CHAPTER THREE: RESEARCH METHODOLOGY

## 3.0 Introduction

3.1 Research Design (Optional)

Given the exploratory nature of the study, a descriptive research design was adopted. This approach allows for the systematic examination and analysis of e-commerce adoption patterns, user behaviors, and associated challenges within the specific context of Banda, Kampala.

3.2 Population and Sample Selection (Optional)

To ensure the representativeness of the findings, a stratified random sampling technique was employed to select participants from the population of Banda residents engaged in e-commerce activities. The sample size was determined based on the desired level of precision and confidence interval, ensuring adequate representation across demographic variables.

3.2.1 Sampling Strategy: The population was stratified based on demographic characteristics such as age, gender, and occupation, and participants were randomly selected from each stratum.

3.2.2 Sample Size Determination: The sample size was calculated using appropriate statistical formulas, taking into account the population size and expected variability.

3.3 Research Instrument Design and Testing (Optional)

The research instruments, including surveys, interviews, and observation protocols, were meticulously designed and validated to ensure their reliability and validity.

3.3.1 Reliability Testing: Pilot testing and internal consistency analysis were conducted to assess the reliability of the research instruments.

3.3.2 Validity Testing: Content validity, criterion-related validity, and construct validity were evaluated to ensure the validity of the research instruments.

3.4 Data Collection and Analysis Methods

A variety of data collection methods were employed to capture diverse perspectives on e-commerce adoption and usage in Banda, Kampala.

3.4.1 Interview Method: Semi-structured interviews were conducted with key stakeholders, including e-commerce platform users, business owners, and industry experts, to gain insights into their

# CHAPTER THREE: RESEARCH METHODOLOGY

## 3.0 Introduction:

This chapter provides a detailed overview of the research methodology that will be used in this study. It includes a description of the research design, data collection methods, sampling design, data analysis methods, and measures taken to ensure validity and reliability of the results. The chapter concludes with a summary of the implications of the research methodology for the study.

## 3.1 Data Collection and analysis Methods

A mixed-methods research design will be used in this study, which involved the collection of both qualitative and quantitative data. The data collection methods will be used as follows:

### 3.1.1 Interview Method

Interviews serve as a fundamental data collection method wherein direct interaction occurs between the researcher and the participant(s). Through structured questioning, researchers gather firsthand information, insights, and perspectives on specific research objectives or topics. This method allows for an in-depth exploration and clarification of ideas, fostering a comprehensive understanding of the subject matter.

### 3.1.2 Observation

Observation involves systematically watching and recording behavior, interactions, and events in real-time within a specific setting or context. Researchers employ observational methods to gain insights into natural behaviors, social dynamics, and situational factors that may not be captured through other data collection techniques.

### 3.1.3 Group discussion

### //////drop the explanations here

## 3.2 System study and analysis methods

### 3.2.1 System Study methods

The researcher will examine the approaches that are currently in use to identify their advantages and disadvantages. The data was collected using a range of devices, such as interview guides and observation, was used to design the desired system. In order to allow respondents to freely express their thoughts, an interview guide containing open-ended questions will be provided. This approach will be chosen because it allowed the researcher to get thorough and detailed data. Another technique that will used to obtain accurate information is observation, albeit a time-consuming one. Here are some of the issues that Mutolere Hospital encountered with its manual file-based system in the past:

1. Inefficient Appointment Scheduling: The manual system have led to inefficiencies in appointment scheduling, resulting in long wait times for patients and difficulties in managing appointments for healthcare professionals.
2. Limited Access to Patient Records: Accessing and updating patient records in a manual file-based system is time-consuming and error-prone. This resulted in delays in accessing important medical information and coordinating patient care.
3. Data Redundancy and Duplication: Manual record-keeping system resulted in data redundancy and duplication, leading to inconsistencies and inaccuracies in patient information. This impact the quality of care and increase the risk of medical errors.
4. Difficulty in Information Retrieval: Finding and retrieving specific patient records or information from a manual filing system is challenging, especially as the volume of records increases over time. This have hindered timely decision-making and patient care delivery.
5. Lack of Real-Time Data Updates: Manual system lacks real-time data synchronization capabilities, leading to delays in updating patient records and sharing information between healthcare professionals. This may have compromised the continuity and coordination of care.
6. Limited Integration with Other Healthcare Facilities: Manual file-based system have limited integration with other healthcare facilities or external systems, making it difficult to exchange information and coordinate care across different healthcare providers.
7. Risk of Data Loss or Damage: Paper-based records are susceptible to damage, loss, or destruction due to environmental factors, such as fire, water damage, or physical wear and tear. This poses a significant risk to the security and integrity of patient information.
8. Inefficient Resource Utilization: The manual handling of patient records and administrative tasks are labor-intensive and resource-intensive. This have resulted in inefficiencies in staffing allocation and resource utilization within the hospital.
9. Difficulty in Monitoring and Reporting: Manual systems lack robust monitoring and reporting capabilities, making it challenging for hospital administrators to track key performance indicators, analyze trends, and generate reports for decision-making and quality improvement purposes.
10. Compliance and Regulatory Challenges: Maintaining compliance with healthcare regulations and standards, such as patient privacy laws (e.g., HIPAA), is more challenging with manual file-based systems due to the risk of unauthorized access or data breaches.

## 3.3 System requirements and specifications

These outline the functions and capabilities of the developed system. This lists the fundamental tasks that the intended system must complete, including computations, updates, and query processing. The terms user requirements, system requirements, functional and non-functional requirements are used to describe them.

## 3.3.1 User requirements

User requirements describe the needs and expectations of the system's end-users. They are focused on the functionality and usability of the system and how it supports the work of the Hospital. They are services the user expects the system to be able to do. These specifications allowed the researcher to identify the system's user-related features, enabling user approval of the created system.

1. User-Friendly Interface: The system should feature an intuitive and easy-to-navigate interface for both healthcare professionals and patients, facilitating seamless interaction and navigation.
2. Appointment Scheduling: Healthcare professionals should be able to schedule, modify, and cancel appointments efficiently, considering patient availability and medical priorities. Patients should have the ability to request, confirm, reschedule, or cancel appointments online through a user-friendly interface.
3. Real-Time Data Synchronization: The system should ensure real-time synchronization of patient data across different departments and healthcare facilities within Mutolere Hospital, enabling timely access to up-to-date information.
4. Integration with Electronic Health Records (EHRs): The system should integrate seamlessly with Mutolere Hospital's existing electronic health records (EHR) system, enabling healthcare professionals to access patient medical histories, diagnoses, and treatment plans during consultations.
5. Appointment Reminders and Notifications: Patients should receive automated appointment reminders and notifications via their preferred communication channels (e.g., email, SMS), reducing the likelihood of missed appointments and improving attendance rates.
6. Customization and Personalization: The system should allow for customization and personalization based on user preferences and roles, ensuring that healthcare professionals and patients have access to relevant features and information.
7. Accessibility Features: The system should incorporate accessibility features, such as adjustable font sizes, screen readers, and language options, to accommodate users with diverse needs, including those with disabilities or language barriers.
8. Secure Data Storage and Privacy Protection: The system should adhere to strict security protocols and encryption standards to safeguard patient data and ensure compliance with healthcare privacy regulations (e.g., HIPAA). Access controls should be implemented to restrict unauthorized access to sensitive patient information and ensure confidentiality.
9. Reporting and Analytics: The system should provide robust reporting and analytics capabilities, allowing hospital administrators to track key performance indicators (KPIs), analyze trends, and generate insights for decision-making and quality improvement purposes.
10. Training and Support: Comprehensive training and support should be provided to healthcare professionals and administrative staff to ensure proficiency in using the system effectively. Ongoing technical support and troubleshooting assistance should be available to address user queries and issues in a timely manner.

### 3.3.2 Functional requirements

Functional requirements describe the specific functionality that the system must provide to meet the needs of the organization. They define what the system must do to be considered successful.

The functional requirements include;

1. User Authentication and Authorization: The system should allow users to authenticate securely using unique identifiers (e.g., username and password) and enforce role-based access control to ensure appropriate access levels for different user groups (e.g., healthcare professionals, administrative staff, patients).
2. Appointment Scheduling: Healthcare professionals should be able to schedule, modify, and cancel appointments, specifying appointment types, dates, times, and durations. Patients should have the ability to request, confirm, reschedule, or cancel appointments through an intuitive user interface, with real-time availability updates.
3. Electronic Health Records (EHR) Integration: The system should seamlessly integrate with Mutolere Hospital's existing electronic health records (EHR) system, allowing healthcare professionals to access and update patient medical histories, diagnoses, treatment plans, and medication records during consultations.
4. Automated Appointment Reminders: The system should generate automated appointment reminders and notifications for patients, sent via their preferred communication channels (e.g., email, SMS), to reduce appointment no-shows and improve attendance rates.
5. Data Synchronization and Backup: The system should ensure real-time synchronization of patient data across different departments and healthcare facilities within Mutolere Hospital, with regular data backups to prevent data loss and ensure data integrity.
6. Customization and Personalization: Users should be able to customize and personalize their user interface settings, such as language preferences, notification preferences, and accessibility options, to enhance user experience and accommodate individual needs.
7. Accessibility Features: The system should incorporate accessibility features, such as adjustable font sizes, screen readers, and language options, to support users with disabilities or language barriers and ensure equitable access to healthcare services.

### 3.3.3 System requirements

Hardware resources and other software resources must be present on a computer for all computer software to be used effectively. The designed system for Mutolere Hospital required to meet certain requirements, which include the following;

1. Hardware Requirements: Server infrastructure: Sufficient hardware resources, including processing power, memory, and storage capacity, are essential to host the system's software components and accommodate anticipated user load. Additionally, client devices must be compatible, such as desktop computers, laptops, tablets, and smartphones, with internet connectivity to access the system's web-based interface or mobile application.
2. Software Requirements: Operating system compatibility is necessary to support the system's software components on compatible operating systems (e.g., Windows, Linux, macOS) for both server and client devices. Moreover, compatibility with commonly used web browsers (e.g., Google Chrome, Mozilla Firefox, Safari, Microsoft Edge) is required for accessing the system's web-based interface. Integration with a reliable and scalable database management system (e.g., MySQL, PostgreSQL, MongoDB) is crucial to store and manage patient records, appointment data, and system configurations. Additionally, utilization of appropriate programming languages, development frameworks (e.g., Django, Ruby on Rails, Laravel), and libraries is necessary to build and maintain the system's software components.
3. Network Requirements: Stable and high-speed internet connectivity is necessary to ensure seamless access to the system's web-based interface or mobile application from both internal and external network environments. Configuration of the hospital's internal network infrastructure is essential to support communication between server and client devices, ensuring low latency and reliable data transmission. Implementation of network security measures, such as firewalls, intrusion detection systems, and encryption protocols, is vital to protect sensitive data and prevent unauthorized access or data breaches.
4. Security Requirements: Implementation of secure authentication mechanisms (e.g., username/password, multi-factor authentication) and role-based access control is essential to ensure only authorized users can access the system and perform specific actions. Data encryption of data transmission and storage using secure encryption algorithms (e.g., SSL/TLS, AES) is crucial to protect patient information and maintain confidentiality. Conducting periodic security audits and vulnerability assessments is necessary to identify and mitigate potential security risks, ensuring compliance with healthcare privacy regulations (e.g., HIPAA, GDPR).
5. Scalability and Performance Requirements: Designing the system architecture to scale horizontally or vertically is necessary to accommodate increased user load, data volume, and system demands over time. Performance optimization through efficient resource utilization, database indexing, caching mechanisms, and code optimization techniques is crucial to minimize latency and response times.

## 3.3.4 Non-functional requirements

Non-functional requirements describe the overall qualities and characteristics that the system must possess, such as performance, reliability, and usability. These requirements help to ensure that the system meets the organization's overall goals and objectives.

1. Performance: The system should respond to user interactions within acceptable time limits, ensuring a smooth and responsive user experience. Additionally, it should handle a high volume of concurrent users and transactions without degradation in performance. The system architecture should support horizontal and vertical scaling to accommodate growth in user base and data volume over time.
2. Usability: The user interface should be intuitive, user-friendly, and accessible to users of varying technical proficiency. Navigation within the system should be logical and intuitive, allowing users to easily find and access desired functionalities. The system should also provide clear and informative error messages to guide users in resolving issues or errors encountered during system use.
3. Reliability: The system should be available for use during planned maintenance windows and minimize unplanned downtime through redundancy and fault tolerance measures. It should also be resilient to hardware failures, software crashes, and other disruptions, ensuring uninterrupted service delivery. Moreover, the system should maintain the integrity of patient data and ensure accuracy, consistency, and completeness of information stored within the system.
4. Security: Patient data should be encrypted during transmission and storage to prevent unauthorized access or disclosure. The system should employ robust authentication mechanisms and role-based access control to restrict access to sensitive information based on user roles and permissions. Additionally, it should log user activities, including access to patient records and system configurations, to facilitate accountability and traceability.
5. Compliance: The system should comply with relevant healthcare regulations and standards, such as HIPAA, GDPR, and local data protection laws. It should also support interoperability with external systems and standards to facilitate data exchange and communication with other healthcare providers and systems.

## 3.4 System Design and Modelling Methods

System design and modeling methods involve the process of conceptualizing, planning, and designing the architecture, components, and functionalities of a system. These methods aim to translate requirements into a structured and cohesive design that meets the objectives of the system. Here are some commonly used system design and modeling methods

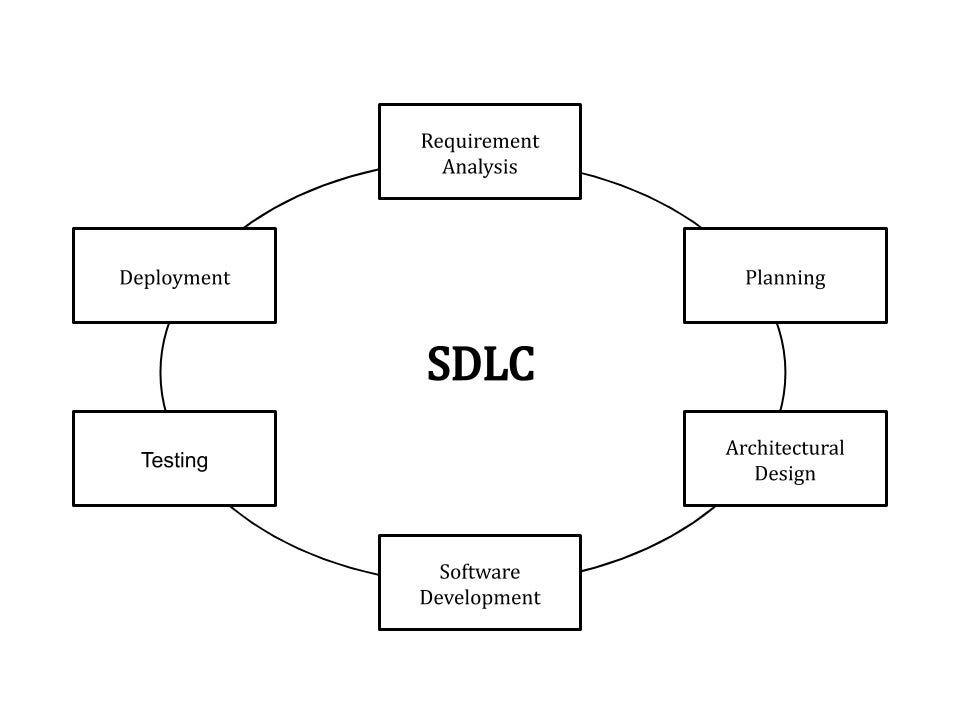


Figure 1Software development life cycle

The software development life cycle (SDLC) is a process followed by software development teams to design, develop, test, deploy, and maintain software. It is a structured approach to software development that helps ensure that software is developed in a consistent, reliable, and predictable manner. The stages of the software development life cycle typically include:

1. Requirements gathering: In this stage, the team works with the client or end-users to gather and document the requirements for the software.
2. Design: In this stage, the team creates a detailed design of the software, including its architecture, interface, and data structures.
3. Implementation: In this stage, the team writes and tests the code for the software.
4. Testing: In this stage, the team tests the software to ensure that it meets the requirements and functions as expected.
5. Testing: In this stage, the software is deployed to the production environment and made available to users.
6. Maintenance: In this stage, the team continues to support the software by fixing bugs, making improvements, and providing ongoing technical support.

### 3.4.1 System Design and Modelling using entity-relationship diagrams (E-R)

Entity-relationship diagrams (ERDs) are a type of graphical representation that are commonly used in system design and modelling. They provide a visual representation of the entities and relationships within the asset management and tracking system and are particularly useful for database design. They are used to represent the entities and relationships involved in tracking and managing assets. For example, the entities could include Asset, Employee, Department, Location, Vendor, and Maintenance Record. The relationships between these entities can then be represented as lines connecting the entities, with the type of relationship being indicated by the type of line and the cardinality of the relationship being indicated by the use of symbols, for example a diamond at the end of the line. ERDs can be a useful tool in the system design and modelling process as they provide a clear, visual representation of the entities and relationships involved in the system. This can help to identify any potential issues or areas for improvement early in the process, as well as provide a clear understanding of the data requirements for the system.

## 3.4.2 System Design and Modelling using Data Flow Diagrams (DFDs)

Data Flow Diagrams (DFDs) are another type of graphical representation that are commonly used in system design and modelling. They provide a visual representation of the flow of data within the Appointment system, from the sources of data to the end points where the data is used. A DFD is used to represent the flow of data involved in tracking and managing assets. For example, a DFD could show the flow of data from a user entering information about a new asset into the system, to the storage of that information in a database, to the retrieval of that information by other parts of the system as needed. DFDs can be a useful tool in the system design and modelling process as they provide a clear, visual representation of the flow of data within the system. This can help to identify potential areas for improvement in the flow of data, as well as provide a clear understanding of the data requirements for the system.

## 3.5 System implementation, testing and validation methods

### 3.5.1 System implementation methods

The software development process's System Implementation phase is when the created system is set up, installed, and made available in a live environment. Usually, this phase is divided into multiple steps, such as:

1. Installation: Installing the software and any necessary hardware components.
2. Configuration: Configuring the system to meet the specific requirements of the Hospital. This can include setting up user accounts, defining security settings, and configuring any system-wide settings.
3. Data Migration: Importing existing data into the system, either by manual data entry or by automated data migration.
4. Testing: Conducting comprehensive testing of the system to ensure it meets the specified requirements and works as expected.
5. User Acceptance Testing: Conducting testing of the system by end-users to ensure it meets their requirements and is user-friendly.
6. Deployment: Deploying the system into a live environment and making it available for use.
7. Training: Providing training to users on how to use the system, including any new processes or procedures that may have been introduced as part of the implementation.
8. Support and Maintenance: Providing ongoing support and maintenance of the system to ensure its continued operation and to address any issues that arise.

The success of the System Implementation phase is critical to the success of the overall project, as it is the stage where the developed system is put into use and the benefits of the project are realized. Careful planning and execution of each of the steps listed above is important to ensure a smooth and successful implementation. MySQL, PHP scripts, HTML, and CSS are used in the development of an asset management and tracking system as follows:

MySQL: MySQL is a popular open-source relational database management system that was used to store the data for the asset management and tracking system. The database could store information such as asset details, location, maintenance history, etc.

PHP scripts: PHP is a server-side scripting language that can be used to build the back-end of the asset management system. PHP scripts was used to retrieve data from the database, perform calculations, and generate dynamic web pages.

HTML: HTML (Hypertext Markup Language) is a standard markup language used to create web pages. HTML was used to create the structure and layout of the web pages that make up the user interface of the asset management system.

CSS: CSS (Cascading Style Sheets) is a style sheet language used to describe the look and formatting of a document written in HTML. CSS was used to control the appearance of the pages in the user interface, such as colors, fonts, and spacing.

Together, these technologies were used to build a complete Expert Medical System for Consultation and Appointment at Mutolere Hospital that has a user-friendly interface, can store and retrieve data efficiently, and can be easily maintained and updated as needed.

### 3.5.2 System testing methods

Testing procedures in the creation of an appointment and consultation system entailed assessing the system to make sure it satisfies the predetermined specifications and functions as intended. The system testing techniques employed were as follows:

Functional Testing: Functional Testing involved evaluating the system to ensure that it performed the required functions correctly and as specified. This type of testing involved verifying that the system is able to perform tasks such as data entry, data retrieval, data processing, and reporting.

Performance Testing: Performance Testing involved evaluating the system to ensure that it performed efficiently and within acceptable performance limits. This type of testing included load testing, stress testing, and scalability testing.

Security Testing: Security Testing involved evaluating the system to ensure that it was secure and that the data stored in it was protected. This type of testing included penetration testing and vulnerability scanning.

User Acceptance Testing: User Acceptance Testing involved evaluating the system with actual users to ensure that it met their requirements and is user-friendly. This type of testing was done before the system was deployed into a live environment.

Unit Testing: Unit Testing is the testing of individual components or modules of the system to ensure that they function as expected. This was done as part of the development process.

Integration Testing: Integration Testing involves testing the interaction between different components or modules of the system to ensure that they work together seamlessly.

System Testing: System Testing involved evaluating the complete system to ensure that it met the specified requirements and work as intended. This included functional testing, performance testing, security testing, and user acceptance testing.

System testing is a crucial step in the software development process that makes that the system functions as intended and satisfies the requirements. Good testing can guarantee a successful system deployment by assisting in the early detection of any problems during the development phase.

### 3.5.3 System validation methods

System validation is the process of assessing the system to make sure it satisfies the criteria and functions as planned in an actual setting. Some of the techniques utilized for system validation included the following:

System validation is an important part of the software development process and helps to ensure that the system meets the specified requirements and works as intended in the real-world environment. Effective validation helped to identify any issues early in the implementation process and helped to ensure a successful deployment of the system.

Security Validation, which involved evaluating the system to ensure that it was secure and that the data stored in it was protected in a real-world environment. This type of testing included penetration testing and vulnerability scanning.

Data Validation, which involved evaluating the system to ensure that the data entered into it was accurate, consistent, and complete. This type of testing included data integrity testing and data quality testing.

## 3.6 Chapter summary

This chapter delineates the research methodology employed in this study, encompassing a mixed-methods approach comprising both qualitative and quantitative data collection techniques. Utilizing the interview method, direct interactions between the researcher and participants facilitated a profound understanding of the subject matter, offering rich data insights and fostering rapport. Despite its time and resource-intensive nature, interviews provided flexibility and depth in exploring complex topics. Additionally, observational methods offered naturalistic insights into behaviors and interactions, although subjectivity and ethical considerations presented challenges. System study methods revealed issues faced by Mutolere Hospital with its manual system, informing the development of user requirements, functional requirements, system requirements, and non-functional requirements for the proposed system. Employing system design and modeling methods such as entity-relationship diagrams and data flow diagrams aided in conceptualizing the system architecture. System implementation involved installation, configuration, data migration, and comprehensive testing methods to ensure quality and reliability. Validation techniques, including security and data validation, were employed to assess system integrity in real-world environments. This rigorous research methodology ensures the validity and reliability of the study's outcomes, providing a robust framework for the development and implementation of an expert medical system for consultation and appointment at Mutolere Hospital.

CHAPTER THREE: RESEARCH METHODOLOGY

3.0 Introduction: This chapter provides a detailed overview of the research methodology employed for the development of the e-commerce website, E-Shop. It encompasses a mixed-methods approach, combining qualitative and quantitative data collection techniques to ensure a comprehensive understanding of the project's requirements and objectives.

3.1 Data Collection and Analysis Methods: In this section, various methodologies were employed to gather essential data and insights for the development of E-Shop. These methodologies included:

3.1.1 Customer Interviews: Direct interactions with potential customers were conducted to gather firsthand information, preferences, and insights regarding their shopping habits, needs, and expectations from an online shopping platform. Through structured questioning, valuable insights were obtained to tailor E-Shop's features and functionalities according to customer preferences.

3.1.2 Market Observation: Observational methods were employed to analyze market trends, competitor strategies, and consumer behavior in the e-commerce industry. By systematically observing market dynamics and customer interactions with existing platforms, valuable insights were gained to inform the design and development of E-Shop.

3.2 Market Analysis and Requirement Identification: This section delves into the analysis of market trends, competitor offerings, and customer preferences to identify the key requirements for E-Shop. By examining existing e-commerce platforms and consumer feedback, essential features and functionalities were identified to ensure E-Shop's competitiveness and appeal to target customers.

3.3 System Requirements and Specifications: Based on the market analysis and requirement identification, the following key system requirements and specifications were identified for E-Shop:

3.3.1 User Requirements:

* Intuitive and user-friendly interface for seamless navigation and shopping experience.
* Efficient product search and filtering options to facilitate product discovery.
* Secure payment gateways and checkout process to ensure transaction security and trust.
* Personalized recommendations and tailored shopping experiences based on user preferences and behavior.
* Mobile responsiveness for accessibility across various devices and platforms.

3.3.2 Functional Requirements:

* Product catalog management for easy addition, editing, and categorization of products.
* Shopping cart functionality with options for item management and order tracking.
* User account management for registration, login, and profile customization.
* Order management system for processing, tracking, and managing customer orders.
* Integration with third-party logistics providers for efficient order fulfillment and delivery.

3.3.3 System Requirements:

* Scalable and reliable hosting infrastructure to support growing user traffic and data volume.
* Compatibility with major web browsers and mobile devices for broad accessibility.
* Integration with secure and scalable database systems for efficient data storage and retrieval.
* Implementation of robust security measures to safeguard user data and transaction integrity.
* Ongoing maintenance and support mechanisms to ensure system stability and performance.

3.4 System Design and Modeling: This section outlines the system design and modeling methods utilized for the development of E-Shop, including:

3.4.1 User Interface Design:

* Creation of visually appealing and intuitive user interfaces to enhance user experience and engagement.
* Implementation of responsive design principles to ensure compatibility across various devices and screen sizes.

3.4.2 Database Design:

* Development of efficient database schemas and structures to support E-Shop's data storage and retrieval requirements.
* Optimization of database queries and indexing to enhance performance and scalability.

3.5 System Implementation, Testing, and Validation Methods: This section describes the implementation, testing, and validation processes employed to ensure the reliability, functionality, and security of E-Shop:

3.5.1 Implementation:

* Installation and configuration of E-Shop's software components and infrastructure.
* Development and integration of core features and functionalities according to the defined requirements.
* Migration of existing data and content to the new platform, ensuring data integrity and consistency.

3.5.2 Testing:

* Comprehensive testing of E-Shop's features and functionalities to identify and resolve any bugs or issues.
* Performance testing to assess system responsiveness, scalability, and load-handling capabilities.
* Security testing to identify and address potential vulnerabilities and ensure transaction security.

3.5.3 Validation:

* User acceptance testing to evaluate E-Shop's usability and functionality from an end-user perspective.
* Security validation to verify the effectiveness of implemented security measures in protecting user data and transactions.
* Data validation to ensure the accuracy, consistency, and integrity of data stored and processed by E-Shop.

3.6 Chapter Summary: In summary, this chapter outlines the research methodology employed for the development of E-Shop, encompassing a comprehensive approach to gather, analyze, and implement key requirements and specifications. Through a combination of qualitative and quantitative methods, E-Shop was designed and implemented to meet the evolving needs and expectations of online shoppers, ensuring a seamless and secure shopping experience.

Top of Form