STUDENT INDUSTRIAL WORK EXPERIENCE SCHEME (SIWES) REPORT

BY

GEORGE, FAVOUR OKON
20/SC/CO/898
DEPARTMENT OF COMPUTER SCIENCE
FACULTY OF COMPUTING
UNIVERSITY OF UYO, AKWA IBOM STATE

SUBMITTED TO

DEPARTMENT OF COMPUTER SCIENCE
FACULTY OF COMPUTING
UNIVERSITY OF UYO, AKWA IBOM STATE

IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF A BACHELOR OF SCIENCE (B.SC) IN COMPUTER SCIENCE

SEPTEMBER, 2024

DEDICATION

I dedicate my dissertation work to my family and many friends. A special feeling of gratitude to my loving mother, Mrs. Ime George, whose words of encouragement and push for being better ring in my ears.

I also dedicate this dissertation to my friends all around me who have supported me throughout the process. I will always appreciate all they have done.

I extend my deep appreciation to my lecturers, mentors, and tutors who have guided and supported me throughout my academic journey.

Finally, I dedicate this work and give special thanks to God.

ACKNOWLEDGMENT

I extend my heartfelt gratitude to the Industrial Training Fund for their visionary initiative in establishing this invaluable program.

My sincere thanks to Strategic Hub Limited for equipping me with the essential skills and exposure needed to excel in my field. I am especially grateful to my industrial supervisor, Mr. Temitope Adeosun, for his dedicated mentorship and guidance, which have been instrumental in honing my abilities.

I am deeply appreciative of my family for their unwavering financial support throughout my industrial training. My profound thanks go to God Almighty, the source of all wisdom, knowledge, and understanding, without whom none of this would have been possible.

To my Head of Department, esteemed lecturers, SIWES Coordinator, and the entire staff and students of the Computer Science Department at the University of Uyo, I am truly grateful for your collective efforts in shaping my academic journey and providing a solid foundation upon which my internship was built.

TABLE OF CONTENT

ABSTRACT

The Students Industrial Work Experience Scheme (SIWES), established by the Federal Government of Nigeria, is designed to equip students from higher institutions with essential industrial skills and practical experience in their chosen fields of study. It also aims to prepare students for the professional work environments they will encounter after graduation.

This technical report documents the experiences and knowledge I gained during my six-month industrial training at Strategic Hub Limited, Port Harcourt, Rivers State. It provides an overview of the operations at Strategic Hub Limited and details my personal experience working there.

I had the privilege of working under the mentorship of Mr. Temitope Adeosun, a seasoned and well-rounded supervisor. Under his guidance, I developed skills in web programming and acquired valuable soft skills pertinent to my field. Alongside fellow interns, I collaborated on several projects, further enhancing my practical knowledge.

This report not only captures the skills and insights I gained but also addresses the challenges I encountered. The report is organised into nine chapters, each focusing on a distinct aspect of Computer Science.

CHAPTER ONE

INTRODUCTION TO VISUAL APPLICATION

Visual applications are pivotal in translating complex data into understandable visual formats. These applications leverage visual computing technologies to enhance user experience, data interpretation, and interaction. As the need for intuitive data representation grows, visual applications are becoming increasingly integral across various industries, including healthcare, finance, and education.

1.1 UNDERSTANDING VISUAL COMPUTING

Visual computing encompasses a range of techniques and technologies used to process, analyze, and render visual data. It merges elements of computer science, computer graphics, and computer vision. The goal is to create visual representations that can simplify data analysis, improve user interactions, and support decision-making processes. This field includes various sub-disciplines, such as image processing, 3D modeling, and visual analytics.

1.2 APPLICATION DOMAINS OF VISUAL COMPUTING

Visual computing plays a pivotal role across various industries and applications:

- **GUI Development**: Enables intuitive, user-friendly interfaces for seamless interaction with software applications.
- **Modeling and Animation**: Involves creating and animating computer-generated faces and characters, used in training tools, sketch-based modeling, and geometric animation.

- **Digital Heritage Preservation**: Utilizes augmented and virtual reality to capture and recreate 3D data, preserving archaeological artifacts and historical environments.
- **Complex System Simulation**: Supports simulations of systems like biological cells and transportation networks, using interactive visualization for analysis.
- **High-Performance Computing**: Employs GPUs and advanced architectures for efficient, real-time processing of large-scale visual data.
- **Interdisciplinary Research**: Integrates visual computing with GPU computing and simulation methodologies, impacting fields like software engineering.

1.3 OVERVIEW OF VISUAL APPLICATIONS

Visual applications are software programs that utilize a Graphical User Interface (GUI), allowing users to interact through clickable icons and visual elements. Unlike command-line interface (CLI) applications that require text-based commands, GUIs offer a more intuitive and user-friendly experience by enabling direct interaction with graphical components.

1.4 COMPONENTS OF A VISUAL APPLICATION

A graphical user interface (GUI) combines technologies and components to create an accessible platform for user interaction. It uses visual elements like windows, icons, menus, and pointers (WIMP) to make information easier to access, especially for non-technical users. These components simplify data visualization and management, enhancing the overall user experience.A typical visual application consists of several key components:

- **User Interface (UI)**: The part of the application that users interact with, including elements such as buttons, menus, controls, tabs, and icons.

    - **Menus**: Menus allow users to execute commands by selecting options from a list, accessible via a menu bar, pull-down menu, or context menu.

    - **Icons**: Icons are small graphical representations of files, programs, or commands, providing a quick way to execute actions and locate items.

- **Rendering Engine**: The rendering engine is responsible for generating visual output from the underlying data. It performs several critical functions:

    - **Graphics Drawing**: Converts data into visual elements like shapes and text, applying transformations and rendering high-quality visuals through techniques such as anti-aliasing.

    - **Animation**: Manages dynamic visual elements by updating them over time, handling frame rates, keyframes, and interpolation for smooth and realistic animations.

- **Data Processing Module**: This module is tasked with handling raw data and preparing it for visualization:
    - **Data Transformation**: Converts raw data into a structured format suitable for rendering. This might include aggregating, filtering, or normalizing data to present it effectively.

- **Interaction Module**: Facilitates user interactions within the application:
    - **Event Handling**: Manages input from various sources, such as mouse clicks, keyboard presses, or touch gestures.
    - **User Feedback**: Provides immediate visual or auditory feedback to users based on their interactions. This includes highlighting selected items, displaying tooltips, or playing sound effects to enhance usability.

- **Backend Infrastructure**: Supports the application's data management and storage needs

These components work together to create a seamless and interactive user experience, transforming data into meaningful visual representations

and managing the underlying processes that support effective application performance.

CONCLUSION

Visual applications play a crucial role in modern computing, offering intuitive ways to represent and interact with complex data. By leveraging visual computing technologies, these applications enhance user experience and support a wide range of industries. Understanding the components and domains of visual computing provides valuable insights into the development and implementation of effective visual applications.

REFERENCE

1. https://en.wikipedia.org/wiki/Visual_computing

2. https://en.wikipedia.org/wiki/Graphical_user_interface

3. https://en.wikipedia.org/wiki/List_of_graphical_user_interface_elemen

CHAPTER TWO

ADVANCED WEB PROGRAMMING

2.0 INTRODUCTION

Web programming is essential for creating and managing websites and web applications. It involves writing code for software that runs on web servers and is accessed via browsers. This field covers various technologies and practices, from designing user interfaces to implementing server-side logic and managing data interactions, enabling the development of dynamic and interactive online experiences.

2.1 WEB PROGRAMMING

Web programming relies on three core languages: HTML for structuring content, CSS for visual presentation, and JavaScript for interactivity and dynamic behavior. On the server side, backend frameworks like Flask, Django, and Node.js handle business logic, data management, and content delivery, ensuring applications are functional, scalable, and maintainable.

2.2 WEB PROGRAMMING LANGUAGES

Web programming languages are essential tools for building and designing web pages and applications. The primary languages used in web development include HTML, CSS, and JavaScript. Each language serves a distinct purpose and plays a crucial role in creating a complete web experience.

2.2.1 HTML (HyperText Markup Language)

HTML is the foundational language for creating web pages. It provides the structure and content of a webpage through a series of elements and tags.

An HTML element consists of three parts: an opening tag, the content, and a closing tag. Attributes are used to provide further information about the element and are included within the opening tag.

2.2.2 CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets) is a stylesheet language used to control the presentation and layout of web pages written in HTML or XML.

CSS works by selecting HTML elements and applying styles to them. Styles are defined in a set of rules, each consisting of a selector and a declaration block. The selector targets specific HTML elements, while the declaration block contains one or more declarations that specify the style properties and their values.

Sample CSS Syntax

```
/* CSS Rule */
 selector {
 property: value;
 }
```

Example Usage

```
/* Style for all paragraph elements */
p {
color: blue;
 font-size: 16px;
 }
```

In this example, the `p` selector targets all paragraph (`<p>`) elements, setting their text color to blue and font size to 16 pixels.

Adding CSS Styles to an HTML Document

CSS (Cascading Style Sheets) can be added to an HTML document in three primary ways: inline styles, internal styles, and external styles. Each method has its specific use cases and advantages.

1. Inline Styles

Definition: Inline styles are CSS rules applied directly to individual HTML elements using the `style` attribute. This method allows you to style a specific element without affecting other elements on the page.

Sample Syntax

```
<tagname style="property: value;">
  Content
</tagname>
```

Example Usage

```
<p style="color: red; font-size: 20px;">
 This is a red-colored text with a font size of 20 pixels.
 </p>
```

2. Internal Styles

Definition: Internal styles are CSS rules defined within the `<style>` tag in the `<head>` section of an HTML document. This method applies styles to elements across the entire document.

Example Usage

```
<!DOCTYPE html>
<html>
<head>
  <style>
    body {
      font-family: Arial, sans-serif;
    }
    h1 {
      color: blue;
    }
  </style>
</head>
<body>
  <h1>This is a blue heading</h1>
  <p>This is a paragraph with Arial font.</p>
</body>
</html>
```

3. External Styles

Definition: External styles are CSS rules defined in a separate `.css` file, which is linked to the HTML document using the `<link>` tag in the `<head>` section. This method is ideal for maintaining consistent styles across multiple web pages.

Sample Syntax

```
<!DOCTYPE html>
<html>
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <!-- HTML content -->
</body>
</html>
```

These methods of adding CSS to an HTML document offer flexibility and scalability for styling web pages, each suited to different needs and project sizes. For best practices, using external stylesheets is generally recommended for larger projects due to its maintainability and reusability.

2.2.3 JavaScript

JavaScript is a versatile scripting language used to add interactivity and dynamic features to web pages. It enables client-side scripting, enhancing user engagement through responsive elements and functionalities.

Sample Syntax

```
// Basic JavaScript Function
function greetUser(name) {
    alert('Hello, ' + name + '!');
}

// Event Listener
document.getElementById('greetButton').addEventListener('click', function() {
    var userName = document.getElementById('nameInput').value;
    greetUser(userName);
});
```

2.2.4 Backend Frameworks

Backend frameworks provide the server-side functionality for web applications, handling data processing, business logic, and communication with databases. One popular backend framework is Flask, a lightweight framework for Python.

Flask

Flask is a microframework for Python that allows developers to build web applications with minimal overhead. It is known for its simplicity and flexibility.

Sample Syntax

```
from flask import Flask, render_template, request, redirect, url_for
app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/submit', methods=['POST'])
def submit():
    user_name = request.form['name']
    return redirect(url_for('greeting', name=user_name))
```

```
@app.route('/greeting/<name>')
def greeting(name):
    return f'Hello, {name}!'

if __name__ == '__main__':
    app.run(debug=True)
```

2.2.5 Databases

Databases are crucial for storing and managing data in web applications. SQL (Structured Query Language) is a standard language for interacting with relational databases.

SQL (Structured Query Language)

SQL is used to perform various operations on a database, including querying, updating, and managing data.

Sample Syntax

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2.2.6 Web Servers and Hosting

Web servers and hosting are essential for making web applications accessible to users over the Internet. They handle requests from clients, serve content, and manage application deployment.

Web Servers

Web servers, such as Apache and Nginx, process incoming HTTP requests and deliver web pages to users. They also manage server-side scripting and handle file uploads.

Hosting

Web hosting services provide the infrastructure needed to deploy and maintain web applications. They offer various plans, including shared hosting, VPS (Virtual Private Server), and dedicated servers. Popular examples of web hosting

services include: Amazon Web Services (AWS), Google Cloud Platform (GCP), DigitalOcean etc

Key Considerations for Hosting:

- **Performance**: Ensure the hosting service offers sufficient resources to handle the application's traffic and load.
- **Scalability**: Choose a provider that allows for easy scaling as your application's needs grow.
- **Security**: Look for features like SSL certificates, firewalls, and regular backups to protect your application and data.

Backend frameworks like Flask handle server-side processing, SQL manages data, while CSS and JavaScript enhance web page presentation and interactivity. JavaScript also updates content in real time, and HTML5 adds features like canvas, audio, video, and geo-location for rich, interactive web experiences.

CONCLUSION

As web programming continues to evolve, developers are presented with new challenges and opportunities to innovate. From enhancing user experiences with cutting-edge technologies to ensuring the security and efficiency of applications, web programming remains a dynamic and integral part of the technological landscape.

REFERENCES

Blanck,peter.ThestruggleforWebAccessibilitybypersonswithCognitive Disability, Cambridge Disability law and policy series, 2015.

Burgstahle,Sheryl.UniversalDesigninHigherEducation:FromPrinciplesto Practice, Harvard Education press, 2008.

Byrne, jim.60 hot to touch Accessible Web Design tips- the tips no web developer can live without! .Jim Byrne, 2006.

Sand Shin. JavaScript Basics &JavaScript Basics & HTML DOM. SunMicrosystems, Inc Sun Microsystems, Inc., 2003.

CHAPTER THREE

SOFTWARE DESIGN TECHNIQUES AND METHODS

3.0 INTRODUCTION

Software design is a critical phase in the software development lifecycle (SDLC) that focuses on defining the architecture and components of a software system. Effective software design ensures that the system meets user requirements, is maintainable, scalable, and efficient. This report explores various software design techniques and methods, providing insights into their application, benefits, and considerations.

3.1 CONCEPT OF SOFTWARE DESIGN

1. Object-Oriented Design (OOD)

Object-Oriented Design (OOD) is a paradigm that shifts the focus from functions to entities within a software system. This approach revolves around objects and their interactions, providing a framework that models real-world entities and their relationships.

1.1 Key Concepts of Object-Oriented Design

Objects: Entities like Person or Bank, combining data (attributes) and methods.

Classes: Blueprints defining attributes and methods for objects.

Encapsulation: Bundling data and methods in a class.

Inheritance: Creating new classes from existing ones, promoting reusability.

Polymorphism: Allows methods to share a name but operate differently based on input, enabling flexibility.

2. Structured Design

Structured Design is a systematic approach that focuses on breaking down a problem into smaller, more manageable components. It emphasizes the creation of a clear and organized design to address complex problems.

2.1 Key Characteristics

Divide and Conquer: Structured design employs a "divide and conquer" strategy, decomposing a complex problem into smaller sub-problems that can be solved independently. This modular approach simplifies the design and implementation processes.

Modularization: The design is organized into modules, each addressing a specific aspect of the problem. Modules interact with each other according to predefined communication rules, such as:

- **Cohesion:** Refers to the degree to which elements within a module are related. High cohesion indicates that a module performs a single, well-defined task.
- **Coupling:** Describes the level of dependency between modules. Low coupling means that modules interact with minimal dependencies, enhancing flexibility and maintainability.

3. Function-Oriented Design

Function-Oriented Design focuses on decomposing a system into functions or procedures that perform specific tasks. This approach views the system as a composition of these functions.

3.1 Key Characteristics

Function Decomposition: Breaks system into smaller, independent functions for abstraction and information hiding.

Information Passing: Functions share data via parameters or global variables, requiring careful data flow management.

State Management: Functions modify system state, suitable when operations rely more on input than system state.

4. Design Process

The software design process involves a series of steps to create a solution that meets the identified needs. Although the process varies depending on the design approach, it generally includes:

Requirements Analysis: Understanding and documenting the needs and constraints of the system.

Design Formulation: Creating a detailed plan that addresses the requirements, considering various design techniques and methodologies.

Implementation and Testing: Developing and testing the design to ensure it meets the specified requirements.

Evaluation and Refinement: Assessing the design and making necessary adjustments based on feedback and testing outcomes.

5. User Interface (UI)

The User Interface (UI) is the point of interaction between the user and the software system. It can be broadly categorized into two types:

- **Graphical User Interface (GUI):** Uses visual elements like windows, icons, and menus for user-friendly interaction.
- **Command Line Interface (CLI):** Text-based interface where users type commands, offering efficiency for experienced users.

3.2 METHODS AND TECHNIQUES FOR SOFTWARE DESIGN

3.2.1 Agile Design

Definition: Agile design is an iterative approach that emphasizes flexibility, collaboration, and incremental delivery.

Advantages:

- Promotes adaptability to changing requirements.
- Enhances collaboration and communication within development teams.

Disadvantages:

- May require frequent changes to design and implementation.
- Can be challenging to manage scope and priorities.

3.2.2 Waterfall Model

Definition: The Waterfall model is a linear and sequential approach to software design and development. It involves distinct phases such as requirements analysis, design, implementation, testing, and maintenance.

Advantages:

- Provides a clear structure and documentation for each phase.
- Suitable for projects with well-defined requirements.

Disadvantages:

- Inflexible to changes once a phase is completed.
- Can lead to lengthy development cycles and delays.

3.2.3 Iterative and Incremental Design

Definition: Iterative and incremental design involves developing software in small, manageable increments. Each increment adds new functionality

or improvements, and iterations refine and enhance the system based on feedback.

Advantages:

- Allows for continuous feedback and improvement.
- Supports gradual delivery of functionality and early detection of issues.

Disadvantages:

- Requires ongoing management of iterations and increments.
- May lead to scope creep if not carefully controlled.

3.2.4 Extreme Programming (XP)

Definition: Extreme Programming (XP) is an Agile methodology that focuses on technical excellence and customer satisfaction. It emphasizes frequent releases, continuous feedback, and collaboration.

Advantages:

- Enhances code quality and responsiveness to changes.
- Encourages collaboration and continuous improvement.

Disadvantages:

- May require significant changes to team dynamics and practices.
- Can be demanding in terms of time and effort for frequent testing and iteration.

CONCLUSION

Software design techniques and methods play a crucial role in the development of robust, scalable, and maintainable software systems.Understanding and applying these techniques and methods can significantly impact the success of software projects and meet the evolving needs of users and stakeholders.

REFERENCES

Barry, B .A Spiral Model of Software Development and Enhancement. Derey publishing ltd, USA,1996. and When to Use Them",2002 www.business- esolutions.com.

BarryBoehm,"SpiralDevelopment:Experience,Principles,andRefinements

.hashirepress,Indonesia,2000.

CHAPTER FOUR

COMPUTER AND SOCIETY

4. INTRODUCTION

Overview of the Relationship between Computers and Society
The integration of computers into society has revolutionized almost every aspect of modern life. From communication and education to healthcare and entertainment, computers have transformed how individuals and organizations operate, creating a deeply interconnected world. This chapter explores the multifaceted relationship between computers and society, examining both the benefits and challenges posed by this technological evolution.

Importance of Understanding the Impact
Understanding the impact of computers is crucial for shaping policies and practices. Key issues include balancing privacy concerns with data benefits and managing employment shifts due to automation, requiring strategies for retraining and reducing inequality.

4.1 THE IMPACT OF COMPUTERS ON SOCIETY

Computers have deeply influenced almost every aspect of modern society, transforming how we communicate, work, learn, entertain ourselves, and access information. Their impact is profound and far-reaching, affecting not only individual lives but also entire industries, economies, and cultures. This section delves into the various ways computers have shaped and continue to shape society.

4.1.1 Computers in Communication

The advent of computers and the internet has revolutionized communication, making it faster, more efficient, and more accessible. Before computers, communication was largely limited to face-to-face

interactions, postal mail, and telephones. Today, computers enable instant communication across vast distances, breaking down geographical barrier

Email and Instant Messaging: Email revolutionized digital communication by allowing instant message exchange, unlike traditional mail which is slower.

Social Media: Platforms like Facebook and Instagram have transformed global connectivity, enabling users to share content and build communities online.

Video Conferencing: Tools like Zoom and Microsoft Teams are vital for virtual meetings, reducing travel needs and facilitating remote work, especially during the COVID-19 pandemic.

4.1.2 Computers in Education

Computers have transformed education by making learning more accessible, personalized, and interactive.

E-Learning and Online Education: Platforms like Coursera and Khan Academy offer flexible, accessible online courses, allowing learners to study at their own pace from anywhere.

The Digital Divide: Unequal access to computers and the internet creates educational disparities. Bridging this gap is crucial to ensure all students benefit from digital learning opportunities.

4.1.3 Computers in the Workplace

The workplace has undergone a dramatic transformation due to the integration of computers, which have streamlined operations, improved productivity, and enabled new forms of work.

Remote Work and Collaboration: Remote work has transformed the modern workplace, using tools like cloud computing and video

conferencing to enable global talent access and reduce office space needs, impacting work-life balance and workplace culture.

Job Displacement and Reskilling: While computers have created new jobs in IT and software, they have also caused job displacement due to automation and AI, raising concerns about unemployment and the need for reskilling.

4.1.4 Computers in Healthcare

The healthcare industry has greatly benefited from computer technology, which has improved patient care, streamlined operations, and advanced medical research.

Electronic Health Records (EHRs)
Electronic Health Records have replaced paper-based medical records, providing a digital platform for storing and managing patient information.

Advanced Diagnostics and Treatment
Computers play a critical role in advanced diagnostics and treatment. Medical imaging technologies such as MRI and CT scans rely on computer algorithms to produce detailed images of the body's internal structures.

4.1.5 Computers and Entertainment

The entertainment industry has been profoundly impacted by computers, with digital media, video games, and streaming services becoming central to how people consume entertainment.

Digital Media and Streaming: The shift to digital media has revolutionized content distribution, with streaming services like Netflix and Spotify offering on-demand access and reducing reliance on physical media.

Video Games: The video game industry, driven by computers, has grown into a major cultural and economic force, evolving from simple games to complex, immersive experiences.

Virtual and Augmented Reality: VR and AR technologies create immersive environments that merge physical and digital worlds, transforming entertainment and interactive experiences.

CONCLUSION

As technology continues to advance, the relationship between computers and society will evolve. It is essential to foster an inclusive, ethical, and sustainable approach to computing that considers the diverse needs and values of all members of society.

REFERENCES

Jacobson,"ComputerCrimes",(2002)Vol.39AmericanCriminalLawReview,273.

Essays, UK. (November 2013). Importance of Computers in Society Information

Lucas, h.C: Information Systems: Concepts for management, McGraw-HillInc, New York, London, 1978.

CHAPTER FIVE

SYSTEM ANALYSIS AND DESIGN

5.0 INTRODUCTION

System Analysis and Design (SAD) plays a pivotal role in the development of effective information systems that cater to the dynamic needs of businesses and organizations. It involves a systematic approach to understanding the requirements of a system, designing solutions to meet those requirements, and ensuring that the final product is aligned with organizational goals.

5.1 KEY CONCEPTS

SYSTEM ANALYSIS

System Analysis is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. It is conducted for the purpose of studying a system or its parts in order to identify its objectives.

SYSTEM DESIGN

System Design is a process of planning a new business system or replacing an existing system by defining its components or modules to satisfy the specific requirements.

WHAT IS A SYSTEM

A system is defined as “a structured collection of interrelated components that work together according to a defined plan to achieve a specific objective.” For a system to function effectively, it must satisfy three fundamental criteria:

- **Structure and Behavior:** The system must possess a defined structure and exhibit behaviors that are aligned with achieving its predetermined goal.

- **Interconnectivity and Interdependence:** The components within the system must be interconnected and interdependent, ensuring that they work cohesively.
- **Priority of Objectives:** The overarching objectives of the organization take precedence over the individual goals of its subsystems.

Key Properties of a System

A well-designed system is characterized by several essential properties:

- **Organization:** Components are structured for effectiveness.
- **Interaction:** Components work together to enable system functionality.
- **Interdependence:** Components rely on each other for efficient operation.
- **Integration:** Components combine into a unified, goal-driven system.
- **Central Objective:** The system focuses on a key goal that guides all components.

5.2 PHASES OF SYSTEM DEVELOPMENT LIFE CYCLE (SDLC)

The System Development Life Cycle (SDLC) is a framework used to guide the development process of an information system. It consists of several phases, each with its own set of activities and deliverables.

5.2.1 Requirement Analysis: Gather and document system requirements from stakeholders using interviews, surveys, and observation to understand needs, expectations, and constraints.

5.2.2 System Design: Design the system's architecture, including hardware, software, user interfaces, and databases, and create detailed specifications for development.

5.2.3 Implementation: Develop the system by coding and using programming languages, frameworks, and tools based on the design specifications.

5.2.4 Testing: Evaluate the system through unit, integration, and user acceptance testing to ensure it meets requirements and functions correctly.

5.2.5 Deployment and Maintenance: Deploy the system to production, then perform regular maintenance to address issues, implement updates, and ensure it continues to meet needs.

5.3 METHODOLOGIES IN SYSTEM ANALYSIS AND DESIGN

There are several methodologies used in SAD, each with its own strengths and weaknesses. Choosing the right methodology depends on the nature of the project, the complexity of the system, and the organization's preferences.

5.3.1 Waterfall Model

The Waterfall Model is a linear and sequential approach where each phase of the SDLC is completed before moving on to the next. It is easy to manage and works well for projects with clear and unchanging requirements.

5.3.2 Agile Methodology

Agile is an iterative and incremental approach that emphasizes flexibility and customer satisfaction. It allows for continuous feedback and adaptation, making it ideal for projects with dynamic requirements.

5.3.3 Spiral Model

The Spiral Model combines elements of both the Waterfall and Agile methodologies. It allows for iterative development with a focus on risk assessment and management. This model is particularly useful for large and complex projects.

5.4 TOOLS AND TECHNIQUES IN SYSTEM ANALYSIS AND DESIGN

Various tools and techniques are used in SAD to ensure accurate analysis and efficient design.

5.4.1 Data Flow Diagrams (DFD): Visually represent how data moves between processes and interacts with external entities.

5.4.2 Entity-Relationship Diagrams (ERD): Model data relationships to design database structures by defining entities, attributes, and relationships.

5.4.3 Unified Modeling Language (UML): A standardized language with diagrams like use case, class, and sequence diagrams to visualize and design system aspects.

CONCLUSION

System Analysis and Design is essential for developing effective information systems. A structured approach ensures systems meet business needs, enhance efficiency, and support decision-making. Key factors for success include understanding the SDLC phases, choosing suitable methodologies, and using appropriate tools and techniques.

REFERENCE

1. System Analysis and Design UML Version 2.0. Alan Dennis, Barbara Haley and David T. 3rd Edition
2.The Software Development Lifecycle. Everatt, G.D.; McLeod, R Jr (2007)
3. Software Development Lifecycle . Wikipedia.com

CHAPTER SIX

COMPUTER SECURITY

6.0 INTRODUCTION

Overview of Computer Security
Computer security, also known as cybersecurity, is the practice of protecting computer systems and networks from various threats, including unauthorized access, attacks, and damage. As technology evolves, so do the methods and tools used to ensure that information remains secure and systems function reliably.

Importance of Computer Security
In today's digital age, computer security is crucial for protecting sensitive information, maintaining privacy, and ensuring the integrity of systems. With increasing reliance on technology in personal, financial, and business contexts, effective security measures are essential for safeguarding against cyber threats.

6.1 FUNDAMENTALS OF COMPUTER SECURITY

- **6.1.1 Confidentiality**
  Confidentiality ensures that information is accessible only to those authorized to view it. Techniques to maintain confidentiality include encryption, access controls, and secure communication protocols.

- **6.1.2 Integrity**
  Integrity refers to the accuracy and completeness of information. Measures to ensure integrity include hashing, data validation, and integrity checking mechanisms to detect and prevent unauthorized modifications.

- **6.1.3 Availability**
  Availability ensures that information and resources are accessible to authorized users when needed. Strategies to maintain availability

include redundancy, backup systems, and denial-of-service (DoS) attack prevention.

- **6.1.4 Authentication and Authorization**
  Authentication is the process of verifying the identity of a user or system, while authorization determines the permissions granted to authenticated entities. Techniques include password-based authentication, multi-factor authentication (MFA), and role-based access control (RBAC).

6.2 THREATS AND VULNERABILITIES

6.2.1 Types of Threats

- **Malware:** Malicious software designed to damage or gain unauthorized access to systems. Types include viruses, worms, trojans, ransomware, and spyware.
- **Phishing:** Social engineering attacks aimed at tricking users into divulging sensitive information.
- **Denial-of-Service (DoS) Attacks:** Attacks designed to disrupt the availability of services by overwhelming them with traffic.
- **Insider Threats:** Risks posed by individuals within an organization who misuse their access for malicious purposes.

6.2.2 Vulnerabilities

- **Software Vulnerabilities:** Flaws or weaknesses in software that can be exploited by attackers. Examples include buffer overflows and injection flaws.
- **Configuration Weaknesses:** Insecure settings or misconfigurations that can expose systems to attacks.
- **Human Factors:** Errors or lapses in judgment by users that can compromise security, such as weak passwords or falling for phishing scams.

6.3 SECURITY MEASURES AND PRACTICES

6.3.1 Encryption

Encryption is the process of converting plaintext into ciphertext to protect data from unauthorized access. Common algorithms include AES (Advanced Encryption Standard) and RSA (Rivest-Shamir-Adleman).

6.3.2 Firewalls

Firewalls are network security systems that monitor and control incoming and outgoing traffic based on predetermined security rules. They can be hardware-based or software-based.

6.3.3 Intrusion Detection and Prevention Systems (IDPS)

IDPS are tools used to detect and respond to malicious activities. Intrusion Detection Systems (IDS) monitor network traffic for suspicious behavior, while Intrusion Prevention Systems (IPS) take action to block threats.

6.3.4 Antivirus and Anti-Malware Software

These programs are designed to detect, prevent, and remove malicious software from computers and networks. They use signature-based detection, heuristic analysis, and behavioral monitoring.

6.3.5 Secure Coding Practices

Secure coding involves writing software in a way that prevents vulnerabilities. Practices include input validation, error handling, and using secure libraries and frameworks.

6.4 IMPORTANCE OF COMPUTER SECURITY

In today's digital landscape, safeguarding sensitive information such as bank account numbers, credit card details, passwords, and work-related documents is crucial due to the heavy reliance on secure data pathways in everyday transactions and communications.

Data stored on computer systems is vulnerable to unauthorized access and misuse. Cybercriminals can infiltrate systems to alter program source

code, manipulate digital content, or compromise personal accounts, leading to the creation of harmful material such as inappropriate images or offensive social media posts.

The consequences of malicious activity extend far beyond individual breaches. Attackers can hijack compromised systems to launch further attacks on other computers, networks, or websites, causing widespread disruptions.

Ensuring the security of your system is not just a precaution but a necessity in the face of modern cyber threat

CONCLUSION

Computer security is a multifaceted field that encompasses a range of techniques, practices, and technologies to protect systems and data from threats. Understanding the fundamental principles, threats, and security measures is essential for maintaining a secure computing environment.

REFERENCES

NationalResearchCouncil.ComputersatRisk:SafeComputingintheInformation Age. Washington, DC: National Academy Press, 1991.

OrganizationforEconomicCo-operationandDevelopment.GuidelinesfortheSecurity of Information Systems. Paris, 1992.

Chapter Seven

SYSTEM DOCUMENTATION

7.0 INTRODUCTION

System documentation is the process of recording detailed information about the structure, design, operation, and use of a computer-based system. This documentation provides a written account of how the system works, its components, and how users and developers can interact with it. Proper documentation is essential for ensuring the long-term functionality, maintenance, and expansion of a system. It serves as a reference for system users, developers, administrators, and stakeholders.

7.1 TYPES OF SYSTEM DOCUMENTATION

System documentation can be categorized into different types, each serving a unique purpose. The most common types include:

- **User Documentation**: This type of documentation is intended for end-users and provides instructions on how to operate the system. It may include user manuals, installation guides, and help files.
- **Technical Documentation**: This documentation is primarily aimed at developers and IT professionals. It includes details about the system's architecture, code, and configurations. Examples include system design documents, API documentation, and configuration guides.
- **Operations Documentation**: These documents describe how to install, maintain, and operate the system on a day-to-day basis. It is often used by system administrators to ensure smooth operations.
- **Process Documentation**: This includes records of system development processes, including development methodologies, testing procedures, and version control. It ensures that future developers can understand the decisions made during system development.

7.2 PURPOSE OF SYSTEM DOCUMENTATION

The primary purpose of system documentation is to provide a detailed description of the system's design, functionality, and user interactions. Some of the key reasons why system documentation is important include:

- **System Maintenance**
- **Knowledge Transfer**
- **User Support**
- **Compliance**

7.3 COMPONENTS OF SYSTEM DOCUMENTATION

A well-structured system documentation typically contains the following components:

- **System Overview:** High-level description of the system's purpose, scope, and functionality.
- **System Architecture:** Details the system's structure, often with component and database diagrams.
- **User Interface Design:** Documents the layout and behavior of the user interface.
- **Data Flow Diagrams:** Visualize how data moves and is processed within the system.
- **Code Documentation:** Comments and explanations in the code for clarity on logic and functions.
- **Testing Documentation:** Outlines testing procedures, cases, and results to ensure quality.

7.4 IMPORTANCE OF COMPREHENSIVE DOCUMENTATION

Comprehensive system documentation offers several benefits, including:

- **Improved System Maintenance**
- **Enhanced Scalability**
- **Effective User Support**
- **Compliance with Standards**

7.5 BEST PRACTICES IN SYSTEM DOCUMENTATION

To ensure that system documentation is effective, organizations should follow best practices such as:

- **Clarity and Simplicity:** Use clear language, minimize jargon, and explain complex concepts.
- **Regular Updates:** Keep documentation current as systems evolve.
- **Consistency:** Maintain uniform terminology and formatting.
- **Visual Aids:** Use diagrams and screenshots to enhance understanding.
- **Collaboration:** Involve developers, admins, and users in creating documentation.

7.6 CHALLENGES IN SYSTEM DOCUMENTATION

Documenting complex systems can present several challenges, including:

- **Time Constraints:** In fast-paced environments, documentation is often deprioritized.
- **Knowledge Gaps:** Differing perspectives can lead to incomplete or inaccurate documentation.

- **Maintenance:** Regular updates are difficult in frequently changing systems.
- **User Engagement:** Poor organization or overly technical language can deter users

CONCLUSION

Documentation is the process of providing written details or info about something. It is a set of instructions, tutorials, manuals, reference tools provided by the system analyst to guide the use of the new system designed. Absence of these documents can throw the organization into problems with the use of the new system.

REFERENCE

Berger,David(2009)."ProceduresandDocumentation". http://www.maintenanceonline.org/maintenanceonline/content_images.

"ISO15787:2001Technicalproductdocumentation--Heat-treatedferrousparts -Rhyous: The Different Types of Technical Documentation for Software and WhyEach Is Important

CHAPTER EIGHT

HARDWARE MAINTENANCE

8.0 INTRODUCTION

Preventive maintenance in computing involves systematic inspections, cleaning, and replacing hardware components to ensure both hardware and software function optimally. By performing these routine checks, potential failures are mitigated, maintaining the reliability of the system.

Troubleshooting is the methodical approach used to pinpoint the cause of a malfunction within a computer system and resolve the related hardware or software issues.

8.1 FUNDAMENTAL HARDWARE ELEMENTS

Hardware comprises the physical components of a computer system, tangible and directly interactive. Key hardware elements include:

- **Input Devices**: Such as keyboards and mice.
- **Output Devices**: Including monitors, printers, and speakers.
- **Secondary Storage Devices**: Like hard drives, CDs, and DVDs.
- **Internal Components**: Such as the CPU, motherboard, and RAM.

Primary Hardware Components:

1. **Case/System Unit**: The enclosure that houses and protects the majority of the computer's internal components.
2. **Power Supply (SMPS)**: Converts AC power from the outlet to the DC power used by computer components, ensuring stable power supply with built-in cooling mechanisms.
3. **Processor & Fan**: The CPU, the central processing unit, executes instructions, while the fan prevents overheating.

4. **Motherboard**: The main circuit board that interconnects all hardware components and allows communication between them.
5. **RAM (Random Access Memory)**: Temporarily stores data for quick access, influencing the system's processing speed.

Storage Drives:

1. **Hard Drive**: A primary storage device for saving files and applications.
2. **CD/DVD Drive**: Used for reading and writing data on optical discs, with DVD drives supporting both CD and DVD formats.

Types of Hard Disks:

- **IDE (Integrated Drive Electronics)**: Utilizes a wide ribbon cable for parallel data transfer.
- **SATA (Serial Advanced Technology Attachment)**: Employs a narrower cable, allowing faster serial data transfer rates. SATA is preferred in modern systems for its efficiency.

Peripheral Hardware:

Peripheral hardware includes external components like mice, keyboards, monitors, printers, and speakers that interact with the computer system.

8.2 TROUBLESHOOTING

Troubleshooting requires a structured, methodical approach to diagnosing and resolving computer issues. Whether during preventive maintenance or when responding to user-reported problems, a logical process helps identify and resolve issues efficiently.

Data Backup: Before initiating any troubleshooting, it's essential to back up data to prevent potential loss. Data backups should be confirmed with the user, ensuring the most recent and complete data set is preserved. In cases where a backup is not available, users should sign a liability release before proceeding.

Troubleshooting Process:

1. **Problem Identification**: Collect comprehensive information from the system and the user. Use clear, jargon-free communication to obtain relevant details and document everything meticulously.
    - **Beep Codes**: During the POST (Power-On Self-Test), specific beep sequences can indicate hardware failures.
    - **BIOS Information**: Incorrect BIOS settings can cause startup failures; verifying settings against documentation is crucial.
    - **Event Viewer**: Logs system errors with detailed information, useful for identifying the cause of issues.
    - **Device Manager**: Identifies hardware issues, with icons indicating device status.
    - **Task Manager**: Monitors running applications and system performance, offering insight into potential problems.
    - **Diagnostic Tools**: Utilize manufacturer-provided tools for deeper hardware diagnostics.
2. **Formulate a Probable Cause Theory**: Develop theories based on the gathered information, starting with the most obvious causes. Prioritize and test these theories systematically.
3. **Test Theories to Determine the Cause**: Begin with the simplest tests and work towards more complex solutions. If all theories are exhausted without resolving the issue, develop new theories or escalate the problem to a more experienced technician.
4. **Action Plan and Implementation**: Once the cause is identified, devise a plan to resolve the issue. Research potential solutions, prioritize them, and implement them one by one, documenting each step.
5. **Confirm Full System Functionality**: After the solution is implemented, verify that the system operates correctly and take preventive measures to avoid future issues. Have the user confirm the resolution.
6. **Document Findings and Actions**: Record the entire troubleshooting process, including the problem description, steps taken, and components used. Clear documentation ensures continuity for future maintenance or troubleshooting.

CONCLUSION

Regular preventive maintenance reduces hardware and software problems. Before beginning any repair, back up the data on a computer. The troubleshooting process is a guideline to help you solve computer problems in an efficient manner. Document everything that you try, even if it fails. The documentation that you create is a useful resource for you and other technicians.

REFERENCES

Acosta,Jeremy."Basic Computer Maintenance". Games and Gears.

Lenk,J.D.Handbookofpracticalsolidstatetrouble-shooting.EnglewoodCliffs, N. J : Prentice-Hall, 1971.

Maguire,B.W.HandbookofComputermaintenanceandTrouble-shooting.Reston, Va: Reston publishing, 1973.

O'Leary,TimothyJandO`Leary, India.IComputerScienceEdwardArnold (publishers) Ltd, London.

CHAPTER NINE

COMPUTER NETWORKING

9.0 INTRODUCTION

A network is a system of interconnected computers that communicate and share data and resources, either through wired or wireless connections. Resources shared across networks include services, storage devices, and various applications. A network can range from a simple connection between two computers with a single cable to a complex system involving hundreds of computers linked through devices that manage data flow.

Networks can be connected using several types of media:

- **Copper cabling**: Transmits data through electrical signals.
- **Fiber-optic cabling**: Uses glass or plastic fibers to transmit information as light pulses.
- **Wireless connections**: Utilize radio signals, infrared technology, or satellite transmissions.

Computer networks support diverse applications and services, from the World Wide Web and digital media to shared servers and communication tools. They vary in transmission media, protocols, size, topology, and organizational structure, with the Internet being the most well-known network.

9.1 BENEFITS OF NETWORKING

Networking offers several key benefits:

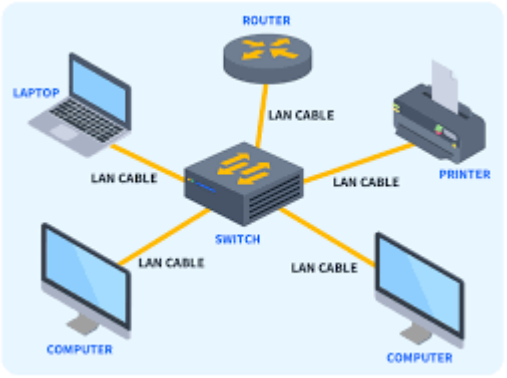
- **Fewer peripherals needed**
- **Enhanced communication**
- **Avoiding file duplication and corruption**

- **Reduced licensing costs**
- **Centralized administration**

9.2 TYPES OF NETWORKS

Computer networks can be classified based on several characteristics, including the area they serve, data storage methods, resource management, organizational structure, and the type of networking devices and media used. The main types of networks are:

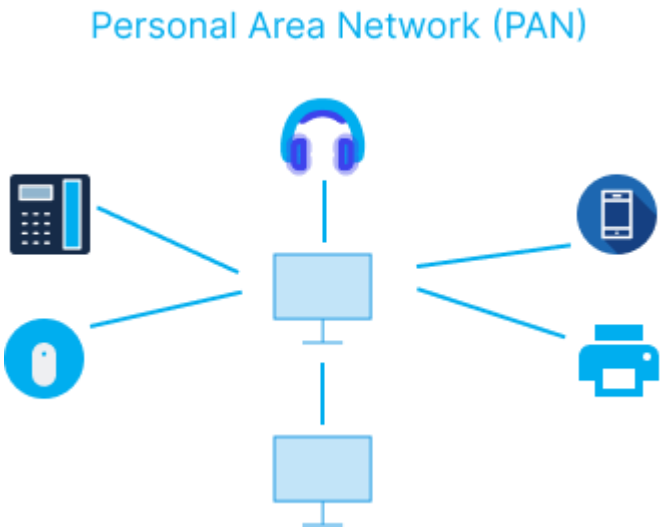
- **LAN (Local Area Network)**: A network within a single geographical area, providing services and applications to users in a common organizational structure, typically connected by copper cabling.



- **WLAN (Wireless Local Area Network)**: A LAN that uses radio waves to transmit data between wireless devices, with access points connected to the network via copper cabling.

- **PAN (Personal Area Network)**: Connects devices like mice, keyboards, printers, smartphones, and tablets within an individual's

personal space, often using Bluetooth technology.



- **MAN (Metropolitan Area Network):** Connects buildings in a city or campus using wireless or fiber backbones.
- **WAN (Wide Area Network):** Links multiple LANs across distant locations; the Internet is the largest WAN.
- **Peer-to-peer networks:** No central server; each device manages its own resources.
- **Client/server networks:** Clients request services from a central server, allowing centralized control.

9.3 NETWORK DEVICES

- **Modem**: A device that connects to the Internet via an ISP, converting digital data to analog signals and vice versa for communication between computers.
- **Hub**: A device that extends the range of a network by receiving data on one port and sending it to all other ports. Hubs can also connect to other networking devices like switches or routers.
- **Switch**: A networking device that interconnects various devices within a network. It transmits data packets only to the intended recipient, optimizing network traffic and enhancing security and efficiency.

9.4 INTERNET SERVICE PROVIDER (ISP)

An ISP is a company that offers Internet services, such as connection, email accounts, and web pages, to individuals and businesses for a fee. Key equipment used to connect to an ISP includes:

- **Wireless Access Points (WAPs)**: Devices that enable Wi-Fi-enabled devices to connect to a wired network. WAPs can be standalone or integrated into a wireless router.
- **Routers**: Devices that connect multiple networks, using IP addresses to forward packets to other networks. Routers may be standalone devices or specialized computers.
- **Network-Attached Storage (NAS)**: A device with one or more hard drives, an Ethernet connection, and an embedded operating system, used for file sharing, media streaming, and data backup within a network.

9.5 NETWORK SECURITY

Network security involves the policies and measures adopted by network administrators to prevent unauthorized access, misuse, or denial of services on a network.

9.6 NETWORK SURVEILLANCE

Network surveillance involves the monitoring of data transmitted over networks, including the Internet, often covertly. This can be done by governments, corporations, or individuals, and may require authorization depending on the jurisdiction.

9.7 END-TO-END ENCRYPTION

End-to-end encryption (E2EE) secures data from sender to recipient, preventing intermediary access or tampering. It's used in protocols

like HTTPS, PGP, and OTR, but doesn't protect against endpoint risks or traffic analysis.

9.8 SSL/TLS

The rise of e-commerce in the mid-1990s led to the development of Secure Socket Layer (SSL) by Netscape. SSL requires servers to have certificates, which are validated by clients before establishing a secure connection. This protocol, later succeeded by Transport Layer Security (TLS), ensures secure communication over the web.

CONCLUSION

Computer systems and computerized systems play a vital role in enhancing human efficiency and pushing the boundaries of what is possible. When these devices are interconnected to create a network, their capabilities are significantly multiplied.

REFERENCES

BARAN, P.:“On Distributed Communications: I. Introduction to Distributed Communication Networks,” Memorandum RM-420-PR, Rand Corporation, Aug. 1964.

BELSNES,D.:“FlowControlinthePacketSwitchingNetworks,”Communicatio ns Networks, Uxbridge, England: Online, pp. 349–361, 1975.

CHAPTER 10

SIWES REPORT AND MINI PROJECT

10.0 INTRODUCTION TO SIWES

The Student Industrial Work Experience Scheme (SIWES) is a skills training program designed for Nigerian students in tertiary institutions pursuing courses in science, engineering, technology, and other related fields. It is a program introduced by the Industrial Training Fund (ITF) in 1973 to bridge the gap between theoretical learning in classrooms and the practical applications in industries. The scheme provides students with the opportunity to gain real-world experience and hands-on training in relevant sectors, preparing them for the challenges of the workforce upon graduation.

10.1 IMPORTANCE OF SIWES

SIWES plays a crucial role in enhancing the employability of students by equipping them with practical knowledge and skills that complement their academic education. It helps to:

1. Provide students with the opportunity to apply their theoretical knowledge in a real industrial environment.
2. Improve the students' technical expertise and problem-solving capabilities through hands-on experiences.
3. Foster collaboration between academic institutions and industries, ensuring that students’ training meets industry standards.
4. Expose students to the realities of professional life, including work ethics, time management, and teamwork.

10.2 OBJECTIVES OF SIWES

The primary objectives of SIWES are as follows:

1. To prepare students for the labor market by providing them with practical experience in their field of study.
2. To develop a better understanding of industrial processes, tools, and techniques.
3. To enhance the technical skills and competencies required for effective job performance.
4. To bridge the gap between theoretical classroom learning and the demands of the workplace.
5. To foster a culture of professionalism and continuous learning among students, ensuring they are industry-ready upon graduation.

10.3 STRATEGIC HUB LIMITED: PIONEERING GROWTH AND INNOVATION

Strategic Hub Limited is a Nigerian consulting firm focused on driving business growth and digital transformation through innovative strategies and technology-driven services. Specializing in tailored solutions, they help brands achieve sustainable growth by enhancing operational efficiency, expanding market reach, and driving customer engagement.

Their goal is to transform struggling businesses into thriving enterprises in both local and global markets. Strategic Hub also offers tech skill programs, led by expert tutors, to equip individuals with in-demand skills for the digital age, some of which are:

- **Fullstack Web Development:** Strategic Hub offers training in both front-end and back-end technologies, teaching participants to build scalable websites and applications.
- **Data Analytics:** The firm provides instruction on data collection, processing, and analysis to help learners drive business insights and informed decision-making.

- **UI/UX Design:** Strategic Hub equips students with skills to design user-friendly, visually appealing interfaces that enhance the user experience.
- **Digital Marketing:** The company teaches strategies for optimizing online presence, increasing engagement, and driving revenue through targeted digital marketing.
- **Web Design:** Strategic Hub trains learners to create visually compelling, functional websites that stand out online.

Frontend Technologies:

- HTML
- CSS
- JavaScript
- Bootstrap

Backend Technologies:

- Python
- Flask
- PostgreSQL (for database management)
- Git
- Postman (for testing and managing APIs)

Deployment Technologies:

- Docker

10.4 PROJECTS I WORKED ON

During my Industrial Training, I worked on key projects that enhanced my development skills:

1. **Signup/Login Authentication System**: I built a secure backend using Python and Flask, featuring password hashing, session management, and JWT-based authentication.
2. **Group Chat Application**: I developed a basic group chat app with real-time communication using Flask and socket programming.

3. **Team Blog Site**: I handled the backend for a blog site, managing user authentication, post creation, and comments, using Flask and PostgreSQL, while collaborating with frontend developers.

   Each of these projects provided me with invaluable experience, improving my ability to work on both individual and team-driven tasks while honing my backend development skills.

10.5 CHALLENGES ENCOUNTERED DURING MY I.T. PROGRAMME

During my Industrial Training Programme, I faced several challenges:

1. **Struggling with Technical Concepts:** Initially, I had difficulty grasping complex backend development topics, which slowed my progress. However, with consistent effort and colleague support, I overcame this.
2. **Poor Socialization Skills:** My introverted nature made it hard to communicate and collaborate effectively at first, but over time, my social and teamwork skills improved, allowing me to contribute better to projects.

These challenges helped me build resilience, deepen my technical knowledge, and improve as a team player.

10.6 MINI-PROJECT (A RENTAL MANAGEMENT SYSTEM)

Using the skills I learnt during my Industrial training, I undertook the project of building a rental management system web application called Rently.

10.6.1 My Motivation for This Project

During my first year at the University of Uyo, I missed out on hostel accommodation due to a late application and struggled to find a nearby apartment. This experience made me wish for a better way to search for available rentals. During my Industrial Training, I realized an app like this would have been invaluable, motivating me to build a platform that helps users find rental properties in any location.

10.6.2 Project Features

- Signup/Login Authentication
- Property Uploads
- Property Search
- Booking Functionality
- Maintenance Requests for Existing Tenants
- Tenant Applications

10.6.3 Project Tech Stack

- Frontend: HTML/CSS/JavaScript
- API: FastAPI for building the API
- Backend: Flask to consume the API
- Database: PostgreSQL
- ORM: SQLAlchemy
- Packaging: Docker
- Hosting: Render

10.6.4 Challenges Faced While Building the Project

Building the API from scratch using FastAPI was challenging as it was a new library for me, but guidance from supervisors and course materials helped me overcome this.

Working with ForeignKeys and Relationships in PostgreSQL required a deep understanding of database schema interactions, which I developed over time.

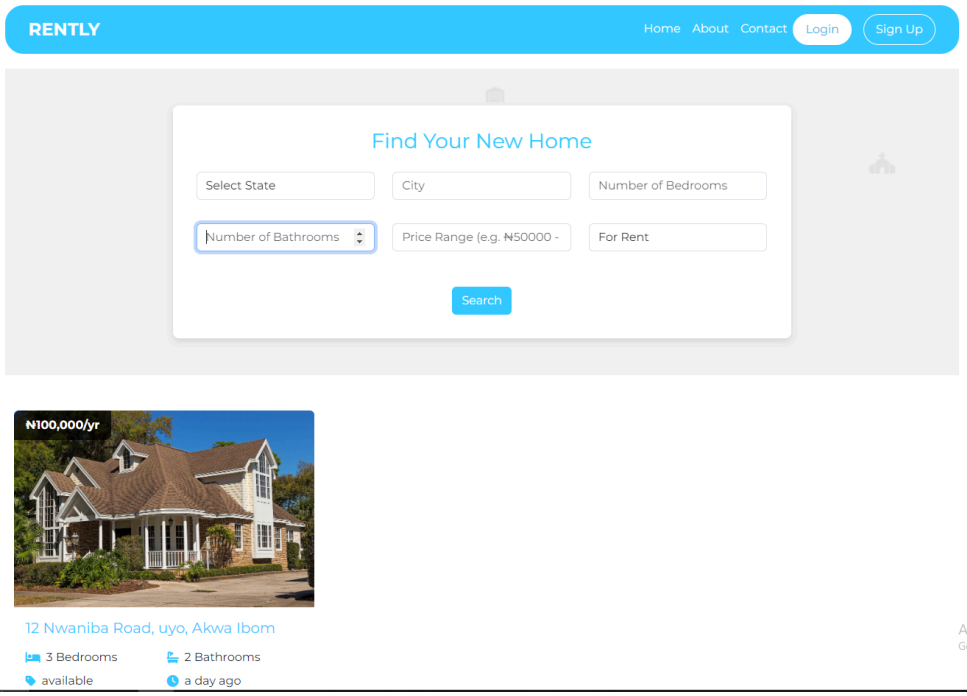
Frequent changes in project requirements led to refactoring, but setting a fixed requirement set helped maintain focus and avoid unnecessary changes.

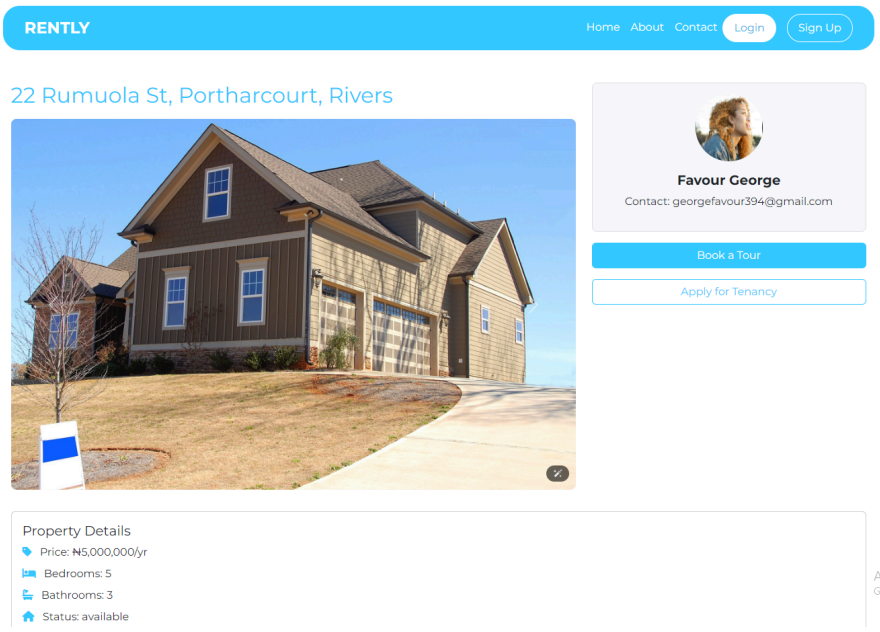
10.6.5 Future Integrations for the Project

In the future, I plan to add a rent payment functionality to allow users to make payments directly through the platform.

I noticed the Flask server's response time is slow due to large payloads. To fix this, I plan to integrate GraphQL endpoints to fetch only the necessary data, improving performance.

PROJECT SNAPSHOTS





RECOMMENDATION

I strongly advocate for the continued inclusion of the SIWES program in the curriculum of tertiary institutions, both at the Federal and State levels. This program is crucial for equipping students with essential life and work skills, enhancing their technical expertise not only in their specific fields of study but also in the wider societal context.

Overall, SIWES has consistently demonstrated its significant value and support. I am confident that it will remain a vital resource for both myself and my peers as we undertake our final projects and confront future challenges. The program has laid a solid foundation for achieving greater success in our careers and personal development.

CONCLUSION

The Student Industrial Work Experience Scheme (SIWES) provided me with a unique opportunity to take on real-world work responsibilities within a company. I can confidently assert that SIWES delivers invaluable technical, professional, and vocational training to university students, effectively preparing them for careers across a wide range of industries.

The knowledge I gained through this program extends beyond academic and technical skills. It has also illuminated the importance of various fields of study and their critical roles in the success of different industries. This experience has deepened my understanding and appreciation of my chosen field, both within the academic setting and in the broader societal context.