose of a data dictionary is to provide clear and concise information about each data element, including its name, definition, data type, length, validations, relationships, and other relevant metadata. It acts as a catalog of data elements and their characteristics, enabling users, developers, and administrators to easily understand and work with the data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Data Format | Field Size | Description |
| User ID | ObjectId |  | 10 |  |
| User name | string |  | 20 |  |
| User Password | string |  | 25 |  |
| User Email | string |  | 60 |  |
| User phone | string |  | 10 |  |
| User address | string |  | 50 |  |
| User answer | string |  | 30 |  |
| User role | number |  | 1 |  |
| Order ID | ObjectId |  | 10 |  |
| Order products | array |  | 500 |  |
| Order status | string |  | 10 |  |
| Order payment | object |  | 80 |  |
| Order creation date | date | DD/MM/YYYY | 8 |  |
| Order updating date | date | DD/MM/YYYY | 8 |  |
| Product id | ObjectId |  | 10 |  |
| Product name | string |  | 50 |  |
| Product slug | string |  | 50 | unique and human-readable identifier for products |
| Product description | string |  | 100 |  |
| Product price | number |  | 6 |  |
| Product quantity | number |  | 20 |  |
| Product photo | object |  |  |  |
| Product shipping | boolean |  | 2 | Can the order be shipped or not |
| Category id | ObjectId |  | 10 |  |
| Category name | string |  | 50 |  |
| Category slug | string |  | 50 | unique and human-readable identifier for categories |

# 

# **Chapter Two: Implementation**

Our app offers a user-friendly interface with a comprehensive product catalog, effective search functionality, and secure checkout and payment options. It includes features like order tracking, account management features that allow users to personalize their shopping experience. Additionally, a responsive design which is crucial, ensuring that the app adapts seamlessly to different devices and screen sizes.

## **Technologies used:**

### **Front end:**

* React Hot Toast (version 2.4.1): React Hot Toast is a library for displaying toast notifications in React applications. It provides customizable and responsive toast messages to communicate with users.
* React Toastify (version 9.1.3): React Toastify is another library for displaying toast notifications in React applications. It offers various configuration options and animations to show informative messages to users.
* Axios (version 1.4.0): Axios is a popular JavaScript library for making HTTP requests from the browser. It provides an easy-to-use API for sending asynchronous HTTP requests and handling responses.
* Ant Design (version 5.5.2): Ant Design is a popular UI library for React that provides a set of high-quality and customizable components following the principles of the Ant Design system.
* Emotion React (version 11.11.0) and Emotion Styled (version 11.11.0): Emotion is a library for CSS-in-JS styling in React applications. It allows you to write CSS styles directly in JavaScript and provides a set of utility functions and components for styling React components.
* MUI (Material-UI) (version 5.13.3): Material-UI is a widely used UI library for React that implements the Material Design principles. It offers a comprehensive set of components and styles to create visually appealing and responsive interfaces.
* Testing Library Jest-DOM (version 5.16.5), Testing Library React (version 13.4.0), Testing Library User-Event (version 13.5.0): These libraries are part of the Testing Library ecosystem and provide utilities and methods for testing React components, interacting with them, and asserting expected behavior.
* Web Vitals (version 2.1.4): Web Vitals is a library that helps measure and track key performance metrics of web pages, such as loading speed and interactivity. It provides insights into user experience and helps optimize website performance.
* React DOM (version 18.2.0): React DOM is the entry point for rendering React components into the actual DOM. It provides the necessary methods and interfaces to interact with the browser's document object model.

### **Back end :**

* Express.js (version 4.18.2): Express.js is a fast and minimalist web application framework for Node.js. It provides a robust set of features for building web servers and APIs, such as routing, middleware support, and template engine integration.
* Express Formidable (version 1.2.0): Express Formidable is a middleware for parsing form data in Express.js. It helps handle form submissions, file uploads, and other types of form data by integrating with the formidable library.
* Mongoose (version 7.2.1): Mongoose is an Object-Document Mapping (ODM) library for MongoDB and Node.js. It simplifies working with MongoDB by providing a schema-based solution for modeling application data, enforcing validation, and offering query building capabilities.
* Morgan (version 1.10.0): Morgan is a logging middleware for Express.js. It logs HTTP requests and responses, providing useful information such as request method, URL, response status, and response time. Morgan helps in debugging and monitoring the server's activity.
* Bcrypt (version 5.1.0): Bcrypt is a library for hashing and salting passwords in Node.js. It offers secure password hashing using the bcrypt algorithm, which is designed to be slow and resistant to brute-force attacks, enhancing the security of user authentication.
* Braintree (version 3.16.0): Braintree is a payment gateway that allows businesses to accept online payments. It provides a simple and secure way to integrate payment processing functionality into applications, supporting various payment methods and handling transactions securely.
* Colors (version 1.4.0): Colors is a library that provides easy-to-use methods for adding colors and styles to the console output in Node.js. It allows developers to format and customize the terminal output, making it more visually appealing and easier to read.
* Concurrently (version 8.0.1): Concurrently is a utility for running multiple commands concurrently in Node.js. It is commonly used during development to run the front-end and back-end servers simultaneously, simplifying the setup and workflow for full-stack applications.
* CORS (version 2.8.5): CORS (Cross-Origin Resource Sharing) is a mechanism that allows web browsers to make requests to a different domain than the one the website originates from. The CORS library provides middleware for Express.js to handle and configure CORS headers, enabling cross-origin requests.
* Dotenv (version 16.0.3): Dotenv is a library for loading environment variables from a .env file into Node.js applications. It simplifies the process of managing sensitive configuration settings, such as API keys or database credentials, by keeping them separate from the codebase.
* Jsonwebtoken (version 9.0.0): Jsonwebtoken is a library for generating and verifying JSON Web Tokens (JWTs) in Node.js. JWTs are used for securely transmitting information between parties as a compact and self-contained token, commonly used for authentication and authorization in web applications.
* Moment.js (version 2.29.4): Moment.js is a popular library for parsing, manipulating, and formatting dates and times in JavaScript. It provides an easy-to-use API for working with dates, handling tiezones, calculating durations, and performing various date-related operations.
* Nodemon (version 2.0.22): Nodemon is a utility that monitors file changes in a Node.js application and automatically restarts the server. It is commonly used during development to improve productivity by eliminating the need to manually restart the server after each code change.
* Reactstrap (version 9.1.10): Reactstrap is a library that provides Bootstrap components as React.js components. It allows developers to easily incorporate pre-styled and responsive UI components into their React applications, speeding up the development process.
* Slugify (version 1.6.6): Slugify is a library for converting strings into URL-friendly slugs. It removes special characters, converts spaces to hyphens, and normalizes text to create clean and readable slugs, commonly used for generating SEO-friendly URLs or creating file names.

## **Benchmarking:**

### **Why use react js and express js:**

React.js and Express.js are popular choices for web development due to their unique features and benefits. Here are some reasons why we decided to use these technologies:

* + React.js:
* Component-Based Architecture: React.js follows a component-based architecture, allowing developers to build reusable UI components. This promotes code reusability, modular development, and simplifies the overall application structure.
* Virtual DOM: React.js uses a virtual DOM, which allows for efficient updates and rendering of components. It optimizes performance by minimizing the number of direct updates to the actual DOM, resulting in faster rendering and improved user experience.
* Rich Ecosystem: React.js has a vast ecosystem with a wide range of libraries, tools, and community support. This makes it easier to find solutions, plugins, and third-party integrations, speeding up the development process.
* One-Way Data Flow: React.js enforces a unidirectional data flow, making it easier to understand and debug the application's state changes. It helps in maintaining data consistency and predictability.
  + Express.js:
* Lightweight and Flexible: Express.js is a minimalistic and unopinionated framework for Node.js. It provides a lightweight and flexible foundation for building web applications, allowing developers to choose the specific components and libraries they need for their project.
* Easy Routing and Middleware: Express.js simplifies the routing process by providing an intuitive API for defining routes and handling HTTP requests. It also supports middleware, which enables developers to add custom functionality to the application's request-response cycle easily.
* Scalability: Express.js is designed to be scalable and performant. It allows developers to handle multiple concurrent requests efficiently, making it suitable for building robust and high-traffic applications.
* Large Community and Ecosystem: Express.js has a large and active community, which means there are plenty of resources, tutorials, and community-driven packages available. This makes it easier for developers to find support, learn new techniques, and extend the functionality of their Express.js applications.

### **MongodB VS MySQL:**

When comparing Mongoose (an ODM for MongoDB) and MySQL (a relational database management system), it's important to consider the context and specific requirements of the project. However, here are some points in favor of Mongoose that we took into consideration.

* + Advantages of MongoDB:
* Schema flexibility: MongoDB, which Mongoose interacts with, is a NoSQL database that offers schema flexibility. With Mongoose, you can define flexible schemas and easily adapt them as your application evolves. This can be beneficial when dealing with dynamic data structures or when your application requirements change frequently.
* Scalability: MongoDB is designed to scale horizontally, meaning it can handle large amounts of data and high traffic loads across multiple servers. Mongoose helps you leverage this scalability by providing easy integration with MongoDB's sharding and replica set features.
* Object-Document Mapping (ODM): Mongoose acts as an ODM for MongoDB, providing a layer of abstraction that simplifies working with the database. It allows you to define models, relationships, and queries using JavaScript, which can be more familiar and productive for developers already working with JavaScript-based stacks.
* JSON-like data storage: MongoDB stores data in a JSON-like format called BSON (Binary JSON). This makes it convenient to work with JSON-based data structures, especially when dealing with complex and nested data. Mongoose provides native support for mapping JavaScript objects to BSON, making it easy to work with JSON-like data in your application.
* Schema enforcement and validation: Mongoose allows you to define schemas with data types, required fields, and validation rules. This helps ensure data integrity and consistency by enforcing a predefined structure. It can be especially useful when working with applications that require strict data validation or have complex relationships between entities.
* Flexibility for Agile development: MongoDB's flexible schema and Mongoose's adaptability make them well-suited for agile development processes. They allow developers to iterate quickly, accommodate changing requirements, and easily handle data evolution over time.

### **Braintree vs stripe:**

Both Braintree and Stripe are popular payment processing platforms, each with its own set of features and advantages. However, here are some points in favor of Braintree that we took into consideration.

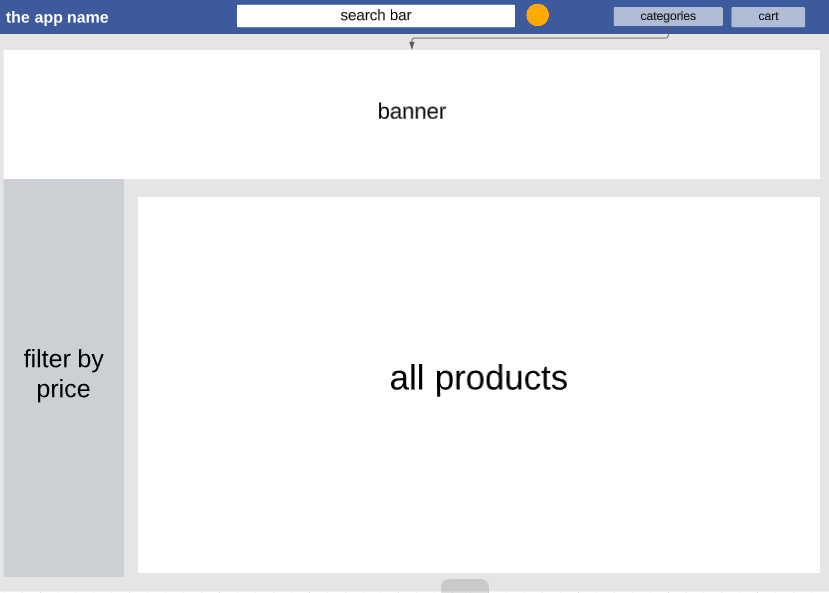
* + Advantages of Braintree:
* Integration Flexibility: Braintree offers a high level of customization and flexibility when integrating payment processing into your application. It provides robust client-side SDKs and server-side libraries for various programming languages, giving you control over the entire payment flow.
* Developer-Friendly: Braintree is known for its developer-friendly APIs and documentation. The platform provides clear and comprehensive documentation, code samples, and SDKs that make it easier for developers to integrate and work with their payment system.
* Global Reach: Braintree supports payments in more than 130 currencies and provides access to multiple payment methods, including credit and debit cards, digital wallets like PayPal and Venmo, and local payment methods. This global reach can be beneficial if you have customers from different regions.
* Advanced Fraud Protection: Braintree offers advanced fraud protection features, including built-in fraud detection tools, 3D Secure authentication, and risk assessment capabilities. These features help mitigate the risk of fraudulent transactions and enhance the security of your payment processing.
* Seamless Scalability: Braintree is designed to handle high volumes of transactions and is well-suited for businesses with scalability needs. It provides a reliable infrastructure that can accommodate growth and handle peak transaction loads without compromising performance or availability.
* Transparent Pricing: Braintree offers transparent pricing with flat-rate transaction fees, making it easier to understand and predict the cost of payment processing. This can be advantageous for businesses that prefer a simple and straightforward pricing structure.

# **Chapter Three: User Manuel**

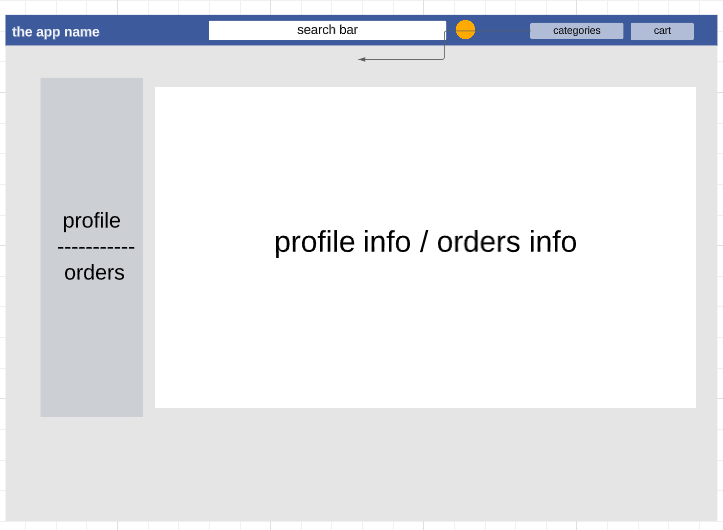
## **WIREFRAMES:**

Wireframes are visual representations or blueprints that outline the structure, layout, and functionality of a website, mobile app, or user interface. They are low-fidelity, simplified designs that focus on the placement of elements, navigation flow, and overall user experience rather than visual aesthetics. Wireframes act as a communication tool between designers, developers, and stakeholders, allowing them to visualize and discuss the skeletal framework of a digital product.

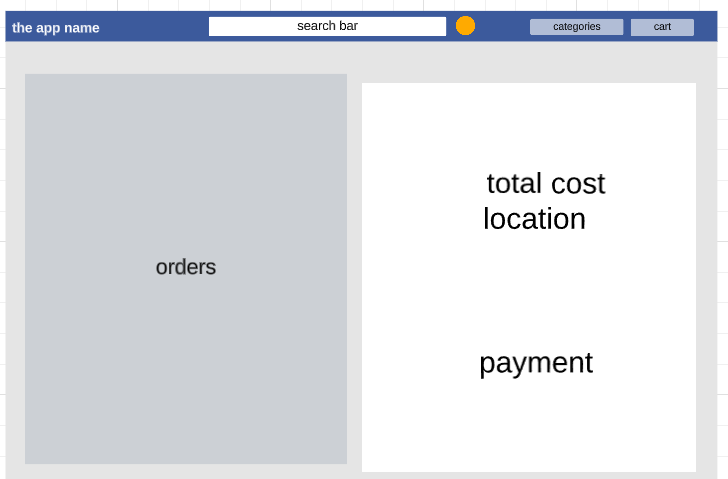
**Home Page:**



**User Dashboard:**



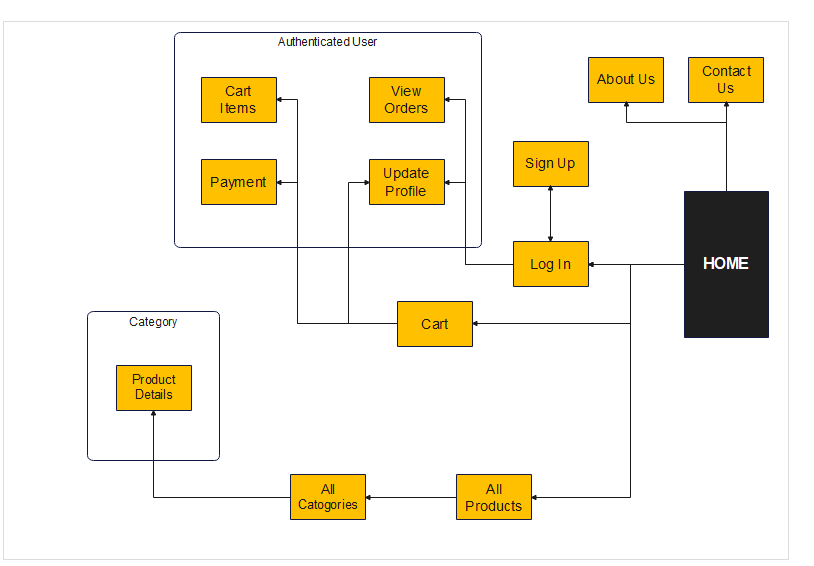
**Cart Page:**



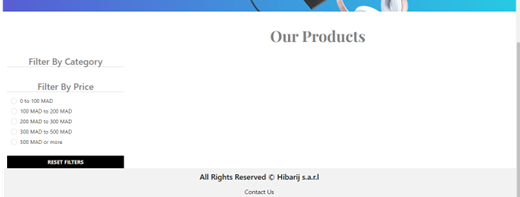
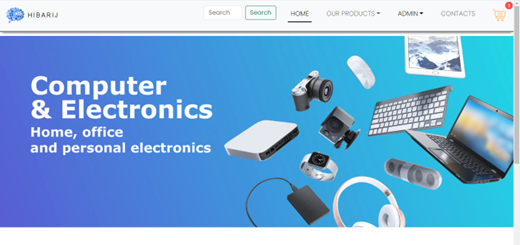
## **Navigation diagram:**

A navigation diagram is a visual representation that illustrates the flow and structure of navigation within a software application or system. It provides an overview of the different screens, pages, or components of the application and how they are interconnected.

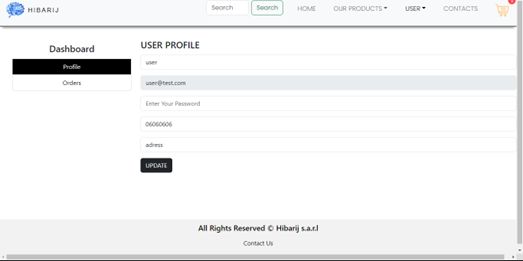
The primary purpose of a navigation diagram is to help designers, developers, and stakeholders understand the overall navigation flow and hierarchy of an application. It shows the relationship between various user interface elements, such as menus, buttons, links, and screens, and how users can navigate between them to accomplish different tasks or access different functionalities.

navigation diagram

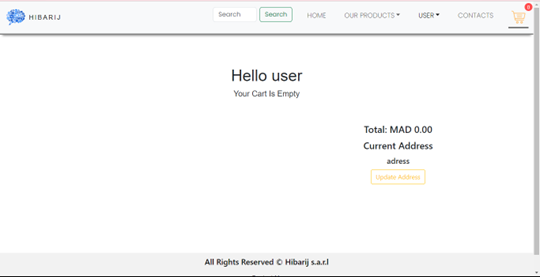
## **Final Pages:**



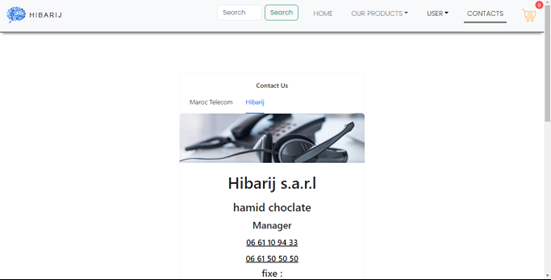
Home page



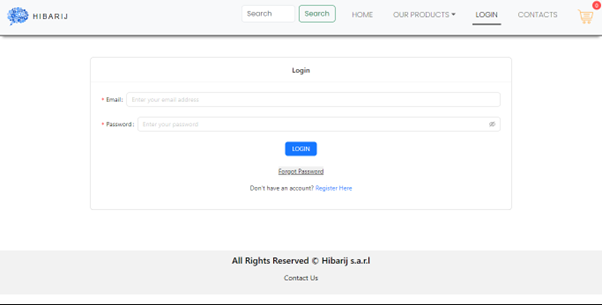
User dashboard



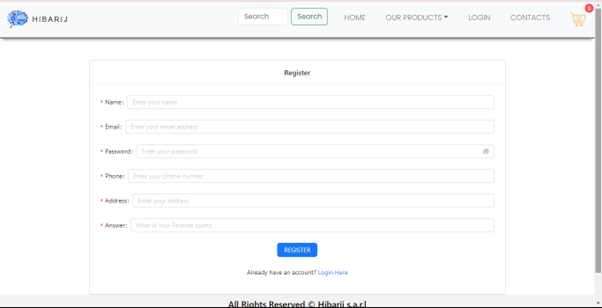
Cart page



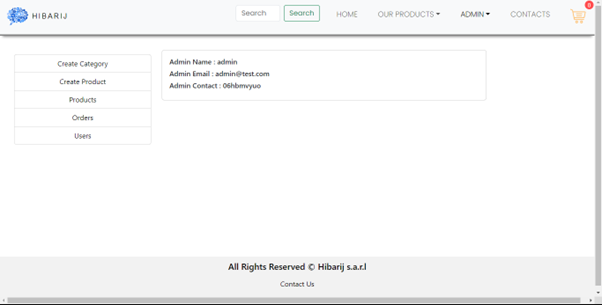
Contact us page



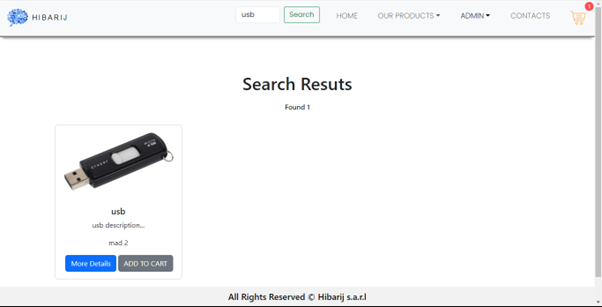
Login page



Register page



Admin dashboard



Search page

# **Chapter Four: Technical specifications for the administrator**

## **Runtime Environment:**

The runtime environment refers to the hardware and software components required to run the application or system. This includes servers, computers, operating systems, databases, software libraries, and other dependencies. The administrator needs to understand the specific requirements of the runtime environment to ensure proper configuration and optimal performance of the system. This involves considering factors such as resource allocation, compatibility, required software versions, and security requirements.

### **the specific requirements of the runtime environment:**

These are the general requirements for the runtime environment. (It's important to refer to the documentation and specific installation instructions for each technology to ensure all necessary dependencies and configurations are met).

* + Front-end:
* Node.js: Ensure that the latest stable version of Node.js is installed on the development and production environments. This will provide the runtime environment for running the front-end build process and executing JavaScript code on the server-side.
* Package Manager (npm or Yarn): Choose either npm or Yarn as the package manager for managing the front-end dependencies. Make sure to have the latest version of the selected package manager installed.
  + Back-end:
* Node.js: Same as the front-end, ensure that the latest stable version of Node.js is installed on the development and production environments. This will provide the runtime environment for running the Express.js server and executing JavaScript code on the server-side.
* MongoDB: Install and set up MongoDB as the database system for storing application data. Ensure that the appropriate version of MongoDB is installed and accessible from the back-end server.
  + General Requirements:
* Code Editor: Choose a code editor of your preference, such as Visual Studio Code, Atom, or Sublime Text. Install the editor and set up any necessary extensions or plugins for better development experience.
* Git: Install Git for version control and collaboration. This will allow you to track changes, manage branches, and collaborate with other developers using Git repositories.
* Command Line Interface (CLI): Have a command line interface or terminal installed on your development environment. This will allow you to execute commands and run scripts for various development tasks.
* Operating System: The application can be developed and deployed on various operating systems like Windows, macOS, or Linux. Ensure that your chosen operating system is compatible with the required technologies and tools.

[For all the versions used click this link.](#_Chapter_Two:_Implementation" \t "19)

# **Conclusion**

In conclusion, our internship experience working on the development of an ecommerce app has been a transformative journey both personally and professionally. Throughout this internship, we have faced various challenges and personal obstacles that tested our resilience and determination. However, we are proud to say that we persevered and pushed ourselves to deliver outstanding work, even in the face of adversity.

One of the most significant aspects of this internship has been the immense personal growth we have experienced. Despite the personal problems we encountered in our lives, we managed to maintain our focus and commitment to our work. This internship has taught us the importance of time management, prioritization, and discipline. We have learned how to effectively balance our personal lives with our professional responsibilities, demonstrating our ability to overcome obstacles and meet deadlines.

Moreover, this internship has provided us with valuable opportunities to enhance our skills and acquire new ones. From technical expertise in app development to collaboration and teamwork, we have gained a wealth of knowledge that will undoubtedly benefit our future careers.

As we conclude this internship report, we would like to express our gratitude to our mentors and the entire team for their guidance, support, and encouragement throughout this journey. We are grateful for the opportunity to have been part of such a dynamic and challenging project, which has truly shaped our professional growth.

# **Bibliography**

* Chat GPT - OpenAI: As an AI language model, Chat GPT has provided valuable information and guidance throughout the internship project.

<https://chat.openai.com/>

* Google: Google search engine has been utilized to gather information, research best practices, and find relevant resources related to ecommerce app development.

<https://www.google.com/>

* Wikipedia: Wikipedia has been consulted for general background information on ecommerce, app development methodologies, and related concepts.

[https://en.wikipedia.org/wiki/E-commerce#Defining\_e-commerce](https://en.wikipedia.org/wiki/E-commerce%23Defining_e-commerce)

* React Hot Toast: <https://react-hot-toast.com/>
* React Toastify: [https://www.npmjs.com/package/react-toastify](https://www.npmjs.com/package/react-toastify%20)
* Axios: [https://axios-http.com/fr/docs/intro](https://axios-http.com/fr/docs/intro%20)
* Ant Design: [https://ant.design/](https://ant.design/%20)
* Emotion React and Emotion Styled: <https://emotion.sh/docs/styled>

# **Abbreviations**

* Gantt Chart:
* WBS: Work Breakdown Structure. It is a hierarchical decomposition of the project deliverables and work required to accomplish them.
* SOW: Statement of Work. It is a document that defines the project's objectives, deliverables, and scope.
* PM: Project Manager. The person responsible for planning, executing, and managing the project.
* ETC: Estimated Time to Complete. It represents the remaining time required to complete a task or project.
* EAC: Estimate at Completion. It is the estimated total cost of completing a task or project.
* UML: Unified Modeling Language
* OOA: Object-Oriented Analysis. It is the process of analyzing a system to identify the objects and their interactions.
* OOD: Object-Oriented Design. It is the process of defining the structure and behavior of the objects identified during the analysis phase.
* CRC: Class Responsibility Collaboration. It is a technique used during object-oriented design to identify classes, their responsibilities, and collaborations.
* ERD: Entity-Relationship Diagram. It is a visual representation of the relationships between entities in a database.
* Scrum:
* PO: Product Owner. The person responsible for managing the product backlog and prioritizing the work.
* SM: Scrum Master. The person responsible for ensuring that the Scrum process is followed and removing any obstacles.
* Sprint: A time-boxed iteration in Scrum, usually lasting 1-4 weeks, where a set of product backlog items is completed.
* User Story: A short, simple, and understandable description of a feature or requirement from the perspective of the end-user.