A

Project Report

On

Online PC Mart

Project report submitted to Ganpat University in the partial fulfillment of the requirement for the award of the Degree of Master of Computer Applications.

**MCA Sem-3 [2024-2026]**

**24034211046**

**PATEL ARYA**

and

**24034211001**

**ACHARYA DHUN**

Under the Guidance of

**Ms. ARUNA GURJAR**

### Assistant Professor

****

Acharya Motibhai Patel Institute of Computer Studies, Ganpat University, Ganpat Vidyanagar – 384012.

**Nov 2025-26**



## PLAGIARISM DECLARATION FORM

This form must be completed, signed and attached to all assignments/ projects / dissertations. Please complete the information below (using BLOCK LETTERS):

Student’s Name: PATEL ARYA | Enrollment No. 24034211046

Name: ACHARYA DHUN | Enrollment No. 24034211001 Batch Code: **MCA 2024-2026**

Subject Name: **M.C.A. SEM – III P13A6SDP1 System Development Project - I**

The following definition of plagiarism is taken from the MLA Handbook for Writers of Research Papers, Theses and Dissertations (MLA: New York, 1977, 99 4-5) “Plagiarism may take the form of repeating another’s sentences as your own, adopting a particularly apt phrase as your own, paraphrasing someone else’s argument as your own, or even presenting someone else’s line of thinking in the development of a thesis as though it were your own. In short, to plagiarize is to give the impression that you have written or thought something that you have in fact borrowed from another. Although a writer may use other person’s words and thoughts they must be acknowledged as such (by the use of the appropriate reference, and by the insertion of quotation marks around any words directly quoted.”

**PLAGIARISM DECLARATION**

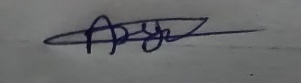
1. I acknowledge and understand that plagiarism is wrong and that it constitutes academic theft.
2. I understand that my written work must be accurately referenced. I have followed the rules and conventions concerning referencing laid out in the course outline for this course.
3. I have not allowed, nor will I in the future allow, anyone to copy my work with the intentions of passing it off as his or her work. I also accept that submitting identical work to someone else (a syndicate assignment) constitutes a form of plagiarism. I accept that the same principle applies to authorized group work.

**PLAGIARISM WARNING**

1. Any student found to have committed or aided and abetted the offence of plagiarism may be subjected to the following penalties depending on the severity of his involvement in the offence.
   1. The student shall receive no marks or a reduction of marks for the relevant academic assignment, project or dissertation;
   2. Subsequent offences will attract more severe penalties, including possible termination of studies.
2. Students should seek clarification from their respective.

**DECLARATION BY STUDENT** lecturers, tutors or supervisors if they are unsure whether they are plagiarizing the work of another person

I have read and understood the above definition of plagiarism. I am aware of and understand the Institute’s policy on plagiarism. I declare that all material in this assignment/project/dissertation is my own work and does not involve plagiarism.

Student’s Signature

Date: 25/11/2025



**25/ 11/ 2025**

**CERTIFICATE**

### TO WHOM SO EVER IT MAY CONCERN

This is to certify that the following students of M.C.A. Semester-III (P13A6SDP1 System Development Project - I) have completed their project work titled

Online PC Mart

satisfactorily fulfill the requirement of M.C.A Semester-III, Ganpat University, Ganpat Vidyanagar, in the Year 2025-26.

|  |  |  |
| --- | --- | --- |
| **Enrollment No** | **Name** | **Exam No** |
| **24034211046** | **PATEL ARYA** | 24034211046 |
| **24034211001** | **ACHARYA DHUN** | 24034211001 |

**External/Internal Guide Project Co-ordinator Principal**

(Prof. Ms. Aruna Gurjar) (Prof. C. D. Patel) (Dr. Nirbhay Chaubey)

## PREFACE

The completion of this project, *“Online* PC Mart*”*, marks an important milestone in my MCA program. This project has been undertaken as part of the academic curriculum with the objective of transforming theoretical knowledge into a practical, real-world application.

In today’s digital age, purchasing a computer system is often confusing for users due to compatibility issues, a lack of proper guidance, and the complexity of comparing prebuilt options. The Online PCMart , developed provides an online solution that addresses these challenges.

The platform allows users to build their own custom PC by selecting compatible components, or directly choose from a range of prebuilt systems. Additionally, the system provides **filtered prebuilt options** tailored for different user needs such as *Gaming PCs*, *Professional PCs*, or *Basic PCs*, with the ability to filter further by **budget range**, making the selection process simpler and more user-friendly.

Key features of the system include the ability to **save custom builds for future use** and **purchase PCs directly online**, whether they are user-built configurations or prebuilt systems. This ensures that users can either experiment and refine their builds at their convenience or proceed directly to buy the system that best suits their needs.

The application is developed using **React** for the front end, and **Spring Boot** for the back end, with **MySQL** as the database. This combination ensures a robust, scalable, and efficient environment for managing components, user preferences, saved builds, and orders.

The development of this project has been a valuable learning experience, enabling me to practically apply concepts of software engineering, system design, and full- stack development. It has helped me gain confidence in creating user-centric solutions that combine both functionality and usability.

We sincerely hope this project serves as a useful resource and contributes to simplifying the computer-buying experience for users while also being of academic value to future learners

## ACKNOWLEDGEMENT

We are sincerely grateful to Ganpat University and to my institute, A. M. Patel Institute of Computer Studies, for providing me with the opportunity to carry out my MCA project titled “Online PCMart”,

We extend my profound gratitude to my project guide, Ms. Aruna Gurjar, for her constant guidance, valuable suggestions, and encouragement throughout the course of this project. Her support and expertise have been pivotal in the successful completion of this work.

we are also thankful to Dr. Nirbhay Chobey, Head of the Department of Computer Applications, for granting me the facilities and resources required for the development of this project. I would also like to acknowledge all the faculty members of the department for their continuous support and motivation.

Finally, we wish to acknowledge the encouragement and cooperation received from my peers and colleagues during this project, which has been a great source of inspiration.

**CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** |  | **Name of Content** | **Page No.** |
| **1** |  | **INTRODUCTION** | 1 |
| 1.1 | Project Profile | 1 |
| 1.2 | About the Organization | 3 |
| **2** |  | **SYSTEM STUDY AND ANALYSIS** | 4 |
| 2.1 | Problem Statement | 4 |
| 2.2 | Existing System Drawbacks | 5 |
| 2.3 | Proposed System   * + 1. Advantages / Comparison Study     2. Modules (with short description) | 6  6  8 |
| **3** |  | **DEVELOPMENT ENVIRONMENT** | 9 |
| 3.1 | Hardware Requirement | 9 |
| 3.2 | Software Requirement | 10 |
| 3.3 | Programming Environment   * + 1. About Front-End     2. About Back-End | 11 |
| 11 |
| 12 |
| **4** |  | **SYSTEM DESIGN AND DEVELOPMENT** | 12 |
| 4.1 | Time line chart (Gantt chart) | 12 |
| 4.2 | Data Dictionary | 13 |
| 4.3 | E-R Diagram | 23 |
| 4.4 | User Modeling Language Diagrams   * + 1. Use Case Diagrams     2. Activity Diagrams     3. Sequence Diagrams     4. Collaboration Diagrams     5. Class Diagrams | 24 |
| 24 |
| 26 |
| 27 |
| 28 |
| 29 |
| 4.5 | Input Design | 30 |
| 4.6 | Output Design | 43 |
| **5** |  | **SYSTEM TESTING** | 44 |
| 5.1 | System Testing | 44 |
|  | * + 1. Output Testing | 45 |
|  | 5.1.2 Validation and Verification Testing | 46 |
| **6** |  | **CONCLUSION AND FUTURE ENHANCEMENT** | 51 |
| 6.1 | Conclusion | 51 |
| 6.2 | Future Enhancements | 52 |
| **7** |  | **BIBLIOGRAPHY & REFERENCES** | 53 |
| 7.1 | Books References | 53 |
| 7.2 | Web References | 54 |

# Chapter – 1 INTRODUCTION

* 1. **Project Profile**

### Project Title:

Online PCMart

### Objective:

The primary objective of this project is to design and develop a web-based application that allows users to build their own personal computers (PCs) by selecting compatible components, or directly choose from prebuilt systems based on their requirements. The application aims to simplify the PC buying process and provide a seamless online shopping experience.

Scope:

The system provides a platform where users can:

* Build a custom PC by selecting CPU, GPU, RAM, Storage, Cabinet, PSU, and other components.
* Filter and purchase prebuilt PCs tailored for different categories such as *Gaming*, *Professional*, or *Basic use*, with the ability to apply budget filters.
* Save their custom builds for later modifications or purchase.
* Buy PCs online through an integrated checkout process.

The application ensures that users face no compatibility issues while building their PCs, and it provides transparency in pricing and configurations. It also serves as a user- friendly and reliable solution for technology enthusiasts, gamers, and professionals seeking tailor-made computing solutions.

### Technologies Used:

* **Frontend:** React
* **Backend:** Spring Boot Framework
* **Database:** MySQL
* **Tools/Server:** Maven, Apache Tomcat (embedded), IntelliJ IDEA

### Expected Outcome:

The system will streamline the process of configuring, saving, and purchasing a PC online, offering both flexibility and convenience to users. It will also serve as a scalable foundation for future enhancements such as payment gateway integration, user reviews, and advanced compatibility checks.

* 1. **About the Organization**
* This project is developed under the guidance of faculty at AMPICS, a recognized

institution affiliated with Ganpat University, known for its commitment to academic

excellence and practical learning in the field of computer applications.

* The Department of Computer Applications provides a strong foundation in

software development, supported by experienced faculty, modern

infrastructure and a curriculum that emphasizes real-world application.

This environment has played a crucial role in the successful development of this

project.

# Chapter – 2 SYSTEM STUDY AND ANALYSIS

* 1. **Problem Statement**

Purchasing a personal computer is often a challenging task for many users due to the wide variety of components, models, and configurations available in the market. Customers frequently face the following difficulties:

* **Compatibility Issues:** When building a PC, users are often uncertain whether the selected components (CPU, motherboard, RAM, GPU, power supply, etc.) will work together properly.
* **Limited Customization in Existing Platforms:** E-commerce platforms such as Amazon or Flipkart allow purchasing of individual components or prebuilt systems, but they do not provide an interactive way to build and verify a complete system configuration.
* **Lack of Transparency:** Offline vendors may bundle incompatible or overpriced components, leaving customers with fewer choices and limited knowledge about the available options.
* **Time-Consuming Comparisons:** Manually comparing different prebuilt systems or attempting to create a custom PC can be overwhelming for non-technical users.
* **No Save or Reuse Option:** Customers who want to experiment with different builds often have to start over each time, as there is no provision to save and refine their builds for future consideration.

Due to these challenges, customers struggle to make well-informed decisions when purchasing or building a PC. There is a strong need for an **online platform** that allows users to:

* Easily configure a PC by selecting compatible components.
* Explore prebuilt systems based on usage (Gaming, Professional, Basic) and budget range.
* Save custom builds for future reference.
* Proceed with online purchase in a simplified and user-friendly manner.
  1. **Existing System Drawbacks**

Currently, customers have two primary options when purchasing a computer:

1. Offline Market / Local Computer Shops
   * Customers visit physical stores where vendors suggest components or prebuilt PCs.
   * The selection depends largely on the vendor’s recommendations and available stock.
   * There is limited scope for exploring alternative configurations or ensuring compatibility independently.
2. Online E-Commerce Platforms (Amazon, Flipkart, etc.)
   * These platforms allow customers to buy individual components or prebuilt PCs.
   * However, they do not provide an integrated PC-building feature where users can configure and verify complete systems.
   * Comparisons between prebuilt systems are limited and not tailored to specific user needs (e.g., gaming vs. professional use).
     1. **Drawbacks of the Existing System**
        + No Compatibility Check: Users must manually research whether selected components are compatible, which is confusing and time-consuming.
        + Lack of Customization: Prebuilt PCs from vendors or e-commerce sites provide limited options, restricting flexibility for customers.
        + Vendor Dependency: In offline shops, customers often depend on vendor suggestions, which may not always be transparent or unbiased.
        + No Save Feature: Users cannot save their selected builds for future modifications or purchases.
   1. **Proposed System**

The Online PCMart provides a comprehensive online solution that addresses the limitations of the existing systems. It enables users to build their own PCs by selecting compatible components or choose from a wide range of prebuilt systems categorized according to their needs. The system ensures transparency, ease of use, and reliability in the PC buying process.

* + 1. **Advantages / Comparison Study**

Compared to the existing systems, the proposed solution offers the following advantages:

* + - * **Custom PC Building:** Allows users to build a PC configuration step by step, tailored to their needs.
      * **Prebuilt Systems with Filters:** Provides categorized prebuilt PCs (*Gaming*,

*Professional*, *Basic*) with budget-based filtering for quick selection.

* + - * **Save Build Feature:** Users can save their custom build for later modifications or purchases.
      * **Simplified Purchase Process:** Users can directly buy a saved build or prebuilt system through an integrated checkout process.
      * **Transparency in Pricing:** Displays real-time pricing of individual components and complete builds.
      * **User-Friendly Interface:** Simplifies the otherwise complex process of configuring and buying a PC.

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature / System** | **Offline Shops** | **E-Commerce (Amazon/Flipkart)** | **Online PCMart** |
| Custom PC Building | Limited | Manual (No integration) | ⬛ Fully supported |
| Prebuilt PC Options | Limited | Available | ⬛ Categorized + Filterable |
| Save Build Feature | + No | + No | ⬛ Yes |
| Transparency in Pricing | + Often lacking | Partial | ⬛ Full |
| Purchase Convenience | Manual process | Available | ⬛ Streamlined Online |

* + 1. **Modules (with short description)**

The proposed system is divided into the following modules:

### User Module

* + Handles user registration, login, and profile management.
  + Stores saved builds and order history.

### Component Selection Module

* + Allows users to choose individual components step by step.
  + Provides compatibility checks between components.

### Prebuilt Systems Module

* + Offers ready-to-purchase PC builds categorized as *Gaming*, *Professional*, and *Basic*.
  + Includes budget filter for easier selection.

### Build Management Module

* + Enables users to save their custom builds for later.
  + Option to load and modify saved builds.

### Cart & Checkout Module

* + Handles selected builds or prebuilt systems added by the user.
  + Provides streamlined checkout and order confirmation.

### Admin Module

* + Allows admin to manage available components, update prices, add/edit prebuilt systems, and monitor orders.

# Chapter – 3 DEVELOPMENT ENVIRONMENTS

* 1. **Hardware Requirements**

To develop and run the Online PCMart efficiently, the following minimum hardware configuration is recommended:

* **Processor (CPU):** Intel i5 (10th Gen or above) / AMD Ryzen 5 or higher
* **RAM:** 8 GB (16 GB recommended for smoother development and testing)
* **Storage: 500 GB HDD (256 GB SSD or higher recommended)**
* **Graphics: Integrated graphics sufficient; dedicated GPU optional**
* **Monitor: Minimum resolution 1366 × 768 (1920 × 1080 recommended)**
* **Other Peripherals: Keyboard, Mouse, Internet Connectivity**
  1. **Software Requirements**

The software stack required for the implementation of the Online

PCMart system includes:

* **Operating System: Windows 10/11 (64-bit) / Linux (Ubuntu 20.04 or higher)**
* **Backend Framework:** Spring Boot (Java-based framework)
* **Frontend:** React
* **Database:** MySQL (v8.0 or above)
* **Development Tools:**
  + IntelliJ IDEA / Eclipse (for Java Spring Boot)
  + Visual Studio Code (for frontend development)
  + MySQL Workbench (for database management)
* **Build Tool:** Maven / Gradle
* **Server:** Apache Tomcat (embedded in Spring Boot)
* **Version Control:** Git / GitHub
  1. **Programming Environment**

The Online PCMart application is developed using a full-stack approach, combining frontend technologies for user interaction and backend technologies for data management and business logic.

* + 1. **About Front-End**

**The frontend of the application is developed using:**

* + - * **HTML5:** Provides the structure and semantic layout of web pages.
      * **CSS3**: Enhances the appearance of the web pages with styling, layouts, and responsiveness.
      * **JavaScript:** Adds interactivity, validation, and dynamic updates to the user interface.
      * **React:** for robustness design

**The frontend is designed to provide an intuitive and user-friendly interface where users can:**

* + - * Build custom PCs by selecting compatible components.
      * Browse and filter prebuilt systems (Gaming, Professional, Basic).
      * Save builds and proceed to purchase.

The design focuses on simplicity, accessibility, and responsiveness across devices.

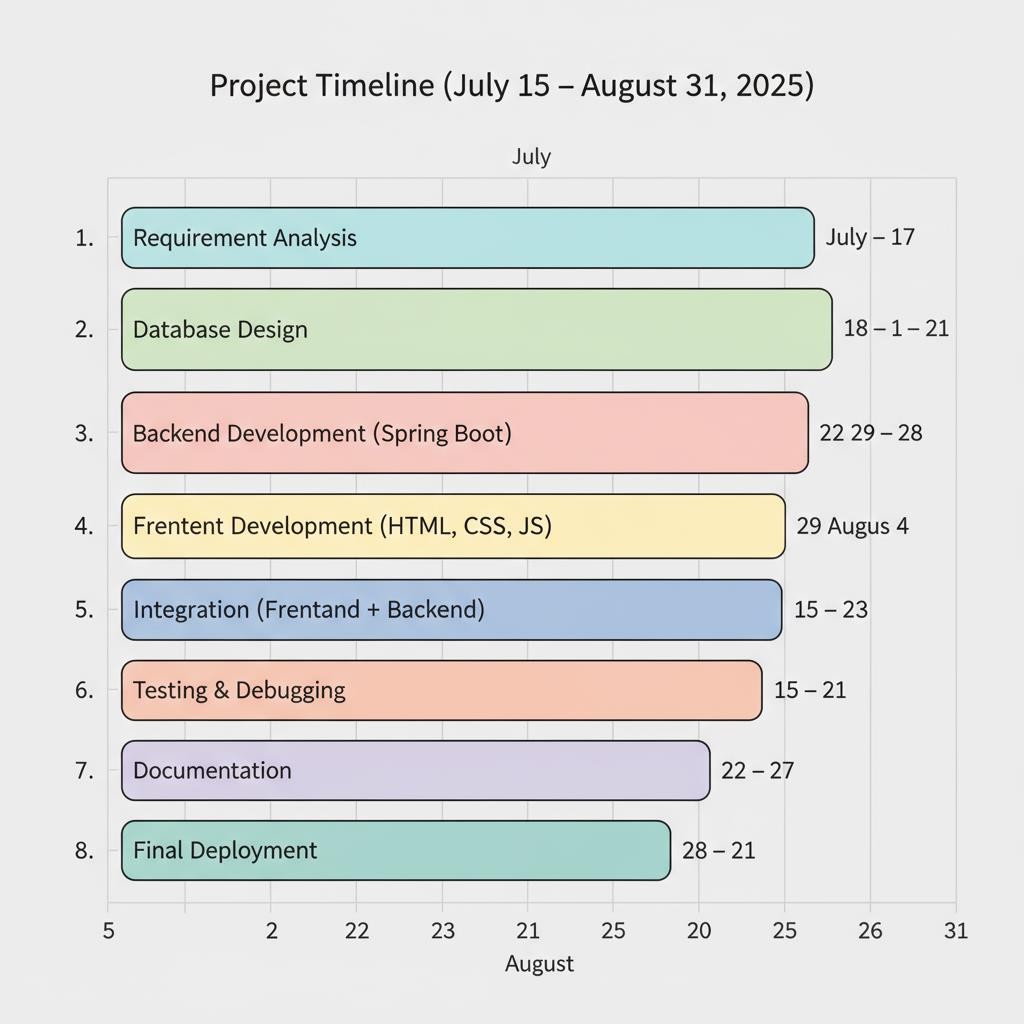
* + 1. **About Back-End**

The backend is developed using Spring Boot, a Java-based framework that simplifies enterprise-level application development. Key points include:

* + - * **RESTful APIs:** Handle communication between the frontend and backend.
      * **Business Logic:** Ensures compatibility checking, build saving, and purchase processing.
      * **Database Integration:** MySQL is used to store component details, prebuilt configurations, user accounts, saved builds, and order history.
      * **Security:** Authentication and authorization mechanisms to ensure safe user access.
      * **Scalability:** The modular design of Spring Boot allows easy future enhancements, such as advanced filters, recommendation engines, and payment gateway integration.

# Chapter – 4 SYSTEM DESIGN AND DEVELOPMENT

* 1. **Time line chart (Gantt chart)**



* 1. **Data Dictionary**

**Table USERS**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Default** | **Extra** | **Description** |
| user\_Id | Bigint | NO | PRI | NULL | auto\_increment | Unique identifier for each user (Primary  Key). |
| username | varchar(50) | NO | UNI | NULL |  | Unique username chosen by the user. |
| email | varchar(50) | NO | UNI | NULL |  | Unique email ID used  for registration and OTP verification. |
| password | varchar(100) | NO |  | NULL |  | Encrypted password  (hash + salt). |
| date\_ | datetime | YES |  | CURRENT\_TIMESTAMP | DEFAULT\_GENERATED | Date and time when the user registered. |
| otp | varchar(255) | YES |  | NULL |  | One-Time Password generated for email  verification or password reset. |
| otp\_expiry | bigint | YES |  | NULL |  | Expiry timestamp  (epoch format) for the OTP. |
| Token\_expiry | bigint | YES |  | NULL |  | Expiry timestamp (epoch format) for  JWT token/session. |
| role | varchar(50) | NO |  | USER |  | Defines user role (e.g., USER, ADMIN).  Default is USER. |
| profile\_picture | varchar(255) | YES |  | NULL |  | Stores the file path or URL of the profile  picture (optional). |

**Orders Table**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Defau lt** | **Extra** | **Description** |
| order\_id | bigint bigint  bigint int  datetime  varchar(50)  decimal(10,2) varchar(50) varchar(255) | NO NO  NO NO  NO  NO  NO NO YES | PRI FK  FK | NULL NULL  NULL  1  CURR ENT\_T IMEST AMP  PENDI NG  0  UNPAI D  NULL | auto\_increment  DEFAULT\_GENERATED | Unique identifier for each order.  Foreign key referencing **users(id)**, links order to a specific user.  Foreign key referencing product table (if applicable).  Number of units ordered.  Timestamp when the order was placed.  Order status (e.g., PENDING, CONFIRMED, SHIPPED, DELIVERED, CANCELLED).  Total price of the order.  Payment state (e.g., UNPAID, PAID, REFUNDED).  Address where the order is to be delivered. |
| user\_id |
| product\_id |
| quantity |
| order\_date |
| status |
| total\_amount |
| payment\_status |
| shipping\_address |

**CART TABLE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Null** | **Key** | **Default** | **Extra** | **Description** |
| cart\_id | bigint | NO | PRI | NULL | auto\_incre ment | Unique identifier for each cart entry. |
| user\_id | bigint | NO | FK | NULL |  | Foreign key referencing **users(id)** – identifies which user owns the cart. |
| product\_id | bigint | NO | FK | NULL |  | Foreign key referencing **products(product\_id)** – item added to the cart. |
| quantity | int | NO |  | 1 |  | Number of units of the product in the cart. |
| price\_per\_un it | decimal(10, 2) | NO |  | 0 |  | Price of one unit of the product (snapshot at the time of adding). |
| total\_price | decimal(10, 2) | NO |  | 0 |  | quantity × price\_per\_unit, stored for quick calculations. |
| date\_added | datetime | YES | CURRENT\_TIMES TAMP | | DEFAULT\_G ENERATED | Timestamp when the item was added to the cart. |
| status | varchar(50) | YES |  | ACTIVE |  | Status of cart item (e.g., ACTIVE, SAVED\_FOR\_LATER, REMOVED). |

**Power Unit supply**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for PSU |
| category | varchar(50) | Category (always PSU) |
| name | varchar(255) | Power Supply model name |
| brand | varchar(100) | Manufacturer brand |
| wattage\_w | int | Power output in watts |
| efficiency\_rating | varchar(50) | Efficiency certification (80+ Bronze/Gold/Platinum) |
| form\_factor | varchar(50) | Form factor (e.g., ATX, SFX) |
| modularity | varchar(50) | Cable modularity type (Non-modular/Semi/Full) |
| pcie\_connectors | varchar(100) | Available PCIe connectors |
| length\_mm | int | Physical length in mm |
| atx\_spec | varchar(50) | ATX specification version |
| pcie5\_12vhpwr | varchar(10) | Support for PCIe 5.0 12VHPWR connector |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| stock | int | Stock quantity |

**Motherboard**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for Motherboard |
| name | varchar(255) | Motherboard product name |
| brand | varchar(100) | Manufacturer brand |
| socket | varchar(50) | CPU socket type |
| chipset | varchar(50) | Chipset |
| form\_factor | varchar(50) | Form factor (ATX, mATX, ITX) |
| ram\_type | varchar(50) | Supported RAM type |
| ram\_slots | int | Number of RAM slots |
| max\_ram\_gb | int | Maximum RAM supported in GB |
| ram\_mhz\_max\_oc | int | Maximum RAM speed with overclock |
| pcie\_x16\_slots | int | Number of PCIe x16 slots |
| pcie\_gen\_primary | varchar(50) | Generation of primary PCIe slot |
| m2\_slots | int | Number of M.2 slots |
| m2\_pcie\_gen | varchar(100) | M.2 PCIe generation details |
| sata\_ports | int | Number of SATA ports |
| hdd\_support | varchar(255) | HDD support description |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (Motherboard) |
| stock | int | Stock quantit |

**Storage Unit**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for storage device |
| category | varchar(50) | Category (Storage) |
| kind | varchar(50) | Kind of storage (SSD/HDD/NVMe) |
| name | varchar(255) | Storage product name |
| brand | varchar(100) | Manufacturer brand |
| family | varchar(100) | Product family |
| model | varchar(100) | Model name |
| interface | varchar(100) | Interface type (SATA/NVMe Gen3/Gen4/Gen5) |
| form\_factor | varchar(50) | Form factor (2.5", M.2, 3.5") |
| capacity\_gb | int | Capacity in GB |
| read\_mb\_s | int | Sequential read speed (MB/s) |
| write\_mb\_s | int | Sequential write speed (MB/s) |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| stock | int | Stock quantity |

**Processor**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for CPU |
| name | varchar(255) | CPU product name |
| brand | varchar(100) | Manufacturer brand |
| series | varchar(100) | Series (Core, Ryzen, etc.) |
| generation | varchar(100) | CPU generation |
| model | varchar(100) | Model designation |
| cores | int | Number of cores |
| threads | int | Number of threads |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| generation\_label | varchar(255) | Generation label (formatted display) |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (CPU) |
| stock | int | Stock quantity |

**Ram**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for RAM |
| name | varchar(255) | RAM product name |
| brand | varchar(100) | Manufacturer brand |
| family | varchar(50) | Product family (RAM) |
| generation | varchar(50) | RAM generation (DDR3/DDR4/DDR5) |
| model | varchar(255) | RAM model designation |
| capacity\_gb | int | Capacity in GB |
| speed\_mhz | int | Speed in MHz |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (always RAM) |
| stock | int | Stock quantity |

**GPU**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for GPU |
| name | varchar(255) | GPU product name |
| brand | varchar(100) | Manufacturer brand |
| family | varchar(100) | GPU family (GeForce, Radeon) |
| generation | varchar(100) | GPU generation (RTX 30, RTX 40, etc.) |
| model | varchar(100) | Model designation |
| vram\_gb | int | VRAM size in GB |
| tdp\_w | int | Thermal Design Power in Watts |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (GPU) |
| stock | int | Stock quantity |

**Cooler**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for Cooler |
| name | varchar(255) | Cooler product name |
| brand | varchar(100) | Manufacturer brand |
| type | varchar(50) | Type (Air, AIO Liquid) |
| radiator\_size\_mm | int | Radiator size in mm (0 for air coolers) |
| height\_mm | int | Cooler height in mm |
| tdp\_w | int | Thermal Design Power supported in Watts |
| socket\_support | varchar(100) | Supported CPU sockets |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (Cooler) |
| stock | int | Stock quantity |

**Case**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| id | varchar(50) | Unique identifier for Case |
| name | varchar(255) | Case product name |
| brand | varchar(100) | Manufacturer brand |
| type | varchar(50) | Type (Mid Tower, Mini ITX, etc.) |
| form\_factor\_support | varchar(100) | Supported motherboard form factors |
| gpu\_length\_mm | int | Maximum GPU length supported (mm) |
| cpu\_cooler\_height\_mm | int | Maximum CPU cooler height supported (mm) |
| psu\_length\_mm | int | Maximum PSU length supported (mm) |
| drive\_bays\_3\_5 | int | Number of 3.5" drive bays |
| drive\_bays\_2\_5 | int | Number of 2.5" drive bays |
| radiator\_support | varchar(100) | Radiator support details |
| front\_io | varchar(255) | Front I/O ports |
| price\_inr | decimal(10,2) | Price in INR |
| usecase | varchar(255) | Intended use case |
| image | varchar(255) | Image URL |
| category | varchar(50) | Category (Case/Cabinet) |
| stock | int | Stock quantity |

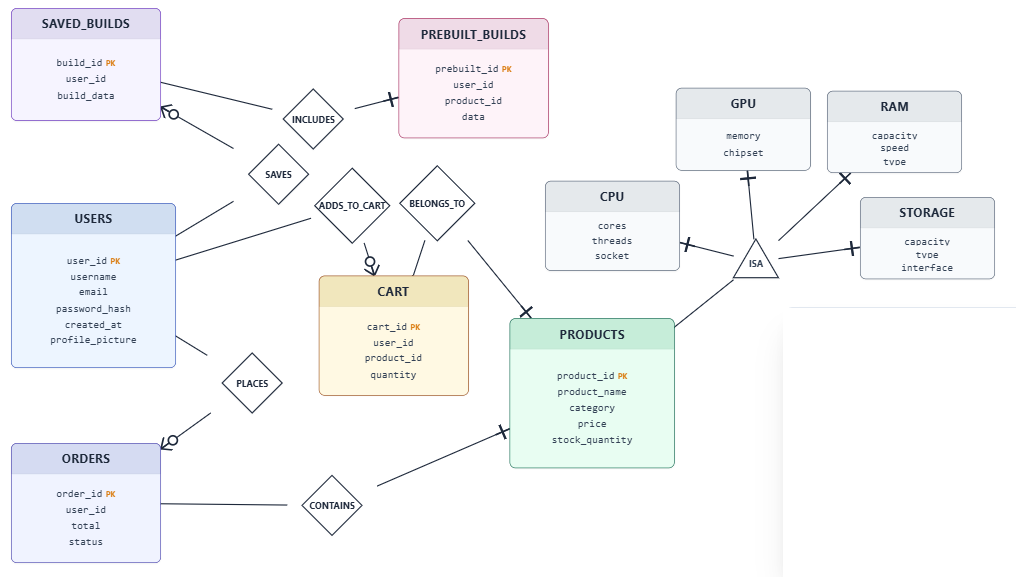
**Saved Builds**

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| build\_id | bigint (PK) | Unique identifier for the saved build. |
| user\_id | bigint (FK) | References **users(id)** – identifies who saved the build. |
| cpu\_id | varchar(50) | References **cpu(id)**. |
| gpu\_id | varchar(50) | References **gpu(id)**. |
| mobo\_id | varchar(50) | References **mother\_board(id)**. |
| ram\_id | varchar(50) | References **ram(id)**. |
| storage\_id | varchar(50) | References **storage(id)**. |
| psu\_id | varchar(50) | References **psu(id)**. |
| cooler\_id | varchar(50) | References **cooler(id)**. |
| case\_id | varchar(50) | References **case(id)**. |
| date\_saved | datetime | Timestamp when the build was saved. |
| total\_price | decimal(10,2) | Sum of component prices at the time of saving (price snapshot). |
| notes | varchar(255) | Optional notes added by user (e.g., “For gaming”, “Budget build”). |

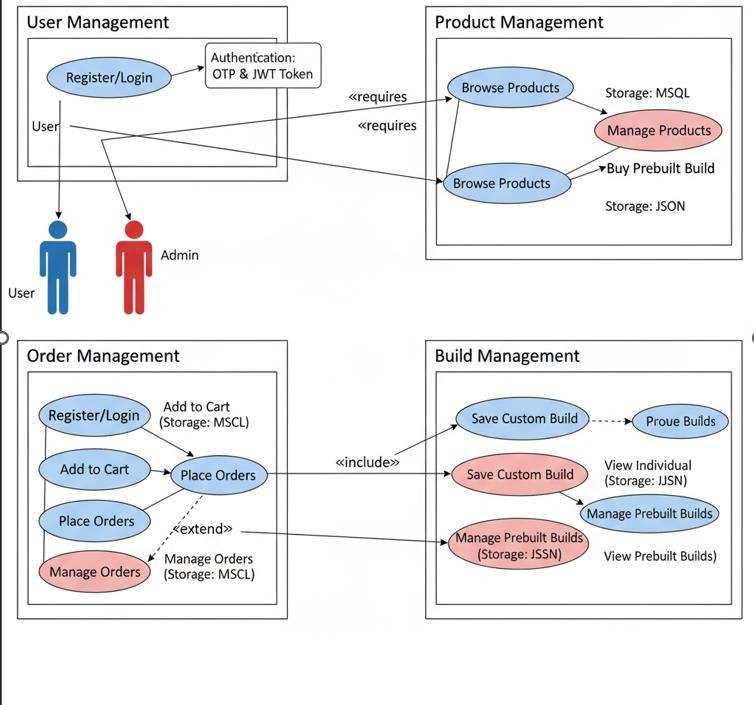
**Prebuilds**

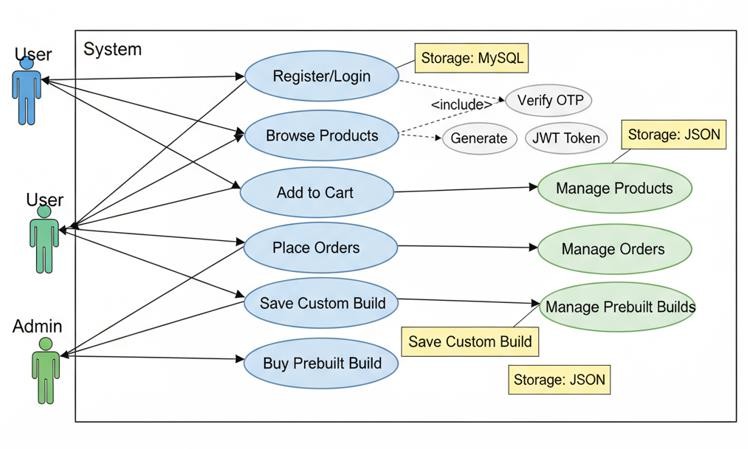
|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Description** |
| build\_id | bigint (PK) | Unique identifier for the saved build. |
| user\_id | bigint (FK) | References **users(id)** – identifies who saved the build. |
| cpu\_id | varchar(50) | References **cpu(id)**. |
| gpu\_id | varchar(50) | References **gpu(id)**. |
| mobo\_id | varchar(50) | References **mother\_board(id)**. |
| ram\_id | varchar(50) | References **ram(id)**. |
| storage\_id | varchar(50) | References **storage(id)**. |
| psu\_id | varchar(50) | References **psu(id)**. |
| cooler\_id | varchar(50) | References **cooler(id)**. |
| case\_id | varchar(50) | References **case(id)**. |
| date\_saved | datetime | Timestamp when the build was saved. |
| total\_price | decimal(10,2) | Sum of component prices at the time of saving (price snapshot). |
| notes | varchar(255) | Optional notes added by user (e.g., “For gaming”, “Budget build”). |

* 1. **E-R Diagram**

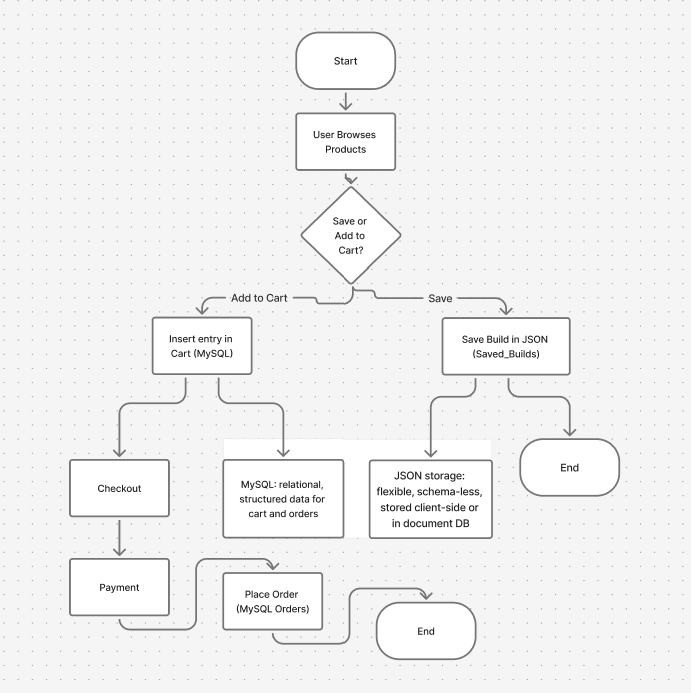
****

* 1. **User Modeling Language Diagrams**
     1. **Use Case Diagrams**

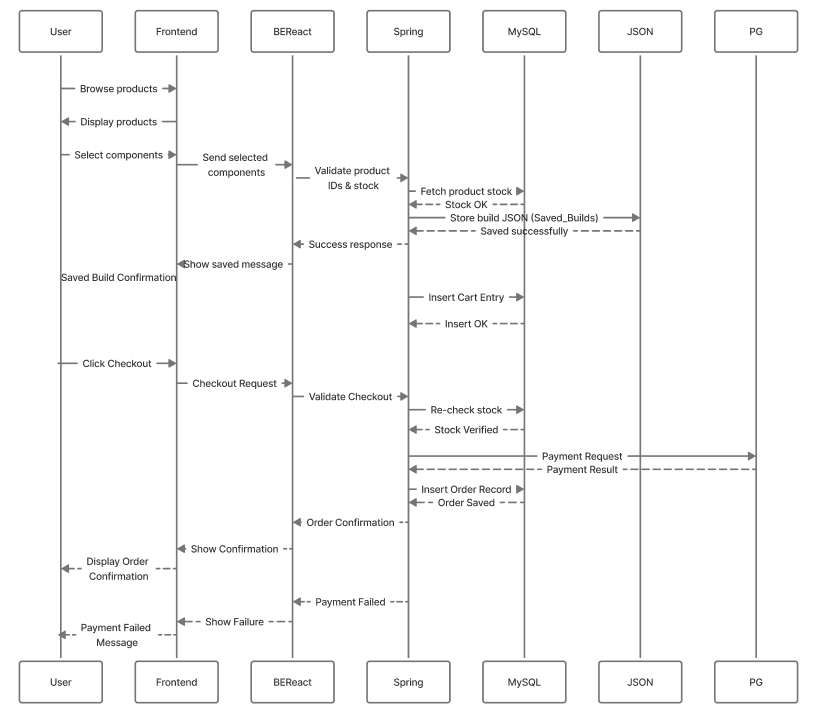
****



* + 1. **Activity Diagrams**

****

* + 1. **Sequence Diagrams**

****

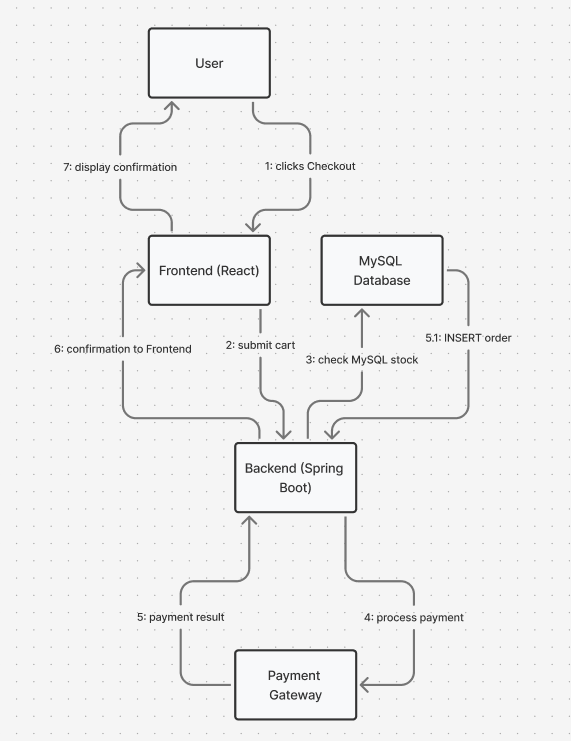
### Class Information

**User**: id, username, email, password, name, category, role, profile\_picture, expiry

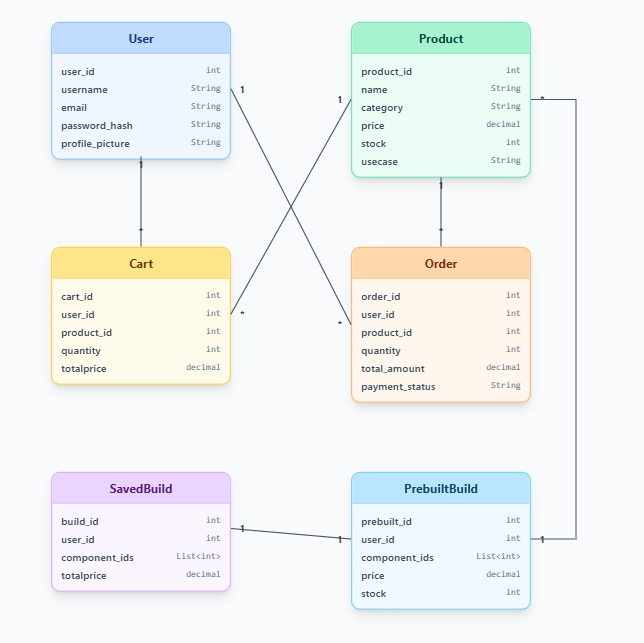
**Cart**: cart\_id, user\_id, product\_id, stock, status, total\_price

**Order**: order\_id, user\_id, product\_id, quantity, total\_amount, payment\_status **SavedBuild**: build\_id, user\_id, component\_ids[], total\_cost, stock **PrebuiltBuild**: prebuilt\_id, name, component\_ids[], price, stock

* + 1. **Collaboration Diagrams**

****

* + 1. **Class Diagrams**

****

* 1. **Input Design**

**Frontend Inputs**

### Login Page

* + Username or Email (string input).
  + Password (masked input).

### Registration Page

* + Username (validated: must start with lowercase letter, alphanumeric only).
  + Email (validated via regex).
  + Password (minimum 8 characters, must include uppercase and special character).
  + Confirm Password (must match password).
  + OTP (One-Time Password sent via email).
  + Profile Picture Upload (optional; accepts image file: jpg/png/gif).

### Forgot Password Page

* + Registered Email.
  + OTP (received in email).
  + New Password.
  + Confirm Password.

### Account Management Page

* + Change Username (validated with regex).
  + Upload Profile Picture.
  + Delete Profile Picture.

### PC Builder Page

* + Dropdown Selections for:
    - CPU
    - GPU
    - RAM
    - Storage
    - Cabinet
    - Operating System
  + Add to Cart button.
  + Save build button.
  + Buy button

### Prebuilt PC Page

* Filters:
  1. Usage Category (e.g., Gaming, Office, Development).
  2. Price Range (Budget, Mid, High).
* Add Prebuilt to Cart.
* Buy
* Customize

### Contact Us Page

* Full Name.
* Email Address.
* Message Text Area.

### Other UI Inputs

* Dark/Light Theme Toggle (switch).
* Navigation Menu Selection (Home, Build PC, Prebuilt, About Us, Account).

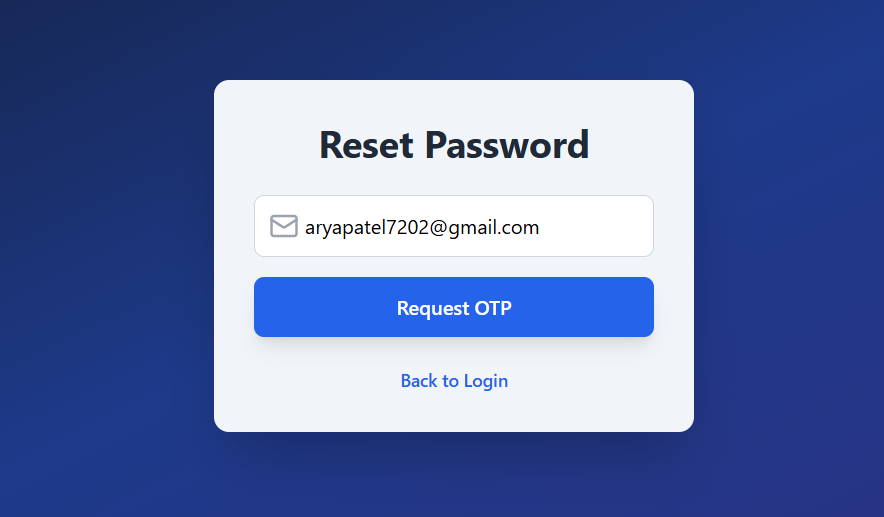
# Designs

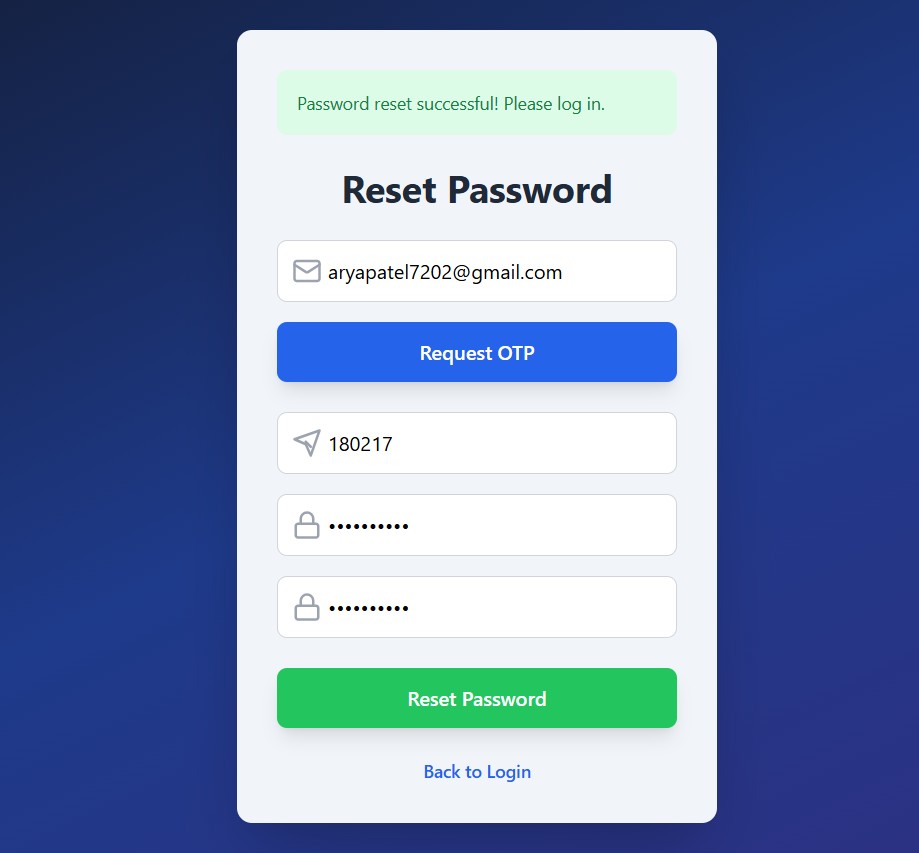
##### Login and Registration Page

##### 

##### 

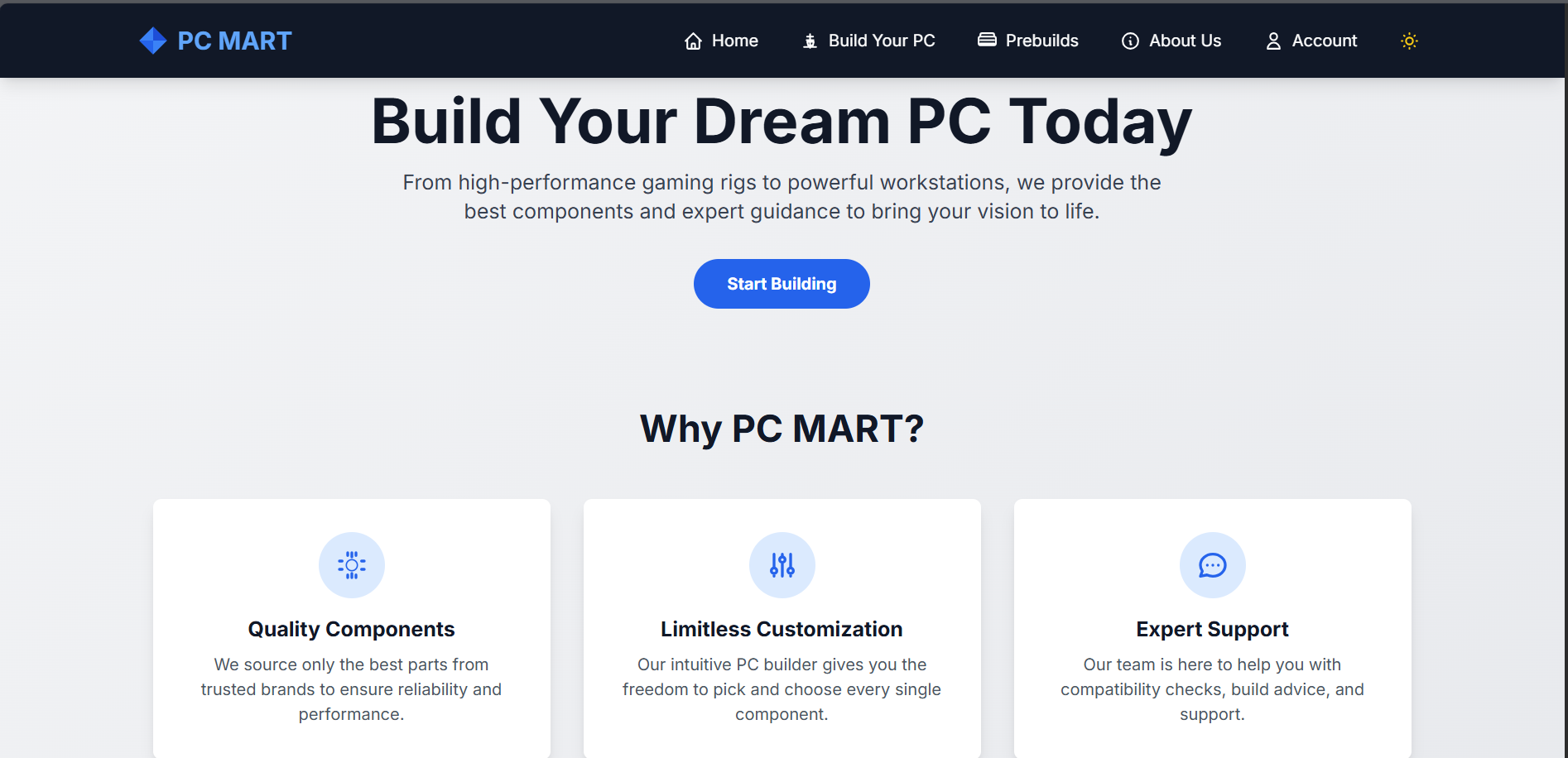
##### Forget Password:



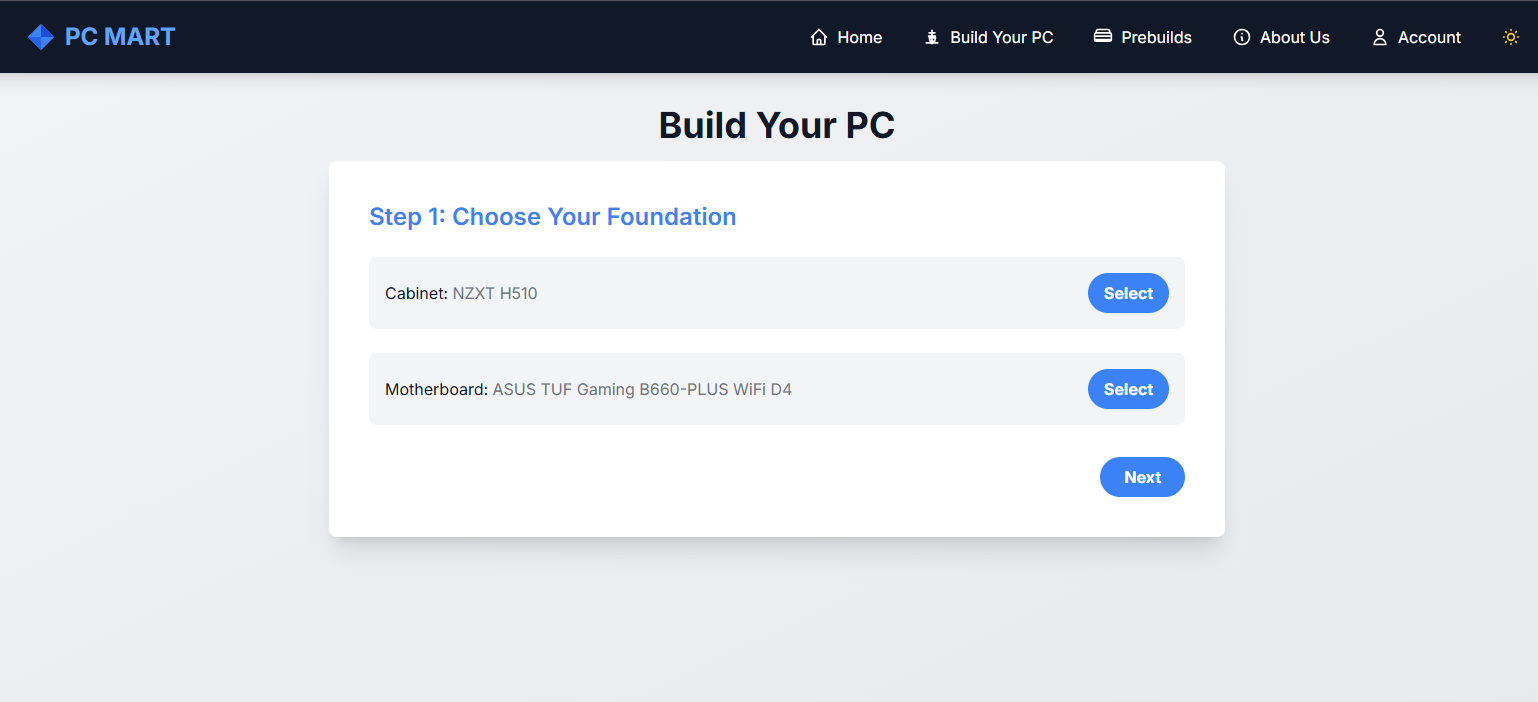


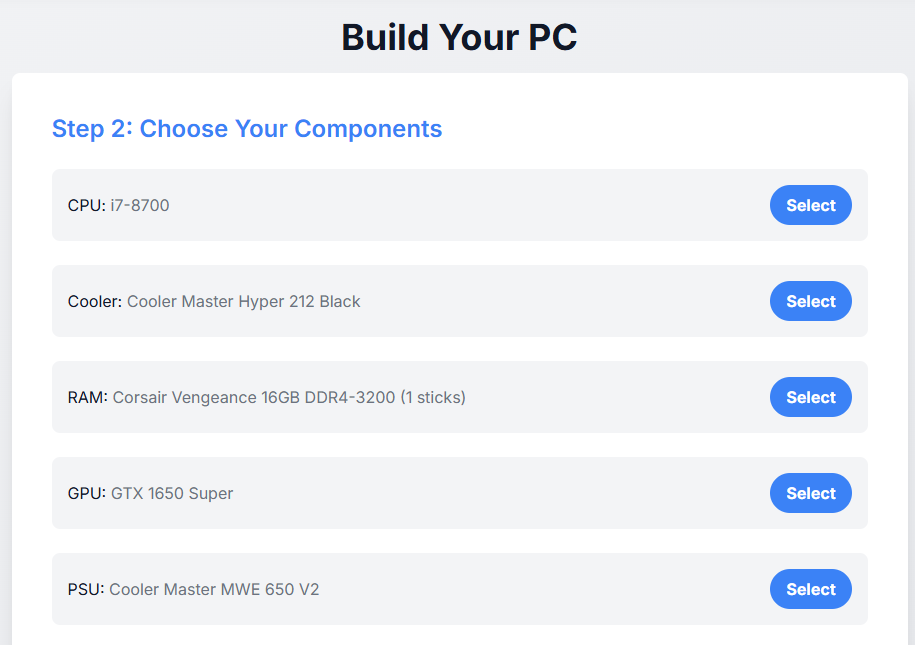
Home Page:

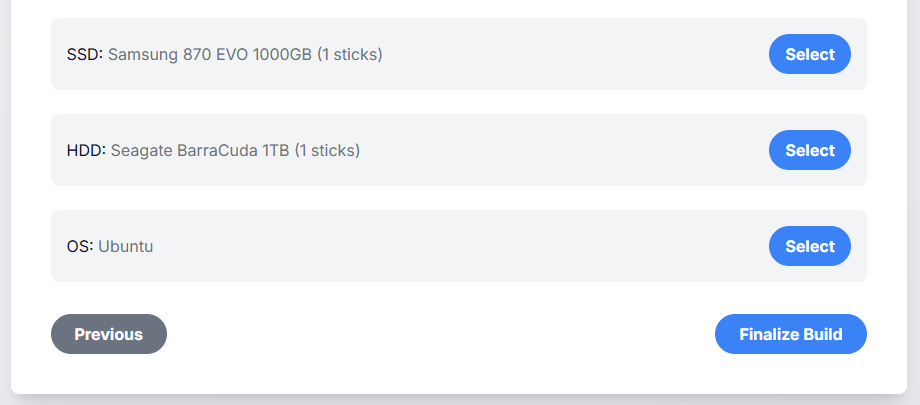
##### 

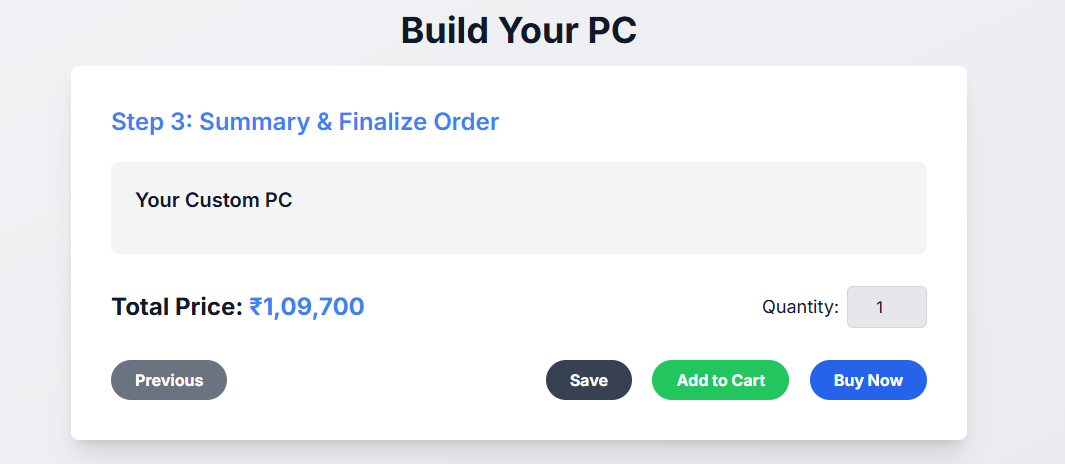


##### Build Section:

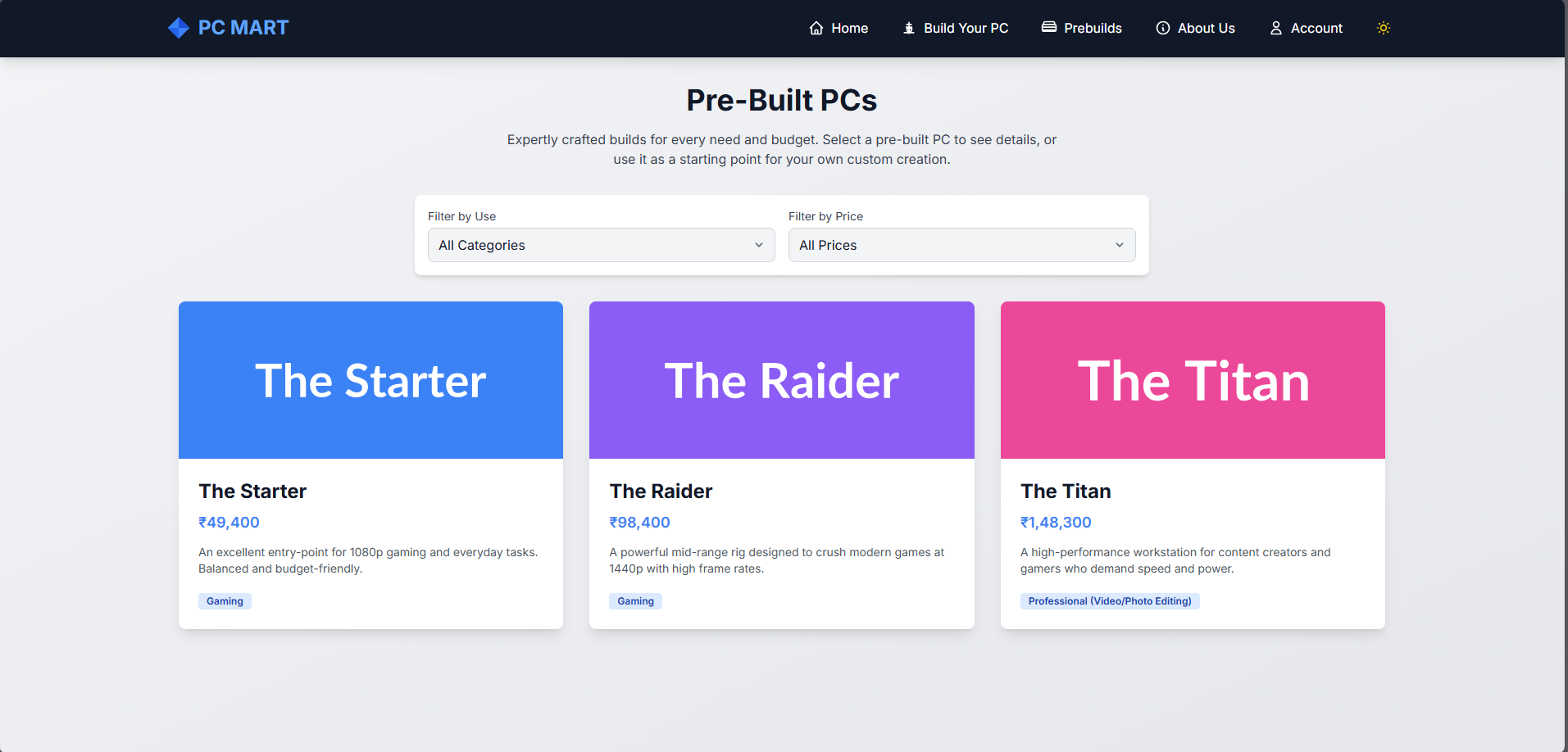


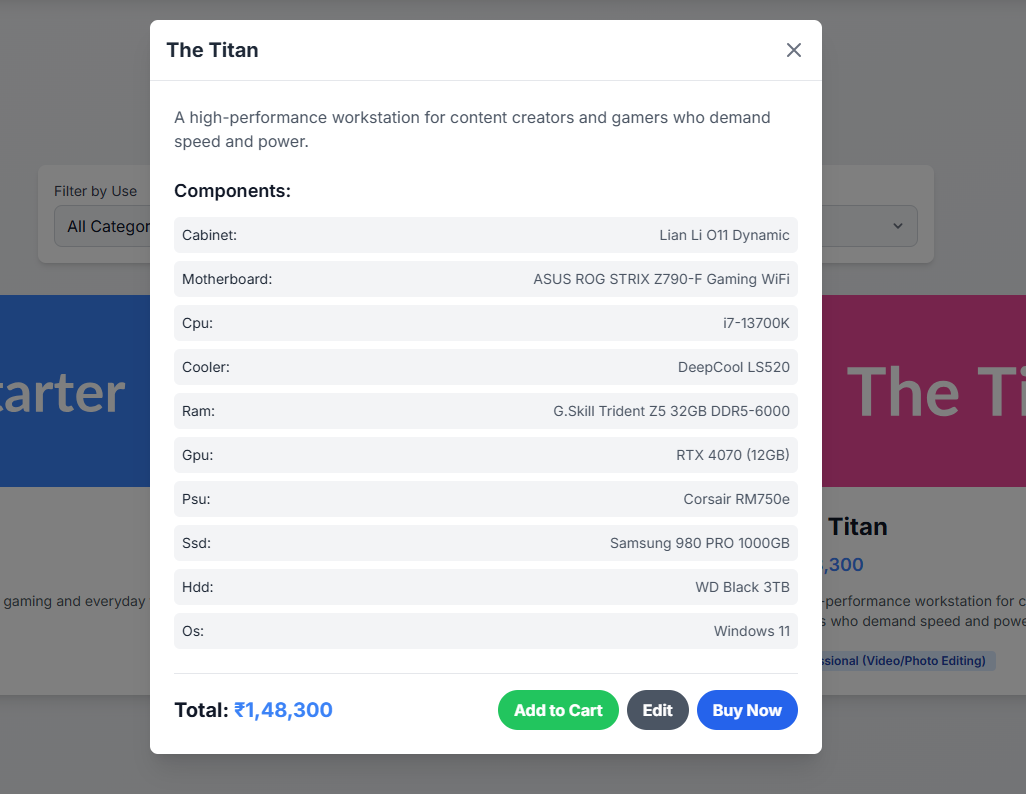






**PreBuild:**

****

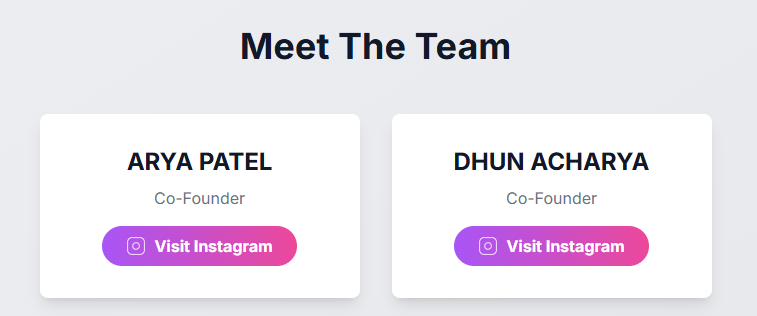
****

##### Accounts Section:

##### 

##### 

##### About us:



### Backend Inputs (Spring Boot APIs)

The backend accepts input via **REST API calls**:

### Authentication APIs

* /login → { username/email, password }
* /logout → clears user session.

### Registration APIs

* /request-otp-register → { username, email }
* /verify-otp-register → { email, otp, password, profile\_picture }

### Password Reset APIs

* /request-otp → { email }
* /verify-otp → { email, otp }
* /reset-password → { email, password }

### Profile Picture APIs

* POST /{userId}/profile-picture → multipart file upload.
* DELETE /{userId}/profile-picture → delete request.

### Username Update API

* PATCH /{userId}/username → { newUsername }

**Database Inputs (users table)**

* **id →** Auto-generated unique identifier (Primary Key, auto-increment).
* **username →** Unique string chosen by the user at registration; validated with regex.
* **email →** Unique string representing user’s email; validated format.
* **password →** Encrypted string (BCrypt hashing) stored securely in DB.
* **date\_ →** Datetime field storing account creation timestamp (auto-generated by DB).
* **otp →** Numeric/string OTP generated for verification (registration/reset password).
* **otp\_expiry →** Timestamp in milliseconds defining validity period of OTP (5 minutes).
* **expiry** → Timestamp in milliseconds defining login session expiry (10 days

default).

* **role →** String defining user role (default: USER, can be extended to ADMIN).
* **profile\_picture →** File path string for storing uploaded profile picture.
  1. **Output Design**

### Frontend Outputs (User Interface)

* **Success Message** → Confirms successful actions (e.g., “Login successful”, “Registration completed”).
* **Error Message** → Displays failure reasons (e.g., “Invalid password”, “OTP expired”).
* **Profile Picture Preview** → Displays uploaded/cropped profile image in account settings.
* **PC Build Summary** → Shows list of selected components with total calculated price.
* **Prebuilt PC Results** → Displays filtered PC builds (by usage or price).
* **Account Dashboard** → Shows saved builds, cart items, and past orders.
* **Contact Form Status** → Success or error message after submitting queries.

### Backend Outputs (API Responses)

* **Login Response** → { token, expiry, username, userId, prof\_pict } returned after successful login.
* **OTP Request Response** → "OTP sent to registered email" message.
* **OTP Verification Response** → "OTP verified, you can reset password" confirmation.
* **Registration Response** → "User registered successfully" message.
* **Password Reset Response** → "Password reset successful" confirmation.
* **Profile Picture Response** → { message, profile\_picture\_url } after upload/delete.
* **Username Update Response** → { message, username, token, expiry } after successful change.
* **Error Responses** → Return HTTP status codes with error messages (e.g., 400 Bad Request, 401

Unauthorized).

### Database Outputs (users table updates)

* **id** → Generated automatically when a new user is registered.
* **username** → Stored unique value; updated if user changes username.
* **email** → Stored unique value; retrieved during login/OTP requests.
* **password** → Encrypted string saved at registration or reset password.
* **date\_** → Auto-generated timestamp recorded at user creation.
* **otp** → Stores OTP temporarily; cleared after successful verification.
* **otp\_expiry** → Stores OTP expiry timestamp; validated during verification.
* **expiry** → Updated on login to extend session validity (default 10 days).
* **role** → Defines access rights; default is USER.
* **profile\_picture** → Stores path of uploaded profile image; updated on change or set to null if

deleted.

# Chapter – 5 SYSTEM TESTING

* 1. **System Testing**

System testing ensures that the application works as intended when all modules (frontend, backend, and database) are integrated.

* + - **Login Module Testing** → Verified login with correct and incorrect credentials; checked JWT

token generation and expiry handling.

* + - **Registration Module Testing** → Tested OTP generation, OTP expiry (5 minutes), duplicate

email/username rejection, and successful registration.

* + - **Password Reset Testing** → Verified OTP validation, password strength enforcement, and

successful reset.

* + - **Profile Management Testing** → Uploaded, changed, and deleted profile pictures; updated

username with regex validation.

* + - **PC Builder Testing** → Verified that component selection updates build summary and calculates

total price correctly.

* + - **Prebuilt PC Testing** → Tested filters (usage, price) to ensure correct PCs are displayed.
    - **Contact Form Testing** → Verified that form inputs submit successfully and show success/error

messages.

* + - **Theme & Navigation Testing** → Checked dark/light theme toggle and smooth navigation

across pages.

* + 1. **Output Testing**

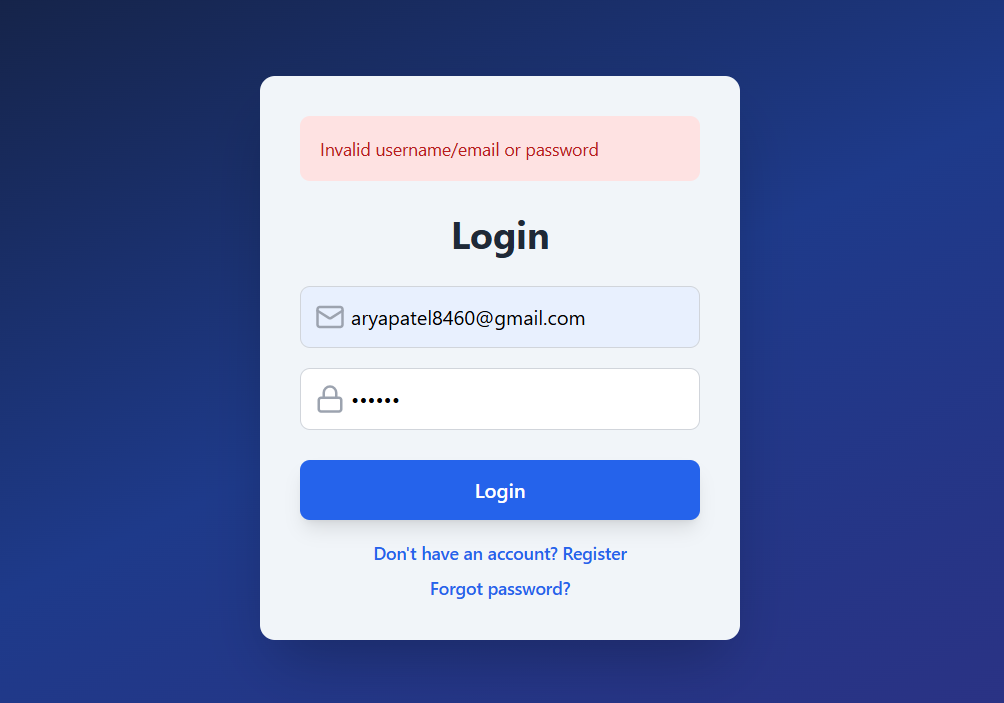
Output testing ensures that expected results are displayed for given inputs.

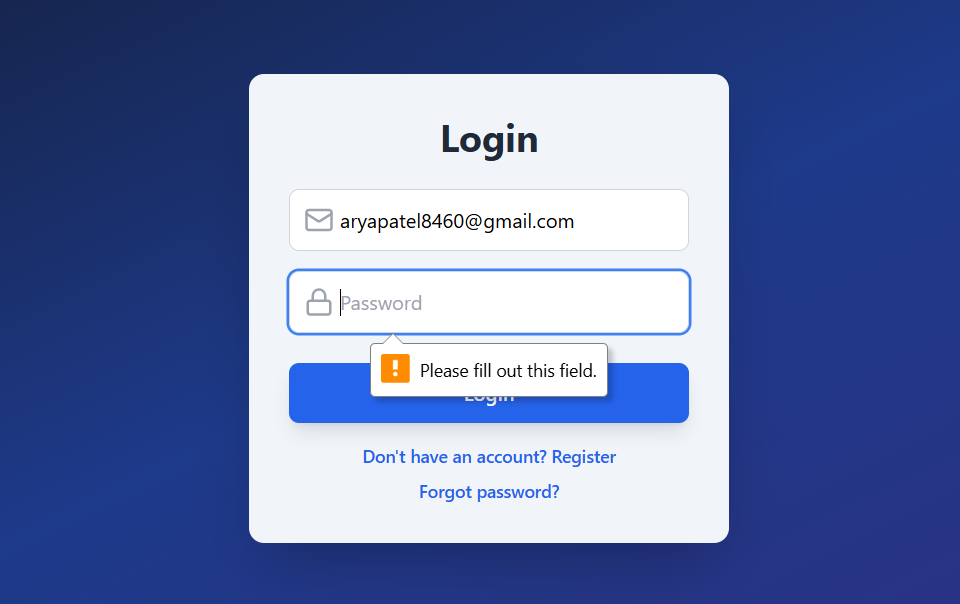
* + - * **Login Success** → Displays dashboard and returns { token, expiry, username, userId }.
      * **Login Failure** → Shows “Invalid username/email or password” error message.
      * **OTP Sent** → “OTP sent to registered email” message displayed.
      * **OTP Verified** → Confirms: “OTP verified, you can reset password now.”
      * **OTP Expired** → Shows “OTP expired” error.
      * **Registration Success** → Displays “User registered successfully.”
      * **Duplicate Email/Username** → Shows “Email/Username already registered” error.
      * **Password Reset Success** → Displays “Password reset successful.”
      * **Profile Picture Upload Success** → Returns file path and displays preview.
      * **Profile Picture Deletion** → Returns success message and removes preview.
      * **Change Username Success** → Returns new username and updated token.
      * **PC Build Summary** → Displays selected components with correct total price.
      * **Prebuilt Filter Output** → Displays correct builds based on filters (gaming, office,

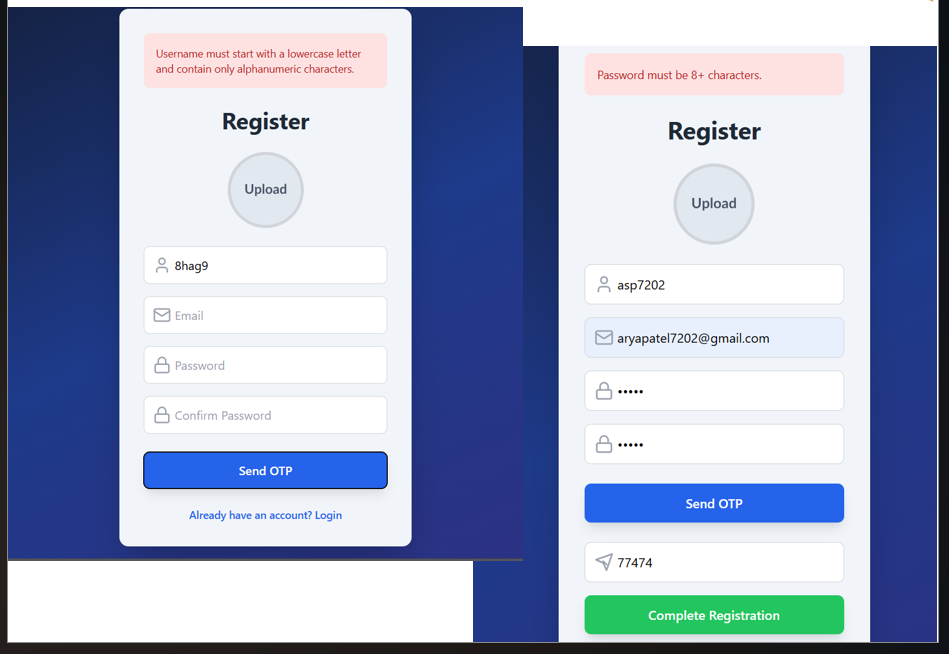
etc.).

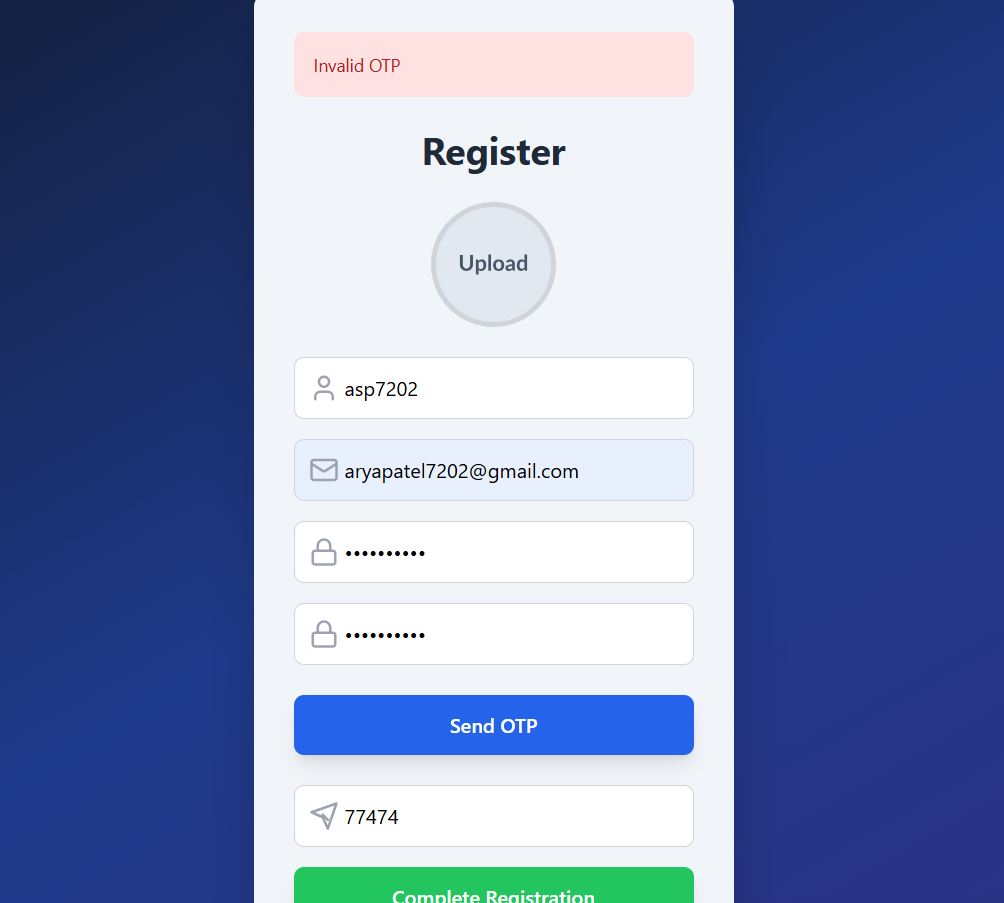
* + 1. **Validation and Verification Testing**

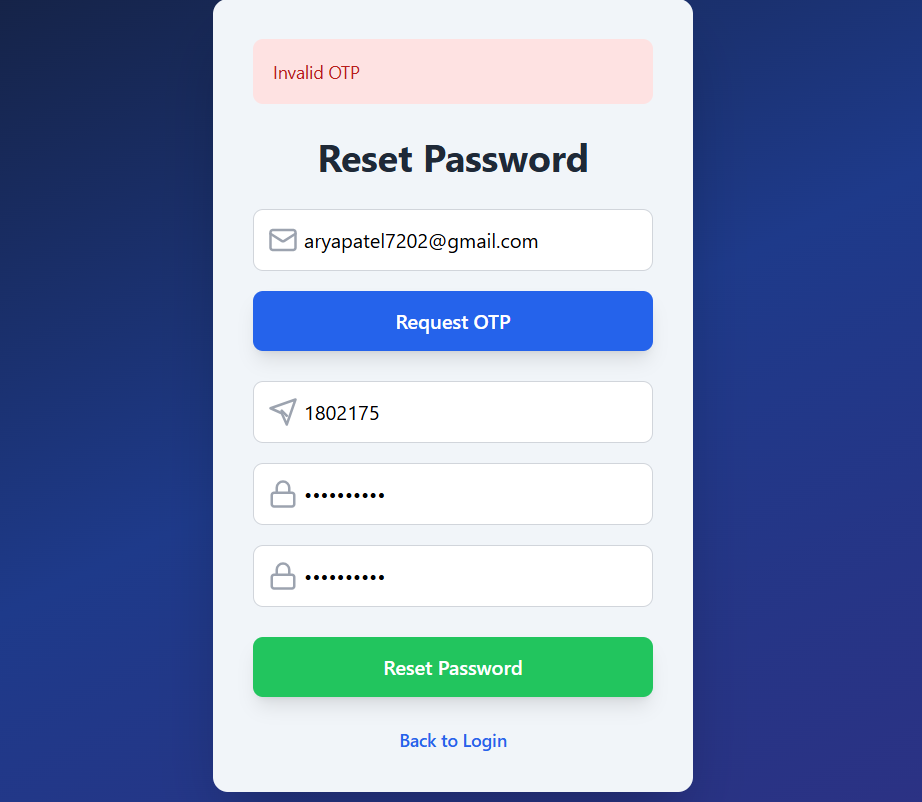
### Validation Testing

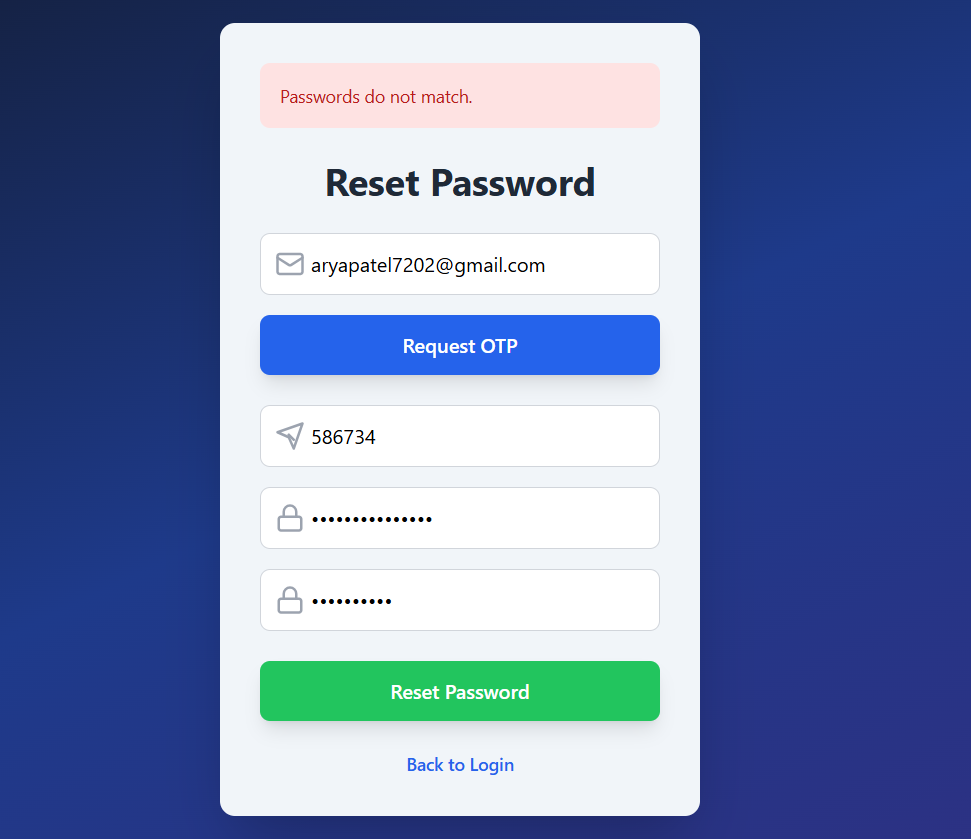
****

****

****

****

****

****

* + - * Ensured only unique usernames and emails are allowed.
      * Verified OTP prevents fake or duplicate accounts.
      * Confirmed password rules (length, strength) are enforced.
      * Validated that users cannot bypass login without JWT.
      * Checked PC Builder always calculates correct total price.

### Verification Testing

* + - * Used **Postman** to test all REST APIs (/login, /register, /reset-password, /profile- picture).
      * Verified JWT expiry handling (10 days session, auto-refresh logic).
      * Checked DB records after each operation (registration, reset, update).
      * Verified image upload/delete flow updates database correctly.
      * Ensured all error messages match invalid inputs (bad email, wrong OTP, expired session).

# Chapter – 6 CONCLUSION AND FUTURE ENHANCEMENT

* 1. **Conclusion**
     + **Secure Authentication Implemented →** The system successfully provides login, registration, password reset using OTP verification and JWT-based session management.
     + **Data Integrity Maintained →** User credentials are encrypted (BCrypt), ensuring data security.
     + **User Account Management →** Features like profile picture upload/delete and username update enhance personalization.
     + **PC Builder Functionality →** Users can customize PCs by selecting CPU, GPU, RAM, Storage, Cabinet, and OS, with real-time total price calculation.
     + **Prebuilt PC Selection →** Filter system allows quick access to recommended builds based on usage and budget.
     + **User-Friendly Frontend →** Developed with React + Tailwind CSS for responsive design, with dark/light theme support.
     + **Database Integration →** MySQL stores user details with secure schema design (unique username/email, encrypted password, OTP expiry, etc.).
     + **Overall Result →** The project integrates frontend, backend, and database layers to deliver a full-stack PC Builder platform that is secure, scalable, and user-friendly.
  2. **Future Enhancements**

### AI Recommendations

* + - Implement AI/ML models to suggest **best PC builds** based on **user preferences and budget**.

### Live Compatibility Check

* + - Automatically validate selected components (e.g., CPU + Motherboard compatibility).

### Mobile Application

* + - Develop a **mobile version using React Native** to allow users to build PCs on the go.

### Enhanced Security

* + - Add **Two-Factor Authentication (2FA)** using SMS/Email OTP or Google Authenticator.
    - Implement **refresh tokens** for extended login sessions.

# Chapter – 7 BIBILIOGRAPHY & REFERENCES

* 1. **Books References**
     + **Java: The Complete Reference** – Herbert Schildt

*(For Java fundamentals and backend logic building with Spring Boot)*

* + - **Spring in Action** – Craig Walls

*(For in-depth understanding of Spring Boot framework, dependency injection, and security concepts)*

* + - **JavaScript: The Definitive Guide** – David Flanagan

*(For frontend logic using JavaScript in React)*

* + - **React Up & Running** – Stoyan Stefanov

*(For ReactJS fundamentals, component design, and state management)*

* + - **Database System Concepts** – Abraham Silberschatz, Henry F. Korth, S. Sudarshan

*(For relational database design, schema creation, and queries)*

* + - **Learning Web Design** – Jennifer Niederst Robbins

*(For HTML, CSS, and responsive UI design concepts)*

* 1. **Web References**
     + **Spring Boot Official Documentation →** <https://spring.io/projects/spring-boot> (For backend API development and integration)
     + **Spring Data JPA Documentation** → <https://spring.io/projects/spring-data-jpa> (For ORM mapping and database operations)
     + **JWT (JSON Web Token) Documentation →** https://jwt.io (For authentication and secure session management)
     + **React Official Documentation →** https://react.dev (For frontend development with React components)
     + **Tailwind CSS Documentation →** [https://tailwindcss.com](https://tailwindcss.com/) (For frontend styling and responsive UI)
     + **MDN Web Docs →** https://developer.mozilla.org (For JavaScript, HTML, and CSS references)
     + **MongoDB Documentation →** <https://www.mongodb.com/docs/> (For NoSQL database concepts and queries)
     + **Cropper.js Library →** https://cropperjs.com (For profile picture cropping and upload feature)
     + **W3Schools →** https://[www.w3schools.com](http://www.w3schools.com/)

(For quick references on HTML, CSS, and JavaScript)