

UNIVERSITY OF ILORIN

Centre for Open and Distance Learning

CSC 319

Internet Technology II



Prepared By: Dare Muyiwa Mobolaji

Study Guide

Course Title: INTERNET TECHNOLOGY II

Credit: 2 units

Introduction

Hi there! Let's talk about something that has become such an integral part of our daily lives that we often take it for granted: the Internet. At its core, the Internet is a massive network of computers connected globally through a system of routers and servers. When you send an email or visit a website, your device communicates with servers using Internet protocols like TCP/IP. Data is broken down into small packets, sent across various networks, and reassembled at the destination. This process happens in mere seconds, allowing for instant communication and information retrieval. It's amazing to think about how this global network has transformed the way we communicate, learn, work, and entertain ourselves. From streaming movies to connecting with friends across the globe via web applications such as Facebook, Twitter, Instagram and etc.

Internet technology II is a second semester course. It is a 2-credit course that is available to students offering Bachelor of Science, (B.Sc.), in Computer Science, Information Systems and Allied degrees. This course gives students advance knowledge about internet. The way it has evolved, incorporated different technologies, techniques and applications under a single umbrella, making it the most popularly used infrastructure in terms of information exchange and communication in today's world. It will explore various concepts in the Internet such as Network Models and Protocols, Intranet, Extranet, and Internet Security. In addendum, this course is designed to equip students with the capability to design and program websites. For the purpose of practical understanding of the web design and development module, Visual Code IDE will be used throughout the module.

Course Goal

The goal of this course is to provide you with a comprehensive understanding of the foundational concepts and practical applications of Internet technologies. The course aims to equip you with the knowledge of how data is transmitted over networks, the functioning of various network models such as the OSI and TCP/IP models, and the operation of essential internet protocols like HTTP, FTP, and SMTP. The course is designed to develop your ability to comprehend the operation of the World Wide Web, understand the challenges associated with Internet security, and stay abreast

of emerging trends in the field. By the end of the course, you are expected to have the skills necessary to apply Internet technologies effectively in real-world scenarios.

Course Contents

Module 1: Introduction to the Internet.....	5
Unit 1: History of the Internet	5
Unit 2: Impact and challenges of the internet	10
Unit 3: Review of Network Technologies.....	13
Module 2: Network Models and Protocols	19
Unit 1: Network Models.....	19
Unit 2: Network Protocols.....	28
Unit 3: Internet Protocol (IP) Addressing, and Subnetting	32
Unit 4: IP Subnetting.....	39
Module 3: Web Design and Web Development	45
Unit 1: World Wide Web	45
Unit 2: HTML and CSS	53
Unit 3: Client-Side Programming	77
Unit 4: Server-Side Programming.....	104
Module 4: Intranet and Extranet	143
Unit 1: Intranet	143
Unit 2: Extranet	147
Unit3: Network Security	156

Course Information

This is a compulsory course for students in the Department of Computer Science, Information Systems and Allied degrees. You are expected to participate in all the course activities and have a minimum of 75% participation to be qualified for the final examination.

Related Course:

Pre-requisite: CSC 211, CSC 212, CSC 224

Required For: CSC 319

Requirement for Success

To be successful in this course, students need to attend up to 75% of lecture hours.

Grading and Assessment

Continuous assessment takes 40% and examination 60%.

Module 1: Introduction to the Internet

Unit 1: History of the Internet

Unit 2: Review of Network Technologies

Unit 1: History of the Internet

1.0 Introduction

Have you ever considered what the world could be like without the internet? In a world without the internet, communication, education, healthcare, and the economy would undergo significant changes. Firstly, communication would be profoundly affected. People would rely on traditional methods such as postal mail and landline telephones. This would slow down the exchange of information, making it more challenging to maintain personal and professional relationships over long distances.

In the field of education, the absence of the Internet would limit access to information and learning resources. Online courses, educational platforms, and digital libraries provide opportunities for remote learning and self-education. Without these tools, learning would be confined to physical classrooms and libraries, potentially exacerbating educational inequalities, especially in remote or underprivileged areas. Healthcare systems would also face challenges. The internet facilitates telemedicine, allowing patients to consult with healthcare professionals remotely. It also enables the rapid sharing of medical research and collaboration among professionals worldwide. Without internet connectivity, accessing medical information and services would be more difficult, potentially impacting patient care and slowing medical advancements.

The economy would experience notable disruptions. E-commerce platforms allow businesses to reach customers globally, and online banking facilitates financial transactions. Without the Internet, businesses would operate on a more local scale, and financial services would revert to traditional methods. This could lead to reduced economic growth and limited opportunities for businesses and consumers alike. Moreover, access to real-time information would be limited. The internet provides immediate updates on news, weather, and emergency alerts. Without it, disseminating information would take longer, possibly affecting public safety and awareness during critical situations. A world without the Internet might encourage more face-to-face

interactions and community engagement. People might spend more time engaging in outdoor activities and personal hobbies. The lack of instant access to information and global connectivity could hinder progress and development in various sectors. In summary, while life without the Internet might have some benefits in promoting direct human interactions, the overall impact would be challenging. In this unit, you will learn about the history of the Internet, the structure and functioning of the Internet, and the impact of the Internet on communication, information access, economics and business.

1.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Give a brief history of the Internet.
- ii. Define internet.
- iii. Itemize 5 key features of the internet.
- iv. Describe at least 5 functions of the internet.

3.0. History of the Internet

The Internet has become an integral part of modern society, transforming the way people communicate, access information, and conduct business. Originating from military and academic research in the mid-20th century, the Internet has evolved into a global network connecting billions of devices and users worldwide. This article provides an introduction to the Internet, exploring its history, structure, impact, and future prospects. The inception of the Internet can be traced back to the 1960s during the Cold War era. The United States Department of Defense initiated a project called the Advanced Research Projects Agency Network (ARPANET) in 1969. ARPANET was designed to enable secure and reliable communication between government and academic institutions in the face of potential nuclear threats.

In the early 1970s, scientists Vinton Cerf and Robert E. Kahn developed the Transmission Control Protocol/Internet Protocol (TCP/IP), which became the foundational communication protocol for the Internet. TCP/IP allowed diverse computer networks to interconnect seamlessly, facilitating data exchange and communication over vast distances. The introduction of the World Wide Web (WWW) in 1989 by Sir Tim Berners-Lee marked a significant milestone. Working at the European

Organization for Nuclear Research (CERN), Berners-Lee proposed a system of interlinked hypertext documents accessible via the Internet. This innovation enabled users to navigate the Internet through web browsers using hyperlinks, making information retrieval more efficient and user-friendly.

3.1 Internet

The Internet is a vast global network that connects millions of private, public, academic, business, and government networks. It facilitates the transmission of data and communication between devices around the world using standardized protocols. The internet enables users to access an extensive array of information and services, including websites, email, social media, file sharing, and streaming content.

Key features of the Internet include:

1. **Connectivity:** It allows diverse devices, such as computers, smartphones, and IoT (Internet of Things) devices, to communicate with each other regardless of their location.
2. **Interoperability:** The use of standard protocols, such as TCP/IP (Transmission Control Protocol/Internet Protocol), ensures that data can be shared and understood across different systems.
3. **Accessibility:** The Internet is available to a large portion of the world's population, providing access to information and resources that were previously difficult to disseminate.
4. **Decentralization:** There is no single governing body for the Internet, which allows for a wide variety of services and content to be hosted by various independent entities.
5. **Dynamic Content:** It hosts a continuously evolving collection of information, including websites, social media posts, videos, and applications, often updated in real time. Overall, the internet has transformed how people communicate, conduct business, learn, and connect with one another, playing a crucial role in modern society.

3.2 Structure and Functioning of the Internet

The Internet is a vast network of interconnected computer networks that communicate using standardized protocols. Its structure is decentralized and relies on a hierarchical framework:

- i. **Backbone Networks:** These are high-capacity data routes that form the core of the Internet, connecting major data centres and network hubs globally. Backbone networks are typically owned by large telecommunications companies and use fibre-optic cables for high-speed data transmission.
- ii. **Internet Service Providers (ISPs):** ISPs deliver Internet access to end-users, including individuals, businesses, and organizations. They connect subscribers to the Internet backbone through various mediums such as Digital Subscriber Lines (DSL), cable, fibre-optics, or wireless connections.
- iii. **Routers and Switches:** Routers direct data packets between networks, ensuring that information reaches its intended destination. Switches connect devices within a single network, facilitating communication and resource sharing.
- iv. **Domain Name System (DNS):** The DNS translates human-readable domain names (e.g., www.example.com) into numerical IP addresses required for locating and identifying devices on the network.
- v. **Protocols:** The Internet relies on a suite of protocols for data exchange. TCP/IP is the primary protocol, managing how data is packetized, addressed, transmitted, routed, and received.

4.0 Self-Assessment Questions

1. WWW was introduced by

(a) Sir Tim Berners-Lee (B) Vinton Cerf (C) Robert E. Kahn

2. ARPANET was introduced by

(a) The United States Department of Defense (B) Soviet Union (C) Government

3. Which of the following options is an impact of the internet on the global economy?

(a) E-commerce (b) Society (C) Spamming

4. Which of the options is an impact of the internet on communication and information access

(a) social media (b) Protocols (c) DNS

5. Which of the following best represents structure and functioning of internet?

(a) Backbone Networks, Internet Service Providers, Routers and Switches, Domain Name System, and Protocols.

(b) Backbone Networks, Internet Service Providers, RJ 45 connector, Domain Name System, and Protocols.

(c) Backbone Networks, Internet Service Providers, Repeater, Domain Name System, and Protocols.

6. Which of these is true about Backbone networks?

(a) high-capacity data routes that form the core of the Internet.

(b) collection of information

(c) Channels

7. The following statements are true about ISPs except

(a) ISPs deliver Internet access to end-users.

(b) ISPs are WIFI, router and modem.

(c) They connect subscribers to the Internet backbone through various mediums such as Digital Subscriber Line (DSL), cable, fibre-optics, or wireless connections

8. Routers direct data packets between networks, while Switches connect devices within a single network, facilitating communication and resource sharing (True / False).

9. The DNS translates human-readable domain names (e.g., www.example.com) into numerical IP addresses. (True/ False)

10. The Internet relies on a suite of protocols for data exchange. (True/ False)

5.0 Tutor Marked Assignments

1. Discuss the history of the internet

2. What do you understand by the following terms

(i) Backbone Networks (ii) Internet Service Providers (iii) Routers and Switches (iv) Domain Name System (DNS) (v) Protocols

Unit 2: Impact and challenges of the internet

1.0 Introduction

Has the internet ever impacted your life in any way you can imagine? It's incredible when you consider its profound impact. From the way we communicate to how we work, learn, and entertain ourselves; the internet has become an integral part of our daily routines. But alongside these amazing benefits come some significant challenges. Let's explore the impact the internet has had on our world and delve into some of the hurdles we face because of it. One of the most remarkable impacts of the internet is how it has revolutionized communication. Remember the days when making a long-distance call was a big deal? Now, we can video chat with someone halfway across the world in real time without a second thought. Social media platforms have connected us in ways we never imagined. We can stay in touch with friends and family, share life updates, and even meet new people who share our interests. This unit presents the impact and challenges of the internet.

2.0 Learning Outcomes

At the end of this unit, you should be able to:

- i. State 5 impacts of the internet on communication and information Access.
- ii. State 3 impacts of Internet on Economy and Business Transformation.
- iii. Itemize 4 challenges of the internet.

3.0 Impact of Internet on Communication and Information Access

The Internet has revolutionized communication, breaking down geographical barriers and enabling instantaneous interaction:

- i. **Email:** One of the earliest Internet applications, email allows users to send and receive messages electronically across the globe.
- ii. **Social media:** Platforms like Facebook, Twitter, and Instagram facilitate social networking, content sharing, and community building.
- iii. **Instant Messaging and VoIP:** Services such as WhatsApp, Skype, and Zoom provide real-time text messaging, voice, and video communication. In terms of information access, the Internet serves as an immense repository of data:
- iv. **Search Engines:** Tools like Google and Bing index vast amounts of web content, enabling users to retrieve information on virtually any topic.
- v. **Online Education:** E-learning platforms and Massive Open Online Courses (MOOCs) like Coursera and edX offer educational resources and courses accessible to a global audience.
- vi. **Digital Libraries and Databases:** Institutions provide access to academic journals, ebooks, and multimedia resources, supporting research and knowledge dissemination.

3.1 Economic and Business Transformations

The Internet has significantly impacted the global economy:

- i. **E-Commerce:** Online marketplaces such as Amazon and Alibaba facilitate buying and selling goods and services over the Internet.
- ii. **Digital Payment Systems:** Services like PayPal and mobile payment apps enable secure online transactions, supporting the growth of the digital economy.
- iii. **Remote Work:** The Internet enables telecommuting and remote collaboration, allowing businesses to operate with distributed teams.
- iv. **Cloud Computing:** Services provided by companies like Amazon Web Services (AWS) and Microsoft Azure offer scalable computing resources over the Internet, reducing the need for physical infrastructure.

3.2 Challenges

Despite its benefits, the Internet presents several challenges:

- i. **Cybersecurity Threats:** Malicious activities such as hacking, phishing, malware distribution, and denial-of-service attacks pose risks to individuals and organizations.

- ii. **Privacy Issues:** The collection and analysis of personal data by corporations and governments raise concerns about surveillance and data misuse.
- iii. **Digital Divide:** Disparities in Internet access due to socioeconomic factors lead to unequal opportunities in education, employment, and information availability.
- iv. **Misinformation:** The rapid spread of false or misleading information online can have serious social and political consequences.

The Internet has profoundly transformed various aspects of human life, from communication and commerce to education and entertainment. Its decentralized structure and standardized protocols have enabled unprecedented connectivity and information exchange. However, the Internet also presents challenges that require ongoing attention, including cybersecurity, privacy, and equitable access. As technology advances, the Internet is poised to further influence society, driving innovation and shaping the future of global interactions.

4.0 Self-Assessment Questions

1. Which of the following lists best describes the impact of the internet on communication and information access?

- (a) Email, Search Engines, Social media, Instant Messaging, and VoIP.
- (b) Telegram, post office, social media, and Email.
- (c) Instant Messaging, and VoIP, Simplex, and Half-Duplex

2. Which of the following lists best describes the impact of the Internet on economic and business Transformations?

- (a) E-Commerce, Digital Payment Systems, Remote Work
- (b) Cloud computing, review, Remote Work
- (d) Cloud computing, sales, investment.

3. Which of the following best describes the challenges of the internet?

- (a) Virus, Antivirus, spyware, Antispyware
- (b) Cybersecurity Threats, Privacy Issues, Digital Divide, Misinformation.

(c) Virus, Trojan horse, worm

4. Which of the options is an impact of the internet on communication and information access

(a) social media (b) Protocols (c) DNS

5.0 Tutor Marked Assignments

1. Discuss the impact of the internet on communication and information access.

2. Briefly describe the functioning of the internet using 5 components of internet structure.

Unit 3: Review of Network Technologies

1.0 Introduction

Have you ever thought about how we've gone from simple wired connections to the complex wireless networks we have today? Let's take a journey through the evolution of network technologies and see how they've transformed the way we communicate. It all started in the 19th century with the telegraph. Invented by Samuel Morse in the 1830s, the telegraph allowed people to send coded messages over wires across long distances. It was revolutionary at the time because it drastically reduced communication times. Then came the telephone. In 1876, Alexander Graham Bell patented the first practical telephone. This invention allowed real-time voice communication between people in different locations. It was a big leap forward from the telegraph because it conveyed more information faster and more naturally.

In the mid-20th century, computers started becoming more common. But initially, they were isolated machines. The idea of connecting computers led to the development of networks. In the late 1960s, the U.S. Department of Defense funded a project called ARPANET. It was the world's first operational packet-switching network and the predecessor of the Internet. ARPANET used a technology called packet switching, which broke data into small packets for transmission. This method was more efficient and robust than traditional circuit-switching methods used in telephones. This led to the rise of the Internet in the 1980s, more networks started to appear, but they weren't all connected creating a need for the development of internet technologies. The unit presents previous network technologies, the latest network technologies, their challenges and future prospects of network technologies.

2.0 Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss the evolution of network technologies.
- ii. Explain at least 4 latest network technologies
- iii. Discuss the challenges of the latest network technologies
- iv. Discuss the prospects of network technologies.

3.0. Network Technologies

Network technology has undergone significant transformations since its inception, playing a pivotal role in the advancement of global communication and the proliferation of information technology. This review provides an analysis of the evolution of network technology, its current state, and future prospects. The origins of network technology can be traced back to the development of the telegraph and telephone systems in the 19th century, which laid the groundwork for wired communication networks. The advent of the Internet in the late 20th century revolutionized network technology, introducing packet-switched networks that allowed for efficient data transmission over distributed systems.

In the 1980s and 1990s, the development of Ethernet technology and the adoption of the Transmission Control Protocol/Internet Protocol (TCP/IP) suite became foundational for local area networks (LANs) and the broader Internet, respectively. The implementation of wireless networking standards, such as Wi-Fi based on the IEEE 802.11 standards, further expanded networking possibilities by enabling wireless local area networks (WLANs). Today, network technology encompasses a wide range of wired and wireless technologies that facilitate global connectivity. Key components of current network technology include broadband networks, wireless networks, network virtualization and the internet of Things.

- i. **Broadband Networks:** High-speed Internet access is predominantly provided through broadband technologies such as Digital Subscriber Lines (DSL), cable modems, and fibre-optic communication.
- ii. **Wireless Networks:** The proliferation of mobile devices has led to the widespread use of cellular networks, including 4G and the emerging 5G networks, providing high-speed wireless communication.

- iii. **Network Virtualization:** Software-defined networking (SDN) and network functions virtualization (NFV) have introduced flexibility in network management by decoupling the control plane from the data plane and virtualizing network services.
- iv. **Internet of Things (IoT):** The integration of sensors and devices into networks has facilitated the growth of IoT, where everyday objects are connected to the Internet, enabling data collection and automation.

3.1. Challenges in Network Technology and Future Prospects

Despite advancements, network technology faces several challenges:

- i. **Security Concerns:** With the increasing complexity of networks, cybersecurity threats have become more sophisticated, necessitating advanced security measures.
- ii. **Scalability Issues:** The exponential growth in data traffic demands networks that can scale efficiently without degradation in performance.
- iii. **Interoperability:** The integration of diverse network technologies requires standards and protocols that ensure seamless communication between different systems.

3.2 Prospects of Network Technology

The future of network technology is poised to be shaped by several emerging trends:

- i. **5G and Beyond:** The deployment of 5G networks promises significant enhancements in speed, latency, and connectivity, enabling new applications such as autonomous vehicles and advanced IoT solutions.
- ii. **Edge Computing:** To address latency and bandwidth issues, edge computing brings computation and data storage closer to data sources, reducing the need for data to traverse the entire network.
- iii. **Artificial Intelligence (AI) in Networks:** AI and machine learning are increasingly being integrated into network management to optimize performance, predict failures, and enhance security protocols.
- iv. **Quantum Networking:** Research into quantum communication networks aims to leverage quantum mechanics principles for ultra-secure data transmission and enhanced computational capabilities.
- v. **Web 3:** The evolution of the internet has been marked by significant transformations, from

the static information delivery of Web 1.0 to the interactive, user-generated content of Web 2.0. As technological advancements continue, a new iteration known as Web 3.0, or simply Web 3, is emerging. This new phase aims to redefine the Internet's architecture by introducing decentralization, enhanced security, and user empowerment through blockchain technology.

Key Characteristics of Web 3

1. Decentralization: Unlike the centralized servers of the current internet, Web 3 operates on decentralized networks. This means data is stored across multiple nodes globally, reducing the control of large corporations over user data and mitigating the risk of single points of failure.

2. Blockchain Integration: Blockchain technology is fundamental to Web 3's infrastructure. It provides a distributed ledger that records transactions transparently and immutably. This ensures data integrity and fosters trust among users without the need for intermediaries.

3. Smart Contracts: These are self-executing contracts with the terms of the agreement directly written into code. Smart contracts facilitate transactions and agreements without the need for a central authority, reducing costs and increasing transaction speed and efficiency.

4. Enhanced User Control: Web 3 empowers users by giving them control over their data and digital identities. Through cryptographic keys, individuals can manage access to their personal information, enhancing privacy and security.

5. Interoperability: Web 3 promotes seamless interaction between different platforms and services. Protocols are designed to be compatible, allowing for the transfer and management of assets across various applications.

Vi. Blockchain: Blockchain technology has emerged as a transformative force with the potential to revolutionize network technology. As a decentralized ledger system, blockchain offers enhanced security, transparency, and efficiency. This article explores how blockchain is positioned to become the cornerstone of future network infrastructures. Blockchain is a distributed ledger technology that records transactions across multiple computers. This ensures that the record cannot be altered retroactively without the alteration of all subsequent blocks and the consensus of the

network. Blockchain operates on a peer-to-peer network, removing the need for a central authority and enabling participants to confirm transactions independently.

Blockchain's decentralized nature eliminates the reliance on a central server, reducing vulnerabilities associated with centralized systems. Each participant in the network holds a copy of the ledger, ensuring data redundancy and resilience against single points of failure. This structure enhances the robustness of network technology by distributing control among participants.

Utilizing cryptographic algorithms, blockchain ensures that once data is recorded, it becomes immutable. Transactions are secured through cryptographic hashes linking each block to the previous one. This chain of blocks makes unauthorized data modification highly improbable, significantly enhancing network security. Blockchain provides transparency by allowing all authorized participants access to the ledger. Every transaction is visible to network members, fostering trust and accountability. This level of transparency is particularly beneficial in industries requiring verifiable records and audit trails.

By streamlining processes and reducing the need for intermediaries, blockchain increases operational efficiency. Smart contracts—self-executing contracts with the agreement encoded into code—automate transactions, reducing delays and costs associated with traditional contract enforcement. Blockchain can secure IoT networks by providing a decentralized framework for device communication. It ensures that data exchange between devices is secure, verifiable, and resistant to tampering. This application addresses critical security concerns in IoT ecosystems. In supply chains, blockchain enhances traceability and transparency. It records every transaction from origin to delivery, enabling stakeholders to verify the authenticity and condition of goods. This reduces fraud and improves efficiency in logistics and inventory management.

Blockchain is transforming financial services by enabling faster, more secure transactions. It facilitates real-time cross-border payments and reduces transaction costs. Blockchain's transparency and security features also enhance compliance and reduce the risk of fraud. Despite its advantages, blockchain faces scalability challenges. As the number of transactions increases, the network can experience latency issues. Additionally, regulatory frameworks for blockchain are

still evolving, posing legal and compliance uncertainties. Addressing these challenges is vital for the widespread adoption of blockchain in network technology.

Blockchain technology holds significant promise as the future of network technology. Its decentralized, secure, and transparent nature addresses many limitations of current network systems. While challenges remain, ongoing advancements and regulatory developments are paving the way for blockchain to become integral to network infrastructures globally.

4.0. Self-Assessment Questions

1. Which of these network technologies is useful when dealing with wireless devices?

(a) LAN (b) Fiber optics (c) Wireless Technology

2. The major challenges in network technologies include.

(a) Security concern (b) Scalability (c) 5G

3. Which of the following lists best describes the prospect of network technology?

(a) 5G, Edge Computing, Artificial Intelligence (AI) in Networks, Web 3, Blockchain

(b) Quantum Networking, Web 3, Blockchain.

(c) Quantum Networking, Web 3, Blockchain

4. Discuss the evolution of network technologies.

5. Briefly discuss at least 4 latest network technologies

6. Discuss the challenges of the latest network technologies

5.0. Tutor Marked Assignments

1. What do you understand by the term Edge computing?

2. Describe network virtualization

3. Describe the 5G network

Module 2: Network Models and Protocols

- Unit 1:** Network Models
- Unit 2:** Network Protocols
- Unit 3:** Internet Protocol (IP) Addressing, and Subnetting
- Unit 4:** IP Subnetting

Unit 1: Network Models

1.0. Introduction

Do you know how the internet actually works behind the scenes? We use it every day—sending emails, browsing websites, streaming videos—but what's the magic that makes all this possible? Let's dive into Network Model to demystify how internet functions operate. Don't worry; we'll keep it simple and conversational! The Network Model serves as a conceptual framework that explains how data travels from one device to another over the internet. Think of it as a set of layers, each with specific responsibilities, working together to ensure smooth communication. The most commonly referenced models are the OSI (Open Systems Interconnection) model with seven layers and the simplified TCP/IP model with four layers. Given the growing complexity of computer networks, during the 1970s network researchers proposed various reference models to facilitate the description of network protocols and services. Of these, the Open Systems Interconnection (OSI) model was probably the most influential. It served as the basis for the standardization work performed within the ISO to develop global computer network standards. The Internet (TCP/IP) model can be considered as a simplified version of the OSI reference model. These models serve as architectural blueprints, shaping the way data navigates through the complex maze of local and global networks. We will examine the inner workings of connectivity through these lenses. Behind the scenes, these models illustrate how each layer contributes to data transmission and the end user's view and experience. We will explain each of these layers in further detail in this unit.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss the term network model.
- ii. List three network models.
- iii. Itemize three components of network models.
- iv. Explain the TCP/IP model
- v. Explain OSI model
- vi. Give 4 advantages of the OSI model

3.0 Network Model

In the realm of computer science and telecommunications, the term "network model" encompasses various frameworks used to understand and describe the structure and behaviour of networks. These models serve as essential tools for theoretical analysis, practical implementation, and optimization of network systems.

Components of Network Models

Nodes: These are individual devices or endpoints within a network, such as computers, routers, and switches.

Links: The connections between nodes, which can be physical (cables, fibre optics) or wireless.

Protocols: Rules governing data transmission across the network, ensuring successful communication and data integrity.

Addresses: Unique identifiers assigned to nodes, facilitating accurate data routing and delivery.

Network models are integral to the functioning and advancement of modern communication systems. By providing structured frameworks for understanding network dynamics, they facilitate the design, implementation, and maintenance of efficient and effective network infrastructure. A comprehensive grasp of these models not only enhances technical expertise but also promotes innovation in an increasingly interconnected world.

3.1 OSI Model

The Open Systems Interconnection (OSI) model is a framework that conceptualizes how computers within a network communicate. It splits this process into seven distinct layers, each one playing a specific role within the overall operation. The OSI model was the first standardized model for network communications. It was adopted by the International Organization for Standardization (ISO) as an international standard in 1984. Even prior to this in the early 1908s, it had become an unofficial industry standard, having already been adopted by all major computer and telecommunication companies.

Layer		Description
7	Application	Enables the end user applications to access the network
6	Presentation	Converts data into an understandable format and encrypts it
5	Session	Enables and manages sessions of communication between computers
4	Transport	Ensures reliable transfer of packets of data between users
3	Network	Determines the transmission path of data using routing protocols
2	Data Link	Formats data on the network and sends it from node to node
1	Physical	Manages the relationship between the physical device and the transmission medium, be it wireless or cable

Figure 1: OSI Model

The OSI model divides networking into a "vertical stack" consisting of 7 layers as shown in Figure 1. Networking starts on the application layer at the top (Layer 7) and proceeds to the bottom layer (Layer 1). It is then passed back up the same hierarchy.

3.2 Seven Layers of OSI Model

OSI Layer 7: Application

The application layer provides networking processes to the end user. Its protocols enable Layer 7 to work with whatever data the client is using. For example, it works with HTTP to support

applications such as web browsers, specific applications and email clients. The application layer sends data to and receives data from, the presentation layer.

OSI Layer 6: Presentation

The presentation layer handles “syntax processing” in other words, it converts data from one format to another. For example, when you complete a transaction on an e-commerce site, your transaction data will exist in the language of the application. So, the presentation layer takes this data and translates it into “networking-compatible” language, enabling it to be transmitted as part of the networking process. The presentation layer takes any data transmitted by the application layer and prepares it for transmission over the session layer. Layer 6 Presentation examples include encryption, ASCII, EBCDIC, TIFF, GIF, PICT, JPEG, MPEG, and MIDI.

OSI Layer 5: Session

The session layer handles connections between different devices in the network. It creates communication channels, called sessions, between devices. It is responsible for opening sessions, ensuring they remain open and functional while data is being transferred, and closing them when communication ends. The session layer can also set checkpoints during a data transfer. This means if the session is interrupted, devices can easily resume data transfer from the last checkpoint. Once a “session” is established, the data is passed to or from the Transport Layer.

OSI Layer 4: Transport

The Transport Layer within the OSI model manages the transfer of data across network connections, or hosts. It takes data transferred in the session layer and breaks it into “segments” to be transmitted. Conversely, as data comes back through the stack, the Transport Layer is responsible for reassembling segmented data and turning it back into a format readable by the Session Layer. The transport layer also manages flow control. It sends data at a rate that matches the connection speed of the receiving device. It also carries out error control, ensuring data was received correctly. If not, it requests the data again. Layer 4 Transport examples include SPX, TCP, and UDP.

OSI Layer 3: Network

Layer 3, the Network Layer, manages the routing of the data. When data arrives here, each frame of data is screened to confirm whether it reached its intended target. This layer manages the mapping between addresses, forwarding packets of data to and from IP addresses. It uses network layer protocols to create logical paths, known as virtual circuits. A packet is a data unit that contains source and destination IP addresses, a protocol specification field, data, and a trailer field. This includes information about error connections. Segmenting the data as packets makes it easier to retransmit interrupted or lost pieces of data. Layer 3 Network examples include AppleTalk DDP, IP, and IPX.

OSI Layer 2: Data Link

The Data Link layer is often divided into two sub-layers: media access control (MAC) and the logical link control (LLC) layer. The MAC sublayer controls how a computer on the network gains access to the data and permission to transmit it. The LLC layer controls frame synchronization, flow control, and error checking. The Data Link layer packages Bits of information into data frames to be sent to the physical layer. A data frame is the protocol data unit, or PDU of the data link layer and represents a group of information. The physical layer just accepts and transmits data without analyzing the meaning of its structure. Therefore, it's the job of the data link layer to create and recognize frame boundaries. Layer 2 Data Link examples include PPP, FDDI, ATM, IEEE 802.5/ 802.2, IEEE 802.3/802.2, HDLC, and Frame Relay.

OSI Layer 1: Physical

OSI Model Layer 1, or the physical layer, conveys the bit stream electrical impulse, light, or radio signal through the network at the electrical and mechanical level. It provides the hardware means of sending and receiving data on a carrier, including defining cables, cards, and physical aspects. Fast Ethernet, RS232, and ATM are protocols with physical layer components.

Advantages

The OSI model helps users and operators of networks in a number of ways.

- i. It enables technicians to easily determine what components and software are required to build their network.
- ii. Makes it easy to visualize the role and communication process of each component in a

network.

- iii. Having a standard model enables troubleshooting, identifying which network layer is causing an issue and focusing efforts on that layer.

The OSI model also helps network device manufacturers and networking software vendors:

- i. It enables them to create fully interoperable devices and software, compatible with products from any other vendor.
- ii. It clearly defines which parts of the network their components should work with.
- iii. It enables them to communicate to users which network layers their product interacts with.

3.2. TCP/IP Model

The TCP/IP model is the default method of data communication on the Internet. It was developed by the United States Department of Defense to enable the accurate and correct transmission of data between devices. It breaks messages into packets to avoid having to resend the entire message in case it encounters a problem during transmission. Packets are automatically reassembled once they reach their destination. Every packet can take a different route between the source and the destination computer, depending on whether the original route used becomes congested or unavailable/IP divides communication tasks into layers that keep the process standardized, without hardware and software providers doing the management themselves. The data packets must pass through four layers before they are received by the destination device, and then TCP/IP goes through the layers in reverse order to put the message back into its original format.

As a connection-based protocol, the TCP establishes and maintains a connection between applications or devices until they finish exchanging data. It determines how the original message should be broken into packets, numbers reassembles the packets and sends them on to other devices on the network, such as routers, security gateways, and switches, then on to their destination. TCP also sends and receives packets from the network layer, handles the transmission of any dropped packets, manages flow control, and ensures all packets reach their destination.

A good example of how this works in practice is when an email is sent using SMTP from an email server. To start the process, the TCP layer in the server divides the message into packets, numbers them, and forwards them to the IP layer, which then transports each packet to the destination email server. When packets arrive, they are handed back to the TCP layer to be reassembled into the

original message format and handed back to the email server, which delivers the message to a user's email inbox. TCP/IP uses a three-way handshake to establish a connection between a device and a server, which ensures multiple TCP socket connections can be transferred in both directions concurrently. Both the device and server must synchronize and acknowledge packets before communication begins, then they can negotiate, separate, and transfer TCP socket connections.

The TCP/IP model defines how devices should transmit data between them and enable communication over networks and large distances. The model represents how data is exchanged and organized over networks. It is split into four layers, which set the standards for data exchange and represent how data is handled and packaged when being delivered between applications, devices, and servers. The four layers of the TCP/IP model are presented in Figure 2.

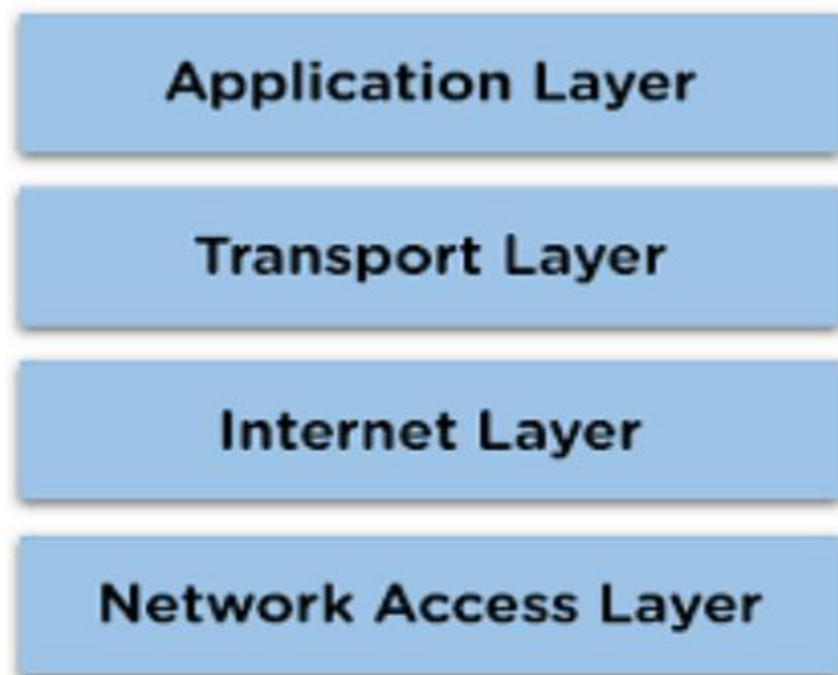


Figure 2: TCP/IP Model

3.3. Four Layers of TCP/IP Model

Datalink layer: The datalink layer defines how data should be sent, handles the physical act of sending and receiving data, and is responsible for transmitting data between applications or devices on a network. This includes defining how data should be signalled by hardware and other transmission devices on a network, such as a computer's device driver, an Ethernet cable, a network

interface card (NIC), or a wireless network. It is also referred to as the link layer, network access layer, network interface layer, or physical layer and is the combination of the physical and data link layers of the Open Systems Interconnection (OSI) model, which standardizes communications functions on computing and telecommunications systems.

Internet layer: The internet layer is responsible for sending packets from a network and controlling their movement across a network to ensure they reach their destination. It provides the functions and procedures for transferring data sequences between applications and devices across networks.

Transport layer: The transport layer is responsible for providing a solid and reliable data connection between the original application or device and its intended destination. This is the level where data is divided into packets and numbered to create a sequence. The transport layer then determines how much data must be sent, where it should be sent, and at what rate. It ensures that data packets are sent without errors and in sequence and obtains the acknowledgement that the destination device has received the data packets.

Application layer: The application layer refers to programs that need TCP/IP to help them communicate with each other. This is the level that users typically interact with, such as email systems and messaging platforms. It combines the session, presentation, and application layers of the OSI model.

4.0. Self-Assessment Questions

1. OSI stands for _____

- (a) Open system interconnection
- (b) Open system Information
- (c) Open system Intervention

2. The number of layers in the ISO OSI reference model is _____

- (a) 5 (b) 6 (c) 7

3. Application layer is implemented in _____

- (a) End system (b) NIC (c) Ethernet

(b)

4. The function of the presentation layer includes _____

(a) Data compression (b) Data encryption (c) Data Description

5. Discuss the term network model.

(a) Frameworks used to understand and describe the structure and behaviour of networks.

(b) serve as essential tools for theoretical analysis, practical implementation, and optimization of network systems.

(c) Real Network

6. List three types of network models.

7. Which of the following best describes a component of network models?

(a) Nodes, modem, Router

(b) Nodes, Links, Protocols, Addresses

(c) Backbone, Protocols, Access Layer

8. Explain the TCP/IP model

9. Explain the OSI model

10. Discuss Peer-to-Peer network model

11. Explain client-server model

5.0. Tutor Marked Assignments

1. With the aid of a diagram describe the OSI model.

2. With the aid of a diagram describe the TCP/IP model.

3. Differentiate between TCP/IP and OSI models.

Unit 2: Network Protocols

1.0. Introduction

Have you ever wondered how your computer talks to others across the world? This is made possible by network protocols. Let's dive into what they are and why they're so important in our everyday digital lives. Think of network protocols as the languages that computers use to communicate with each other. Just like how we need a common language to have a conversation, computers need these protocols to send and receive data smoothly. Without them, transferring information over the internet would be chaotic and error-prone. Imagine trying to send a letter without any rules about addresses or postal codes. It would probably get lost, right? Network protocols establish the rules and formats that ensure data reaches the correct destination efficiently and accurately. They handle everything from how data is broken down into packets to how those packets are reassembled at the other end. In this unit, key network protocols like HTTP, FTP, IP, TCP etc. are discussed.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Define network protocols.
- ii. Explain key network protocols.
- iii. Discuss Interoperability and Standardization.

3.0. Network Protocols

In the contemporary digital era, the seamless transmission of data across vast and intricate networks is fundamental to both personal and professional spheres. Central to this capability are network protocols. A network protocol is a set of established rules that specify how to format, send and receive data so that computer network endpoints, including computers, servers, routers and virtual machines, can communicate despite differences in their underlying infrastructures, designs or standards. To successfully send and receive information, devices on both sides of a communication exchange must accept and follow protocol conventions. In networking, support for protocols can be built into the software, hardware or both. Without network protocols, computers and other devices would not know how to engage with each other. As a result, except for speciality networks built around a specific architecture, few networks would be able to

function, and the internet as we know it wouldn't exist. Virtually all network end users rely on network protocols for connectivity. These protocols are essential for ensuring interoperability among diverse hardware and software, enabling the global exchange of information that underpins modern society.

3.1. Understanding Network Protocols

A network protocol is a formal set of guidelines that dictate how data is formatted, transmitted, and received in a network. These protocols encompass mechanisms for device identification, data compression, error checking, and signalling. By adhering to common protocols, devices ranging from smartphones to servers can communicate effectively, regardless of underlying differences in their design or operation.

3.2 Key Network Protocols and Their Functions

Several protocols are fundamental to network communication:

- i. **Transmission Control Protocol/Internet Protocol (TCP/IP):** Often referred to collectively as the Internet Protocol Suite, TCP/IP is essential for internet connectivity. TCP ensures reliable transmission by establishing a connection-oriented session and handling packet sequencing and error checking. IP handles addressing and routing, directing packets to their intended destinations using IP addresses.
- ii. **User Datagram Protocol (UDP):** Unlike TCP, UDP is connectionless and does not guarantee delivery, making it faster but less reliable. It is used in applications where speed is critical, such as streaming media or online gaming.
- iii. **Hypertext Transfer Protocol (HTTP/HTTPS):** HTTP is the foundation of data communication on the World Wide Web. HTTPS adds a layer of security through encryption via the Transport Layer Security (TLS) protocol, ensuring secure communication over a computer network.
- iv. **Simple Mail Transfer Protocol (SMTP):** SMTP is the standard protocol for sending emails across networks. It works in conjunction with protocols like the Post Office Protocol (POP) and the Internet Message Access Protocol (IMAP), which are used for retrieving emails from a server.
- v. **File Transfer Protocol (FTP):** FTP is used for transferring files between a client and a

server on a network. It allows users to upload and download files, manage directories, and change permissions on files.

- vi. **Domain Name System (DNS):** DNS translates human-readable domain names into IP addresses. This system allows users to access websites using domain names instead of having to remember numerical IP addresses.
- vii. **The Role of Protocols in Network Security**
- viii. With the proliferation of cyber threats, security has become a critical aspect of network protocols. Protocols such as Secure Shell (SSH), Secure Sockets Layer (SSL), and its successor TLS provide encryption and secure channels for data transmission. These protocols protect sensitive information from interception and unauthorized access.
- ix. **Transport Layer Security (TLS):** TLS is widely used to secure web communications and is the backbone of HTTPS. It encrypts data between the application layer and the transport layer.
- x. **Internet Protocol Security (IPsec):** Operating at the network layer, IPsec secures IP communications by authenticating and encrypting each IP packet in a communication session.
- xi. **Secure Shell (SSH):** SSH provides a secure channel over an unsecured network, commonly used for secure logins, file transfers, and port forwarding.

3.3. Interoperability and Standardization

The effectiveness of network protocols depends on widespread adoption and adherence to standards. Organizations such as the Internet Engineering Task Force (IETF) and the Institute of Electrical and Electronics Engineers (IEEE) play pivotal roles in developing and promoting these standards.

- i. **IETF:** An open standards organization, the IETF develops and promotes voluntary internet standards, particularly the standards that comprise the Internet Protocol Suite (TCP/IP).
- ii. **IEEE:** The IEEE sets standards for a broad range of technologies, including network protocols like Ethernet (IEEE 802.3) and Wi-Fi (IEEE 802.11).

Standardization ensures that devices and applications developed by different manufacturers or organizations can work together seamlessly, fostering innovation and global connectivity.

Network protocols are the foundational elements that enable the vast, interconnected digital landscape of today. They provide the necessary rules and conventions for devices to communicate, ensuring that data is transmitted accurately, efficiently, and securely. As technology continues to advance, the development and refinement of network protocols remain critical to meeting the evolving demands of global communication. Continued innovation and cooperation in protocol development are essential to harnessing the full potential of current and future networked systems.

4.0. Self-Assessment Questions

1. Which protocol is used to automatically provide IP addresses to network computers
(a) DHCP (b)DNS (c)ARP
2. Which protocol is used to upload email messages to the email server
(a) DNS (b)SMTP(c)HTTP
3. Which protocol is used to resolve domain name IP addresses?
(a) DNS (b) TCP (c) DHCP
4. Which protocol is used to resolve IP address to MAC address
(a) DNS(b)ARP (c) FTP
5. Define network protocols.
6. List and explain key network protocols.
7. What do you understand by Interoperability and Standardization on the internet?

5.0. Tutor Marked Assignments

1. Differentiate between TCP/IP and UDP
2. Which of the two (TCP/IP or UDP) would you use in an online gaming application
3. Which of the two (TCP/IP or UDP) would you use to implement a real-time chatting application?

Unit 3: Internet Protocol (IP) Addressing, and Subnetting

1.0. Introduction

Have you ever wondered how devices communicate over the internet? One of the key components that make this possible is the IP address. An IP address, or Internet Protocol address, is a unique numerical label assigned to every device connected to a computer network that uses the Internet Protocol for communication. Think of it as a mailing address for your computer or smartphone. Just like your home address allows mail to reach you, an IP address enables data to find your device on a network.

IP addresses are essential because they provide a way for devices to identify and communicate with each other. Whenever you send an email, browse a website, or stream a video, your device uses an IP address to request and receive information. Without IP addresses, data wouldn't know where to go, and the internet wouldn't function as we know it. Understanding IP addresses is fundamental to navigating today's digital world. They are critical for device identification and communication over networks. Whether you're troubleshooting connectivity issues or simply curious about how the internet works, knowing about IP addresses equips you with valuable knowledge. This unit presents Internet protocol. The concepts discussed are, IPV4, IPV6, IP subnetting, and NAT.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Describe IP Address
- ii. State five classes of IP address with 2 examples each.
- iii. Discuss NAT.
- iv. Explain three types of NAT.
- v. State 3 advantages of NAT.
- vi. State 2 importance of IP Addressing.
- vii. Discuss three challenges associated with IP addressing.
- viii. Describe the structure of IPV4 and IPV5.
- ix. Differentiate between static and dynamic IP addresses.
- x. Students should be able to subnet IPV4.

- xi. Differentiate between public and private IP addresses.

3.0 IP Address

The Internet Protocol (IP) is a fundamental component of the modern Internet infrastructure, facilitating communication between devices across diverse networks. IP addressing is a critical aspect of this protocol, enabling the unique identification and location of devices in a network. This article provides a comprehensive overview of IP addressing, including its structure, types, and significance in network communication. An IP address is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. It serves two principal functions: host or network interface identification and location addressing. IP addresses allow devices to find and communicate with each other over an IP-based network, such as the Internet.

IPv4 Addresses

IPv4 (Internet Protocol version 4) is the fourth version of the Internet Protocol and the first widely used protocol. An IPv4 address is a 32-bit number, which allows for approximately 4.3 billion unique addresses. These addresses are typically represented in dotted-decimal notation, consisting of four octets separated by periods (e.g., 192.168.1.1).

Structure of IPv4 Addresses

IPv4 addresses are divided into two parts:

- i. **Network Portion:** Identifies the specific network to which the host belongs.
- ii. **Host Portion:** Identifies the specific host (device) within the network.

The division between the network and host portions is determined by the subnet mask or the address class.

IPv4 Address Classes

IPv4 addresses are categorized into five classes (A to E) based on the leading bits:

- i. **Class A:** Designed for large networks with many hosts. The first octet ranges from 1 to 126.
- ii. **Class B:** Intended for medium-sized networks. The first octet ranges from 128 to 191.

- iii. **Class C:** Meant for small networks. The first octet ranges from 192 to 223.
- iv. **Class D:** Reserved for multicast groups. The first octet ranges from 224 to 239.
- v. **Class E:** Reserved for experimental purposes. The first octet ranges from 240 to 254.

Subnetting in IPv4

Subnetting is the process of dividing a network into smaller subnetworks (subnets). It enhances network efficiency and security by reducing broadcast domains and organizing networks logically. Subnet masks determine how an IP address is partitioned into the network and host portions. For example, a subnet mask of 255.255.255.0 implies that the first three octets represent the network portion, and the last octet represents the host portion.

3.2 IPv6 Addresses

Due to the exhaustion of IPv4 addresses, IPv6 (Internet Protocol version 6) was developed as a successor to IPv4. IPv6 addresses are 128-bit numbers, significantly increasing the available address space to accommodate the growing number of internet-connected devices.

Structure of IPv6 Addresses

IPv6 addresses are represented as eight groups of four hexadecimal digits, separated by colons (e.g., 2001:0db8:85a3:0000:8a2e:0370:7334). Leading zeros in each group can be omitted, and consecutive groups of zeros can be compressed using a double colon (::).

Types of IPv6 Addresses

Unicast Addresses: Identifying a single interface.

Multicast Addresses: Identifying a group of interfaces, typically on different nodes.

Anycast Addresses: Assigned to multiple interfaces; packets are delivered to the nearest interface as determined by routing protocols.

Public vs. Private IP Addresses

IP addresses are categorized as public or private:

Public IP Addresses: Globally unique addresses assigned by the Internet Assigned Numbers Authority (IANA) and can be accessed over the Internet.

Private IP Addresses: Reserved for use within private networks and not routable on the internet. Common private IP ranges include:

Class A: 10.0.0.0 to 10.255.255.255

Class B: 172.16.0.0 to 172.31.255.255

Class C: 192.168.0.0 to 192.168.255.255

3.3. Network Address Translation (NAT)

Network Address Translation (NAT) is a fundamental networking process that enables multiple devices on a local network to access external networks using a single public IP address. Operating at the network layer, NAT modifies the network address information in the IP header of packets while they are in transit across a router or firewall, facilitating the communication between private networks and the public Internet. The primary motivation for the development of NAT was the depletion of IPv4 addresses. As the number of devices connected to the Internet grew exponentially, the available pool of IPv4 addresses became insufficient. NAT addresses this limitation by allowing a set of private IP addresses within a local network to be mapped to one or more public IP addresses. This conserves the global IPv4 address space and extends the usable life of IPv4.

There are several types of NAT.

1. **Static NAT:** This involves a one-to-one mapping between a private IP address and a public IP address. It is often used when a device inside a private network needs to be accessible from the outside Internet.
2. **Dynamic NAT:** This maps a private IP address to a public IP address selected from a pool of available public addresses. Unlike static NAT, the mapping is not fixed and can change over time.

3. Port Address Translation (PAT): Also known as NAT Overload, PAT allows multiple devices on a local network to be mapped to a single public IP address but with a different port number for each session. This is the most common form of NAT used in home and office networks.

NAT offers several advantages:

IP Address Conservation: By allowing multiple devices to share a single public IP address, NAT significantly reduces the demand for global IP addresses.

Enhanced Security: NAT acts as a basic firewall by preventing external hosts from initiating unsolicited connections to devices within the private network. Since internal IP addresses are not exposed to the external network, it adds a layer of privacy.

However, NAT also introduces certain challenges:

End-to-End Connectivity Issues: NAT can interfere with protocols that require end-to-end connectivity and maintain IP address integrity, such as some VPNs and peer-to-peer applications.

Performance Overhead: The translation process consumes processing power and can become a bottleneck, especially in high-throughput networks.

Complications with IPv6 Transition: NAT is less relevant with IPv6 due to the vastly increased address space, but its widespread use can slow down the adoption of IPv6.

In modern networking, NAT remains a critical component despite the push towards IPv6. It provides a practical solution for IP address shortages and adds a layer of network security. Network administrators must understand NAT to effectively manage network resources and troubleshoot connectivity issues.

3.4. Dynamic vs. Static IP Addresses

Static IP Addresses: Permanently assigned to a device, providing a consistent address, which is essential for hosting services like web servers.

Dynamic IP Addresses: Assigned temporarily by Dynamic Host Configuration Protocol (DHCP) servers. Commonly used for end-user devices where a permanent address is not necessary.

3.5. Importance and Challenges of IP Addressing in Network Communication

IP addressing is essential for:

- i. **Routing:** Determines the path data packets take to reach their destination.
- ii. **Security:** Enables implementation of firewalls and access controls based on IP addresses.
- iii. **Network Management:** Facilitates device identification, troubleshooting, and network resource allocation.

Challenges of IP Addressing

- i. **IPv4 Exhaustion:** The limited address space of IPv4 led to the depletion of available addresses.
- ii. **Transition to IPv6:** Adoption of IPv6 is ongoing but faces challenges due to compatibility issues and the need for infrastructure upgrades.
- iii. **Security Concerns:** IP spoofing and address-based attacks require advanced security measures.
- iv. IP addressing is a cornerstone of internet communication, enabling the unique identification and connection of devices worldwide. Understanding the fundamentals of IP addressing, including the differences between IPv4 and IPv6, the role of subnetting, and the importance of NAT, is crucial for professionals in networking and information technology. As the internet continues to evolve, IP addressing schemes and protocols will adapt to meet the growing demands for connectivity and security.

4.0. Self-Assessment Questions

1. Which of these IP addresses belong to class c.

(a)192.168.2.1(b)10.0.0.1(c) 16.0.0.0

2. Which of these IP addresses is referred to as a loopback address.

(a)127.0.0.1 (b)17.0.0.1(c) 126.0.0.1

3. How many usable hosts does class C address contain.

(a)254(b)252(c)256

4. Which of these options represents the subnet mask of the class A address?

(a) 255.0.0.0 (b) 255.255.0.0 (c) 255.255.255.0

5. What do you understand by the term IP Address?

6. Which of these best describes the classes of IP Address

(a) Class A, Class B, Class C, Class D, Class E

(b) Class A, Class B, Class C, Class D, Class E, Class F

(c) Class 1, Class 2, Class 3, Class 4, Class 5

8. Give two examples of IP addresses in each class of IP address.

9. Discuss the term NAT.

10. explain three types of NAT.

11. state 3 advantages of NAT.

12. state 2 importance of IP Addressing.

13. discuss three challenges associated with IP addressing.

14. Create two networks with 5 nodes each and address the networks with IPV 4 and IPV6 respectively.

15. differentiate between static and dynamic IP addresses.

16. differentiate between public and private IP addresses.

5.0. Tutor Marked Assignments

1. Which of the classes of IP addresses can be used on a production network?
2. Calculate the number of usable hosts in each class of IP address.
3. What do you understand by the term IP exhaustion?

Unit 4: IP Subnetting

1.0.Introduction

IP address exhaustion refers to the depletion of available IP addresses under the current Internet Protocol version 4 (IPv4) system. IPv4 provides approximately 4.3 billion unique addresses. While that might seem like a vast number, the explosive growth of the Internet, smartphones, computers, and Internet of Things (IoT) devices has rapidly consumed these addresses. The exhaustion of IPv4 addresses became a critical concern because every device connecting to the internet requires a unique IP address. With the limited number of IPv4 addresses, organizations and internet service providers have faced challenges in allocating addresses to new devices and services. This shortage impacts the ability to expand networks and can hinder innovation and growth in the digital space.

Subnetting is a method used to divide a larger network into smaller, more manageable subnetworks or subnets. By breaking down a network into subnets, organizations can efficiently use their allocated IP addresses and improve network performance and security. When a network is subnetted, the IP address space is divided into smaller blocks. This division allows for better management of IP addresses by allocating them according to the specific needs of different network segments. Subnetting reduces the wastage of IP addresses that can occur when large blocks are assigned to networks that don't need them.

For example, if a company has a network that requires only 50 IP addresses, assigning an entire class C network (which provides 256 addresses) would be inefficient. By subnetting, the company can allocate just the number of addresses needed, conserving the remaining addresses for other uses. Subnetting also enhances network performance. Smaller networks reduce broadcast traffic, which can consume bandwidth and degrade network efficiency. Additionally, subnetting improves security by isolating network segments, making it easier to monitor and control access. This unit presents IP V4 subnetting. You will also learn different ways to represent IP address and their subnet mask.

2.0.Learning Outcomes

At the end of this unit, you should be able to:

- i. Define IP subnetting
- ii. Students should be able to represent IP addresses in CIDR notation.
- iii. Students should be able to subnet IPV 4 addresses and provide subnetting information.

IP Subnetting

IP subnetting is a fundamental concept in computer networking that involves dividing a single IP network into smaller, more manageable sub-networks, known as subnets. This practice enhances both the efficiency and security of network management. By breaking down a larger network into subnets, organizations can optimize their address space, improve network performance, and simplify troubleshooting.

3.0 IP Subnet Mask

An IP subnet mask is used to indicate the host and network portion of an IP address. The subnet mask of Class A, Class B and Class C is presented in Figure 3.

Class A	Network (8 bit)	Host (24 bit)
	255 . 0 . 0 . 0	
Class B	Network (16 bit)	Host (16 bit)
	255 . 255 . 0 . 0	
Class C	Network (24 bit)	Host (8 bit)
	255 . 255 . 255 . 0	

Figure 3: IP Addressing Subnet Mask

When changing a number in the Network part of an IP address we will be in a different network from the previous address. For example, the IP address 11.0.0.1 belongs to class A and has a default subnet mask of 255.0.0.0; if we change the number in the first octet (a block of 8 bits, the first octet is the leftmost 8 bits) we will create a different network. For example, 12.0.0.1 is in a different network from 11.0.0.1. But if we change a number in the Host part, we are still in the same Network. For example, 11.1.0.1 is in the same network as 11.0.0.1. The problem here is if we want to create 300 networks how can we do that? In the above example, we can only create different networks when changing the first octet so we can create a maximum of 255 networks because the first octet can only range from 1 to 255 (in fact it is much smaller because class A only ranges from 1 to 126). Now we have to use a technique called "subnetting" to achieve our purpose.

"Subnetting" means we borrow some bits from the Host part to add to the Network part. This allows us to have more networks than using the default subnet mask. For example, we can borrow some bits in the next octet to make the address 11.1.0.1 belong to a different network from 11.0.0.1. Do you remember that I said, "In the subnet mask, bit 1 represents for Network part while bit 0 presents for Host part"? Well, this also means that we can specify how many bits we want to borrow by changing how many bits 0 to bit 1 in the subnet mask. Let's come back to our example with the IP 11.0.0.1, we will write all numbers in binary form as shown in Figure 4 to reveal what a computer really sees in an IP address.

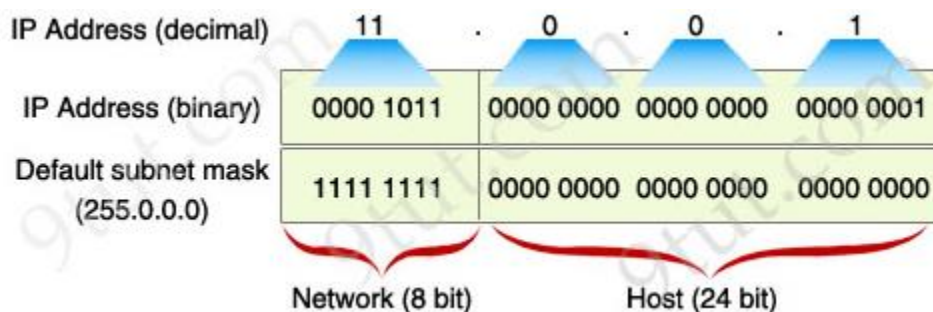


Figure 4: Class A Binary Representation of IP Address

Now you can clearly see that the subnet mask will decide which is the Network part, and which is the Host part. By borrowing 8 bits, our subnet mask will be like this:

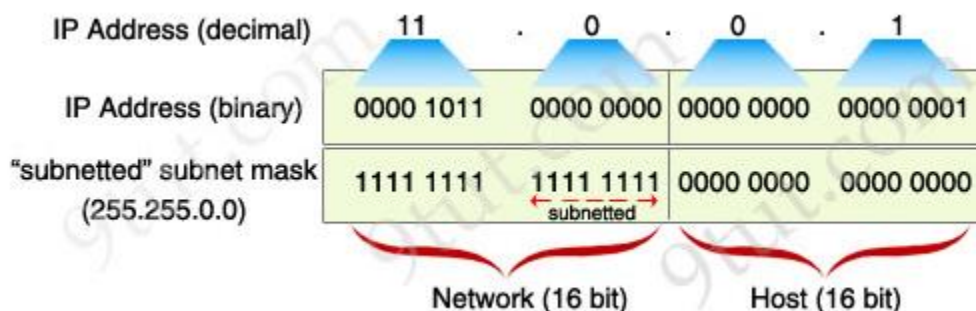


Figure 5: Class B Binary Representation of IP Address

After changing the second octet of the subnet mask from all "0" to all "1", the Network part is now extended. Now we can create new networks by changing the number in the first or second octet. This greatly increases the number of networks we can create. With this new subnet mask, IP 11.1.0.1 is in a different network from IP 11.0.0.1 because "1" in the second octet now belongs to

the Network part. So, in conclusion, we "subnet" by borrowing bit "0" in the Host portion and converting it to bit "1". The number of borrowed bits depends on how many networks we need.

Note: A rule of borrowing bits is we can only borrow bit 0 from the left to the right without skipping any bit 0. For example, you can borrow like this: "1111 1111. 1100 0000.0000 0000.0000 0000" but not this: "1111 1111. 1010 0000.0000 0000.0000 0000". In general, just make sure all your bit "1" s are successive on the left and all your bit "0" s are successive on the right.

Exercise 1

Your company has just been assigned the network 4.0.0.0. How many subnets and hosts-per-subnet can you create with a subnet mask of 255.255.255.0?

Solution

First of all, you have to specify which class this network belongs to. It belongs to class A (simply, class A ranges from 1 to 126) and its default subnet mask is 255.0.0.0. Therefore, if we use a subnet mask of 255.255.255.0, it means we borrowed 16 bits (to convert from 0 to 1).

255.0.0.0 = 1111 1111.0000 0000.0000 0000.0000 0000

255.255.255.0 = 1111 1111.1111 1111.1111 1111.0000 0000

Now use our above formulas to find the answers:

The number of newly created subnets = $2^{16} = 65536$ (with 16 being the borrowed bits). The number of hosts per subnet = $2^8 - 2 = 254$ (with 8 being the bit "0" s left in the 255.255.255.0 subnet mask)

Exercise 2

Your company has just been assigned the network 130.0.0.0. How many subnets and hosts-per-subnet can you create with a subnet mask of 255.255.128.0?

Solution

130.0.0.0 belongs to class B with the default subnet mask of 255.255.0.0. But is the subnet mask of 255.255.128.0 strange? Ok, let's write all subnet masks in binary:

255.255.128.0 = 1111 1111.1111 1111.1000 0000.0000 0000

This is a valid subnet because all bits "1" s and "0" s are successive. Compared to the default subnet mask, we borrowed only 1 bit:

255.255.0.0 = 1111 1111.1111 1111.0000 0000.0000 0000

Therefore:

The number of newly created subnets = $2^1 = 2$ (with 1 being the borrowed bits)

The number of hosts per subnet = $2^{15} - 2 = 32766$ (with 15 being the bit "0"s left in the 255.255.128.0 subnet mask)

4.0. Self-Assessment Questions

1. Which of these is not true about IP address

(a) IP subnetting is a fundamental concept in computer networking that involves dividing a single IP network into smaller, more manageable sub-networks, known as subnets.

(b) This practice enhances both the efficiency and security of network management.

(c) This practice describes a network

2. You need to subnet a network that has 5 subnets, each with at least 16 hosts. Which classful subnet mask would you use?

(a) 255.255.255.192

(b) 255.255.255.224

(c) 255.255.255.240

3. The network address of 172.16.0.0/19 provides how many subnets and hosts.

(a) 7 subnets, 30 hosts each

(b) 8 subnets, 8,190 hosts each

(c) 8 subnet, 2,046 hosts each

4. Which mask should be used on point-to-point WAN links to reduce the waste of IP addresses?

(a)/27 (b) /28 (c)/30

5. What is the subnetwork address for a host with the IP address 200.10.5.68/28.

- (a) 172.16.45.0
- (b) 172.16.45.4
- (c) 172.16.45.12

5.0. Tutor Marked Assignments

Provide the subnetting detail of the following IP address.

1. 10.20.4.0/13
2. 192.168.2.0/28
3. 192.168.4.0/27
4. 192.168.4.0/30

Module 3: Web Design and Web Development

- Unit 1:** World Wide Web
- Unit 2:** Tools for Web Design and Web Development
- Unit 3:** Client-Side Programming
- Unit 4:** Server-Side Programming

Unit 1: World Wide Web

1.0. Introduction

Have you ever wondered how we can access so much information from around the world with just a few clicks? That's all thanks to the World Wide Web, often simply called the Web. It's an incredible system that has transformed the way we live, work, and connect with each other. The World Wide Web is a vast collection of interconnected documents and other resources, linked by hyperlinks and URLs. It's a way to access information over the medium of the Internet. While the Internet is the network of computers themselves, the Web is a service that uses this network to share information in a user-friendly way.

The web was invented in 1989 by Sir Tim Berners-Lee, a British scientist working at CERN, the European Organization for Nuclear Research. He wanted to find a new way for scientists to easily share data and results. By proposing a system of hyperlinked documents accessible over the Internet, he laid the foundation for the web as we know it today. The first website went live in 1991, and it wasn't long before the web expanded beyond the scientific community.

At its core, the web uses a technology called Hypertext Transfer Protocol (HTTP) to transmit data. Websites are built using Hypertext Markup Language (HTML), which structures the content. When you enter a web address into your browser or click a link, your browser sends an HTTP request to the web server hosting that site. The server then responds by sending back the requested web page, which your browser displays. The World Wide Web has revolutionized how we access and share information. It has made vast amounts of knowledge accessible to people all over the globe, breaking down geographical barriers. This has had profound impacts on education,

commerce, entertainment, and social interaction. Businesses can reach customers worldwide, students can access educational resources from home, and we can stay connected with friends and family no matter where they are.

Today, the web is more interactive and dynamic than ever. Technologies like JavaScript, CSS, and various web frameworks allow developers to create rich, engaging websites and applications. We can stream videos, play games, shop online, and collaborate in real time. The rise of social media platforms has also changed how we communicate and share our lives with others. With all its benefits, it's important to be mindful of safety when using the web. Protecting personal information, using strong passwords, and being cautious about the websites you visit and the information you share can help safeguard against potential risks like identity theft or malware. This unit presents the history of the World Wide Web, HTML, DNS and packet-to-server interaction.

2.0.Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss the brief history of WWW
- ii. Explain the term HTML
- iii. Explain the term DNS
- iv. Discuss client-to-server interaction.
- v. Discuss Packet

3.0 History of the Word wide web

The World Wide Web, commonly referred to as the Web, is a global information system that enables users to access and share data over the Internet. It was invented by Sir Tim Berners-Lee in 1989 while working at the European Organization for Nuclear Research (CERN). The advent of the World Wide Web revolutionized the way information is disseminated and consumed, leading to significant transformations in communication, commerce, and society as a whole. The fundamental technologies that constitute the World Wide Web include Hypertext Markup Language (HTML), Uniform Resource Locators (URLs), and Hypertext Transfer Protocol (HTTP). HTML is the standard markup language used to create and structure content on the Web. URLs serve as specific addresses that identify resources on the Internet, allowing users to locate

and access web pages. HTTP is the protocol that facilitates the transfer of data between a web server and a client browser, enabling the retrieval and display of web content.

Initially, the World Wide Web was conceived as a tool to meet the demand for automatic information-sharing among scientists in universities and institutes around the world. In 1991, the first website was launched, providing information about the World Wide Web project itself. The Web began to gain widespread popularity with the development of user-friendly web browsers such as Mosaic in 1993 and Netscape Navigator in 1994. These browsers made it easier for the general public to navigate the Web, contributing to its exponential growth in the mid-1990s.

The impact of the World Wide Web on various sectors is profound. In the realm of education, it has facilitated access to a vast array of information and educational resources, supporting distance learning and research collaboration. In commerce, it has given rise to e-commerce, transforming traditional business models and enabling online retail, digital marketing, and global trade. Governments and public institutions utilize the Web to provide services, enhance transparency, and engage with citizens through e-government initiatives. Despite its numerous benefits, the World Wide Web also presents challenges. Issues such as the digital divide, cybersecurity threats, and the proliferation of misinformation and disinformation are significant concerns. The digital divide refers to the gap between individuals who have access to modern information and communication technology and those who do not, which can exacerbate social and economic inequalities. Cybersecurity threats, including data breaches and cyberattacks, pose risks to personal privacy and national security. The spread of misinformation can undermine public discourse and erode trust in institutions. Efforts to address these challenges involve international cooperation, policy development, and technological innovation. Initiatives to expand Internet access aim to bridge the digital divide, while cybersecurity measures and regulations strive to protect users and infrastructure. Promoting digital literacy is essential to enable individuals to navigate the Web safely and critically assess the information they encounter.

The World Wide Web is a pivotal invention that has reshaped modern life. Its ability to connect people, facilitate the exchange of information, and drive economic growth underscores its significance. Ongoing efforts to mitigate its challenges are crucial to ensure that the Web remains a secure, inclusive, and empowering resource for all users.

3.1 Hyper Text Markup Language (HTML)

HTML, or HyperText Markup Language, serves as the foundational framework for web development. As a markup language, HTML structures content on the internet, allowing browsers to render text, images, and other media in a coherent format. Its significance extends beyond mere formatting; it enables the interactivity and accessibility of digital content. At its core, HTML functions through a series of elements and tags that define the structure and semantics of web pages. Each tag is enclosed within angle brackets and can include attributes that provide additional information or functionality. For instance, the `` tag creates hyperlinks, fostering navigation between different web pages. This interconnectedness is crucial for creating a seamless user experience and establishing the hyperlinked nature of the web.

As web technologies have evolved, so too has HTML. The introduction of HTML5 brought significant enhancements, including new semantic elements like `

`, `

`, and ``, which improved the clarity of document structure. These advancements facilitate better search engine optimization (SEO) and enhance accessibility for users with disabilities, aligning with contemporary web standards. Furthermore, HTML works in conjunction with CSS (Cascading Style Sheets) and JavaScript, two other core technologies that shape the visual presentation and behaviour of web pages. While HTML provides the skeleton, CSS allows for aesthetic customization, and JavaScript introduces dynamic functionalities, such as form validation and interactive features.

HTML is a critical element in the ongoing evolution of the internet. Understanding HTML and its pertinent role within the broader scope of web development equips graduates with the essential skills needed to navigate the complexities of modern digital ecosystems. As technology continues to advance, mastery of HTML remains a crucial asset for anyone aspiring to thrive in various fields, including web design, software development, and digital marketing.

Clients and servers

Computers connected to the Web are called **clients** and **servers**. A simplified diagram of how they interact might look like this:

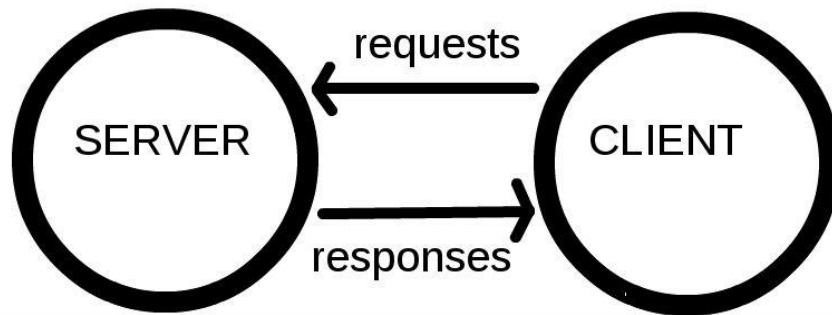


Figure 6: Interaction between the Client and Server

- i. Clients are the typical Web user's Internet-connected devices (for example, your computer connected to your Wi-Fi or your phone connected to your mobile network) and Web-accessing software available on those devices (usually a web browser like Firefox or Chrome).
- ii. Servers are computers that store webpages, sites, or apps. When a client device wants to access a webpage, a copy of the webpage is downloaded from the server onto the client machine to be displayed in the user's web browser as shown in Figure 6.
- iii. **Your Internet connection:** Allows you to send and receive data on the Web. It's basically like the street between your house and the shop.
- iv. **TCP/IP:** Transmission Control Protocol and Internet Protocol are communication protocols that define how data should travel across the Web. This is like the transport mechanisms that let you place an order, go to the shop, and buy your goods. In our example, this is like a car or a bike (or however else you might get around).
- v. **DNS:** Domain Name System Servers are like an address book for websites. When you type a web address in your browser, the browser looks at the DNS before retrieving the website. The browser needs to find out which server the website lives on, so it can send HTTP messages to the right place (see below). This is like looking up the address of the shop so you can access it.
 - a. **HTTP:** Hypertext Transfer Protocol is an application protocol that defines a language for clients and servers to speak to each other. This is like the language you use to order your goods.
 - b. **Component files:** A website is made up of many different files, which are like the different parts of the goods you buy from the shop. These files come in two main

types:

- i. **Code files:** Websites are built primarily from HTML, CSS, and JavaScript, though you'll meet other technologies a bit later.
- ii. **Assets:** This is a collective name for all the other stuff that makes up a website, such as images, music, video, Word documents, and PDFs.

When you type a web address into your browser (for our analogy that's like walking to the shop):

- i. The browser goes to the DNS server and finds the real address of the server that the website lives on (you find the address of the shop).
- ii. The browser sends an HTTP request message to the server asking it to send a copy of the website to the client (you go to the shop and order your goods). This message, and all other data sent between the client and the server, is sent across your internet connection using TCP/IP.
- iii. Provided the server approves the client's request, the server sends the client a "200 OK" message, which means "Of course, you can look at that website! Here it is", and then starts sending the website's files to the browser as a series of small chunks called data packets (the shop gives you your goods, and you bring them back to your house).
- iv. The browser assembles the small chunks into a complete website and displays it to you (the goods arrive at your door — new stuff, awesome!).

3.2. DNS

Real web addresses are not the memorable strings typed into the address bar to find favourite websites. They are strings of numbers, like this: 63.245.217.105. This is called an IP address, and it represents a unique location on the Web. However, it's not very easy to remember, is it? That's why Domain Name Servers were invented. These are special servers that match up a web address you type into your browser (like "mozilla.org") to the website's real (IP) address.

Websites can be reached directly via their IP addresses. Try going to the Mozilla website by typing 63.245.217.105 into the address bar on a new browser tab.

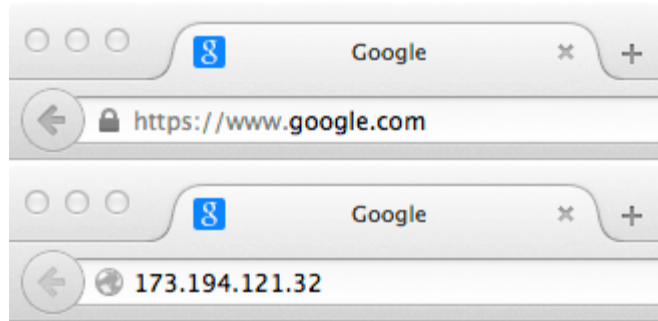


Figure 7: IP Address and DNS

3.3. Packets

The term "packets" is used to describe the format in which the data is sent from server to client. Basically, when data is sent across the Web, it is sent as thousands of small chunks, so that many different web users can download the same website at the same time. If websites were sent as single big chunks, only one user could download one at a time, which obviously would make the Web very inefficient and not much fun to use.

4.0. Self-Assessment Questions

1. What does the term HTTPS mean?

- (a) Hyper Text Transfer Protocol Secure
- (b) Hyper Text Translation Protocol Service
- (c) Hyper Text Translation Protocol Secure

2. The term _____ is used to describe the format in which the data is sent from server to client.

- (a) Packet (b) HTTP (c) Data

3. _____ are communication protocols that define how data should travel across the Web.

- (a) TCP (b) IP (c) FTP

4. _____ are like an address book for websites.

- (a) DNS (b) ARP (c) Protocols

5. Write a brief history of WWW

6. Which of these is true about HTML?

- (a) Define how the Web works.

(b) HTML structures content on the internet, allowing browsers to render text, images, and other media in a coherent format.

(c) HTML define how the internet works.

7. Which of these is true about DNS?

(a) DNS match up a web address you type into your browser (like "mozilla.org") to the website's real (IP) address.

(b) DNS match up different domain names together.

(c) DNS enables IP addresses to be useful on networks

8. Discuss the interaction between client and server.

9. discuss Packet

5.0. Tutor Marked Assignments

1. What do you understand by the term HTML
2. With the aid of a diagram describe DNS.

Unit 2: HTML and CSS

1.0.Introduction

If you've ever browsed the internet and wondered how websites are created, the answer lies in two fundamental technologies: HTML and CSS. These are the cornerstone languages that form the structure and style of every web page you visit. Let's explore what they are and how they work together to bring websites to life, HyperText Markup Language is the standard language used to create and structure content on the web. Think of HTML as the blueprint of a house, outlining where each room (or element) should be placed. It allows developers to organize content such as text, images, links, and more using a system of tags and elements.

At its core, HTML uses tags enclosed in angle brackets, like `

` for a paragraph or `

` for a main heading. These tags tell web browsers how to display the content enclosed within them, or Cascading Style Sheets, which is the language used to describe the presentation of an HTML document. While HTML structures the content, CSS styles it. Using our house analogy, if HTML is the blueprint, CSS is the paint, wallpaper, and decorations. With CSS, you can control the colour, font, layout, and overall aesthetic of your web pages. This separation of content (HTML) and style (CSS) makes it easier to maintain and update websites. By learning them, you unlock the ability to create and style web pages, bringing your ideas to life online.

This unit introduces students to web design using HTML, CSS and Latest tools.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Identify 5 tools for web design.
- ii. Create simple web pages using HTML.
- iii. Design the web pages using CSS.

3.0. Tools for Web design

- i. **A computer.** Maybe that sounds obvious to some people, but some of you are reading this article from your phone or a library computer. For serious web development, it's better to invest in a desktop computer (Windows, Mac, or Linux).
- ii. **A text editor,** to write code in. This could be a text editor (e.g. Brackets, Atom or Visual Studio Code), or a hybrid editor (e.g. Dreamweaver). Office document editors are not suitable for this use, as they rely on hidden elements that interfere with the rendering engines used by web browsers.
- iii. **Web browsers,** to test code in. Currently, the most-used browsers are Firefox, Chrome, Opera, Safari, and Internet Explorer. You should also test how your site performs on mobile devices and on any old browsers your target audience may still be using extensively (such as IE 6–8.)
- iv. **A graphics editor,** like GIMP, Paint.NET, or Photoshop, to make images for your web pages.
- v. **A version control system,** to manage files on servers, collaborate on a project with a team, share code and assets, and avoid editing conflicts. Right now Git is the most popular version control tool, and the GitHub code hosting service, based on Git, is also very popular.
- vi. **An FTP program,** used on older web hosting accounts to manage files on servers (Git is increasingly replacing FTP for this purpose). There are loads of (S)FTP programs available including Cyberduck, Fetch, and FileZilla.
- vii. **An automation system,** like Grunt or Gulp, performs repetitive tasks automatically, for example, minifying code and running tests.
- viii. Templates, libraries, frameworks, etc., to speed up writing common functionality.
- ix. More tools besides!

3.1 HTML Basics

HTML (Hypertext Markup Language) is the code that is used to structure and display a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables. As the title suggests, this article will give you a basic understanding of HTML and what its function is. HTML is not a programming language; it is a markup language and is used to tell your browser how to display the web pages you visit. It can be as complicated or as simple as the web designer wishes it to be. HTML consists of a series

of elements, which you use to enclose, or wrap, different parts of the content to make it appear a certain way, or act a certain way. The enclosing tags can make a word or an image a hyperlink to somewhere else, can italicize words, and can make the font bigger or smaller, and so on. For example, take the following line of content:

My cat is very grumpy.

If we wanted the line to stand by itself, we could specify that it is a paragraph by enclosing it in a paragraph tag (`<p>`) element:

```
<p>My cat is very grumpy</p>
```

Anatomy of an HTML element

Let's explore this paragraph element a bit further.



Figure 8: Paragraph Element

The main parts of our element are:

1. **The opening tag:** This consists of the name of the element (in this case, `p`), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect — in this case where the start of the paragraph is.
2. **The closing tag:** This is the same as the opening tag, except that it includes a forward slash before the element name. This states where the element ends — in this case where the end of the paragraph is. Failing to include a closing tag is one of the common beginner errors and can lead to strange results.
3. **The content:** This is the content of the element, which in this case is just text.
4. **The element:** The opening tag, plus the closing tag, plus the content, equals the element.

Elements can also have attributes, which look like this:

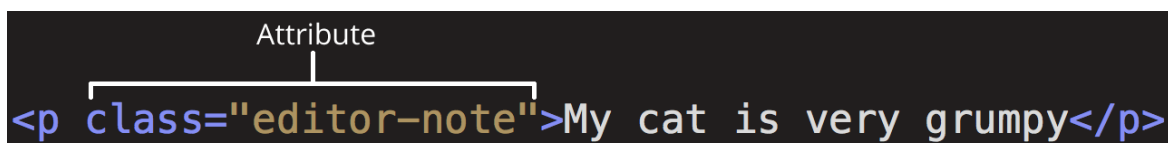


Figure 9: Element Attributes

Attributes contain extra information about the element that you don't want to appear in the actual content. Here, `class` is the attribute *name*, and `editor-note` is the attribute *value*. The `class` attribute allows you to give the element an identifier that can be later used to target the element with style information and other things.

An attribute should always have:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name is followed by an equals sign.
3. Opening and closing quote marks wrapped around the attribute value.

Nesting elements

You can put elements inside other elements too — this is called **nesting**. If we wanted to state that our cat is VERY grumpy, we could wrap the word "very" in a `` element, which means that the word is to be strongly emphasized:

```
<p>My cat is <strong>very</strong> grumpy.</p>
```

You do however need to make sure that your elements are properly nested: in the example above we opened the `<p>` element first, then the `` element, therefore we have to close the `` element first, then the `<p>`. The following is incorrect:

```
<p>My cat is <strong>very grumpy. </p></strong>
```

The elements have to open and close correctly so they are clearly inside or outside one another. If they overlap like above, then your web browser will try to make a best guess at what you were trying to say, and you may well get unexpected results. So don't do it!

Empty elements

Some elements have no content and are called **empty elements**. Take the `` element we already have in our HTML:

```

```

This contains two attributes, but there is no closing `` tag and no inner content. This is because an image element doesn't wrap content to have an effect on it. Its purpose is to embed an image in the HTML page, in the place it appears.

Anatomy of an HTML document


```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    
  </body>
</html>
```

Here we have:

- `<!DOCTYPE html>` — the doctype. In the mists of time, when HTML was young (about 1991/2), doctypes were meant to act as links to a set of rules that the HTML page had to follow to be considered good HTML, which could mean automatic error checking and other useful things. However, these days no one really cares about them, and they are really just a historical artifact that needs to be included for everything to work right. For now, that's all you need to know.
- `<html></html>` — the `<html>` element. This element wraps all the content on the entire page and is sometimes known as the root element.
- `<head></head>` — the `<head>` element. This element acts as a container for all the stuff you want to include on the HTML page that *isn't* the content you are showing to your page's viewers. This includes things like keywords and a page description that you want to appear in search results, CSS to style our content, character set declarations, and more.
- `<body></body>` — the `<body>` element. This contains *all* the content that you want to show to web users when they visit your page, whether that's text, images, videos, games, playable audio tracks, or whatever else.
- `<meta charset="utf-8">` — this element sets the character set your document should use to utf-8, which includes most characters from all known human languages. Essentially it can now handle any textual content you might put on it. There is no reason not to set this, and it can help avoid some problems later on.

- `<title></title>` — this sets the title of your page, which is the title that appears in the browser tab the page is loaded in, and is used to describe the page when you bookmark/favourite it.

Images

```

```

As we said before, it embeds an image into our page in the position it appears. It does this via the `src` (source) attribute, which contains the path to our image file.

We have also included an `alt` (alternative) attribute. In this attribute, you specify descriptive text for users who cannot see the image, possibly because:

1. They are visually impaired. Users with significant visual impairments often use tools called Screen Readers to read out the `alt` text to them.
2. Something has gone wrong causing the image to not display. For example, try deliberately changing the path inside your `src` attribute to make it incorrect. If you save and reload the page, you should see something like this in place of the image:

The key words about `alt` text are "descriptive text". The `alt` text you write should provide the reader with enough information to have a good idea of what the image conveys. In this example, our current text of "My test image" is no good at all. A much better alternative for our Firefox logo would be "The Firefox logo: a flaming fox surrounding the Earth."

Marking up text

This section will cover some of the basic HTML elements you'll use for marking up text.

Headings

Heading elements allow you to specify that certain parts of your content are headings —or subheadings— of your content. In the same way that a book has a main title, chapter titles and subtitles, an HTML document can too. HTML contains six heading levels, `<h1>`–`<h6>` although you'll commonly only use 3–4 at most:

```
<h1>My main title</h1>
```

```
<h2>My top-level heading</h2>
```

`<h3>My subheading</h3>`

`<h4>My sub-subheading</h4>`

Now try adding a suitable title to your HTML page just above your `` element.

Paragraphs

As explained above, `<p>` elements are for containing paragraphs of text; you'll use these frequently when marking up regular text content:

`<p>This is a single paragraph</p>`

Lists

A lot of the Web's content is lists and HTML has special elements for these. Marking up lists always consists of at least two elements. The most common list types are ordered and unordered lists:

1. **Unordered lists** are lists where the order of the items doesn't matter, like a shopping list. These are wrapped in a `` element.
2. **Ordered lists** are for lists where the order of the items does matter, like a recipe. These are wrapped in an `` element.

Each item inside the lists is put inside an `` (list item) element.

For example, if we wanted to turn the part of the following paragraph fragment into a list:

`<p>At Mozilla, we're a global community of technologists, thinkers, and builders working together ... </p>`

We could modify the markup to this:

`<p>At Mozilla, we're a global community of</p>`

``

`technologists`

`thinkers`

`builders`

``

`<p>working together ... </p>`

Links

Links are very important — they are what makes the Web A WEB. To add a link, we need to use a simple element — `<a>` — the *a* being short for "anchor". To make text within your paragraph into a link, follow these steps:

1. Choose some text. We chose the text "Mozilla Manifesto".
2. Wrap the text in an `<a>` element, like so:
`<a>Mozilla Manifesto`
3. Give the `<a>` element an href attribute, like so:
`Mozilla Manifesto`
4. Fill in the value of this attribute with the web address that you want the link to link to:
`<a href="https://www.`

3.2 CSS Basics

CSS (Cascading Style Sheets) is the code you use to style your webpage. Like HTML, CSS is not really a programming language. It is a *style sheet language*, that is, it lets you apply styles selectively to elements in HTML documents. For example, to select **all** the paragraph elements on an HTML page and turn the text within them red, you'd write this CSS:

```
p {  
  color: red;  
}
```

Type of CSS

- i. External CSS
- ii. Inline CSS
- iii. Internal CSS

External CSS

Let's try it out: paste those three lines of CSS into a new file in your text editor, and then save the file as `style.css` in your styles directory.

But you still need to link the CSS to your HTML document, otherwise, the CSS styling won't affect how your browser displays the HTML document

1. Open your `index.html` file and paste the following line somewhere in the head, that is, between the `<head>` and `</head>` tags:
`<link href="styles/style.css" rel="stylesheet" type="text/css">`

2. Save index.html and load it in your browser. You should see something like this:

If your paragraph text is now red, congratulations, you've now written your first successful

Inline CSS

An inline style may be used to apply a unique style to a single element. To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property. `<p style="color:red;">This is a paragraph.</p>`

External CSS

An internal style sheet may be used if one single HTML page has a unique style. The internal style is defined inside the `<style>` element, inside the head section.

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <style>
```

```
      p {color: red;}
```

```
    </style>
```

```
  </head>
```

```
    <body>
```

```
      <p>This is a paragraph. </p>
```

```
    </body>
```

```
</html>
```



Figure 10: Cascading Stylesheet

The whole structure is called a **rule set** (often "rule" for short). Note also the names of the individual parts:

Selector

The HTML element name is at the start of the rule set. It selects the element(s) to be styled (in this case, p elements). To style a different element, just change the selector.

Declaration

A single rule like color: red; specifying which of the element's **properties** you want to style.

Properties

Ways in which you can style a given HTML element. (In this case, colour is a property of the p elements.) In CSS, you choose which properties you want to affect in your rule.

Property value

To the right of the property, after the colon, we have the **property value**, which allows us to choose one of many possible appearances for a given property (there are many colour values besides red).

Note the other important parts of the syntax:

- i. Each rule set (apart from the selector) must be wrapped in curly braces ({}).
- ii. Within each declaration, you must use a colon (:) to separate the property from its values.
- iii. Within each rule set, you must use a semicolon (;) to separate each declaration from the next one.

So, to modify multiple property values at once, you just need to write them separated by semicolons, like this:

```
p {  
  color: red;  
  width: 500px;  
  border: 1px solid black;  
}
```

Selecting multiple elements

You can also select multiple types of elements and apply a single rule set to all of them. Include multiple selectors separated by commas. For example:

```
p, li, h1 {  
  color: red;  
}
```

Different types of selectors

There are many different types of selectors. Above, we only looked at **element selectors**, which select all elements of a given type in the HTML documents. But we can make more specific selections than that. Here are some of the more common types of selectors:

Table 1: CSS Selector

Selector name	What does it select	Example
Element selector (sometimes called a tag or type selector)	All HTML element(s) of the specified type.	p Selects <p>
ID selector	The element on the page with the specified ID (on a given HTML page, you're only allowed one element per ID).	#my-id Selects <p id="my-id"> or

Class selector	The element(s) on the page with the specified class (multiple class instances can appear on a page).	.my-class Selects <p class="my-class"> and
Attribute selector	The element(s) on the page with the specified attribute.	img[src] Selects but not
Pseudo class selector	The specified element(s), but only when in the specified state, e.g. being hovered over.	a: hover Selects <a>, but only when the mouse pointer is hovering over the link.

Anything in a CSS document between `/*` and `*/` is a **CSS comment**, which the browser ignores when it renders the code. This is a place for you to write helpful notes on what you are doing.

Boxes

One thing you'll notice about writing CSS is that a lot of it is about boxes — setting their size, colour, position, etc. Most of the HTML elements on your page can be thought of as boxes sitting on top of each other.

Not surprisingly, CSS layout is based principally on the *box model*. Each of the blocks taking up space on your page has properties like this:

- i. padding, the space just around the content (e.g., around paragraph text)
- ii. border, the solid line that sits just outside the padding
- iii. margin, the space around the outside of the element

In this section, the following were made use of:

- i. width (of an element)

- ii. background-color, the colour behind an element's content and padding
- iii. color, the color of an element's content (usually text)
- iv. text-shadow: sets a drop shadow on the text inside an element
- v. display: sets the display mode of an element (don't worry about this yet)

Changing the page color

```
html {  
  background-color: #00539F;  
}
```

This rule sets a background color for the whole page. Change the color code above to whatever color you chose when planning your site.

Sorting the body out

```
body {  
  width: 600px;  
  margin: 0 auto;  
  background-color: #FF9500;  
  padding: 0 20px 20px 20px;  
  border: 5px solid black;  
}
```

Now for the body element. There are quite a few declarations here, so let's go through them all one by one:

- width: 600px; — this forces the body to always be 600 pixels wide.
- margin: 0 auto; — When you set two values on a property like margin or padding, the first value affects the element's top **and** bottom side (make it 0 in this case), and the second value the left **and** right side (here, auto is a special value that divides the available horizontal space evenly between left and right). You can also use one, three, or four values, as documented here.

- `background-color: #FF9500;` — as before, this sets the element's background color. I used a sort of reddish orange for the body as opposed to dark blue for the html element. Go ahead and experiment. Feel free to use white or whatever you like.
- `padding: 0 20px 20px 20px;` — we have four values set on the padding, to make a bit of space around our content. This time we are setting no padding on the top of the body, and 20 pixels on the left, bottom and right. The values set top, right, bottom, left, in that order.
- `border: 5px solid black;` — this simply sets a 5 pixel wide solid black border on all sides of the body.

Positioning and styling our main page title

```
h1 {
  margin: 0;
  padding: 20px 0;
  color: #00539F;
  text-shadow: 3px 3px 1px black;
}
```

You may have noticed there's a horrible gap at the top of the body. That happens because browsers apply some **default styling** to the `<h1>` element (among others), even when you haven't applied any CSS at all! That might sound like a bad idea, but we want even an unstyled webpage to have basic readability. To get rid of the gap we overrode the default styling by setting `margin: 0;`.

Next up, we've set the heading's top and bottom padding to 20 pixels, and made the heading text the same color as the html background color. One rather interesting property we've used here is `text-shadow`, which applies a text shadow to the text content of the element. Its four values are as follows:

- The first pixel value sets the **horizontal offset** of the shadow from the text — how far it moves across a negative value should move it to the left.
- The second-pixel value sets the **vertical offset** of the shadow from the text — how far it moves down, in this example; a negative value should move it up.
- The third-pixel value sets the **blur radius** of the shadow — a bigger value will mean a

blurrier shadow.

- iv. The fourth value sets the base colour of the shadow.

Again, try experimenting with different values to see what you can come up with.

Centering the image

```
img {  
  display: block;  
  margin: 0 auto;  
}
```

Finally, we'll centre the image to make it look better. We could use the margin: 0 auto trick again as we did earlier for the body, but we also need to do something else. The body element is **block level**, meaning it takes up space on the page and can have margin and other spacing values applied to it. Images, on the other hand, are **inline** elements, meaning they can't. So to apply margins to the image, we have to give the image block-level behavior using display: block;

Table 2: Web Page Layout Elements

Element	Description
Header	The header tag is used to add a header section in HTML web page. All the content inside this tag will be on top of the webpage.
Nav	It represents a section of a page within the webpage, where it has hyperlinks to other pages or parts within the page (just like the menu bar).
Section	It defines a major part of the web page where all the important content will be displayed.
Footer	The footer tag defines the footer section of the webpage. This section contains copyright information and other important details. It will be always at the bottom of the webpage.

Article	It specifies an independent self-containing content such as a forum post, magazine, any blog post and so on.
Aside	It specifies a section of content that is directly or indirectly related to the main content (like a sidebar).
Summary	It specifies a caption for the <details> element.
Details	It specifies a tag for additional information. It requires the <summary> element.

3.3 Website Layout

In this section, you will put together what you have learnt so far to create website layout. An effective website layout is essential for enhancing user experience and achieving organizational objectives. A well-structured layout facilitates easy navigation, allowing users to find information efficiently. This improves user satisfaction and increases the likelihood of repeat visits. The layout influences the visual hierarchy of a website, guiding users' attention to important content and calls to action. By strategically organizing elements such as headings, images, and buttons, designers can convey the site's purpose clearly. This clarity can lead to higher engagement and conversion rates. HTML 5 layout elements are presented in Table 2. In the section you will learn about two website layouts. The first one employed div tag as shown in Figure 11, and 12 while the second one make use of HTML 5 header, article, footer, section and nav tag as presented in Figure 14 , 15 , 16.

```

<!DOCTYPE html>
<html lang="en">
<head>
|   <link rel="stylesheet" type="text/css" href="style.css" media="screen" />
</head>
<body >
|   <div class="header">
|       You can put you website banner here
|   </div>
|   <div class="container">
|       <div class="sidebar">
|           <p>Menu</p>
|           <a href="#">Home</a>
|           <a href="#">Learn HTML</a>
|           <a href="#">About Us</a>
|       </div>
|       <div class="content">
|           <h1>Home</h1>
|           <p>Put your content here</p>
|       </div>
|   </div>
|   <div class="footer">
|       Website footer
|   </div>
</body>
</html>

```

Figure 11: HTML Code for Website Layout 1

```
.header {
  background-color: #b3e0f2;
  text-align: center;
  padding: 20px;
  font-size: 2em;
  font-weight: bold;
}
.container {
  display: flex;
}
.sidebar {
  width: 20%;
  background-color: #f4f4f4;
  padding: 20px;
}
.content {
  width: 80%;
  background-color: #f9f9f9;
  padding: 20px;
}
.footer {
  background-color: #b3e0f2;
  text-align: center;
  padding: 10px;
  position: fixed;
  width: 100%;
  bottom: 0;
}
```

Figure 12: CSS Code for Website Layout 1

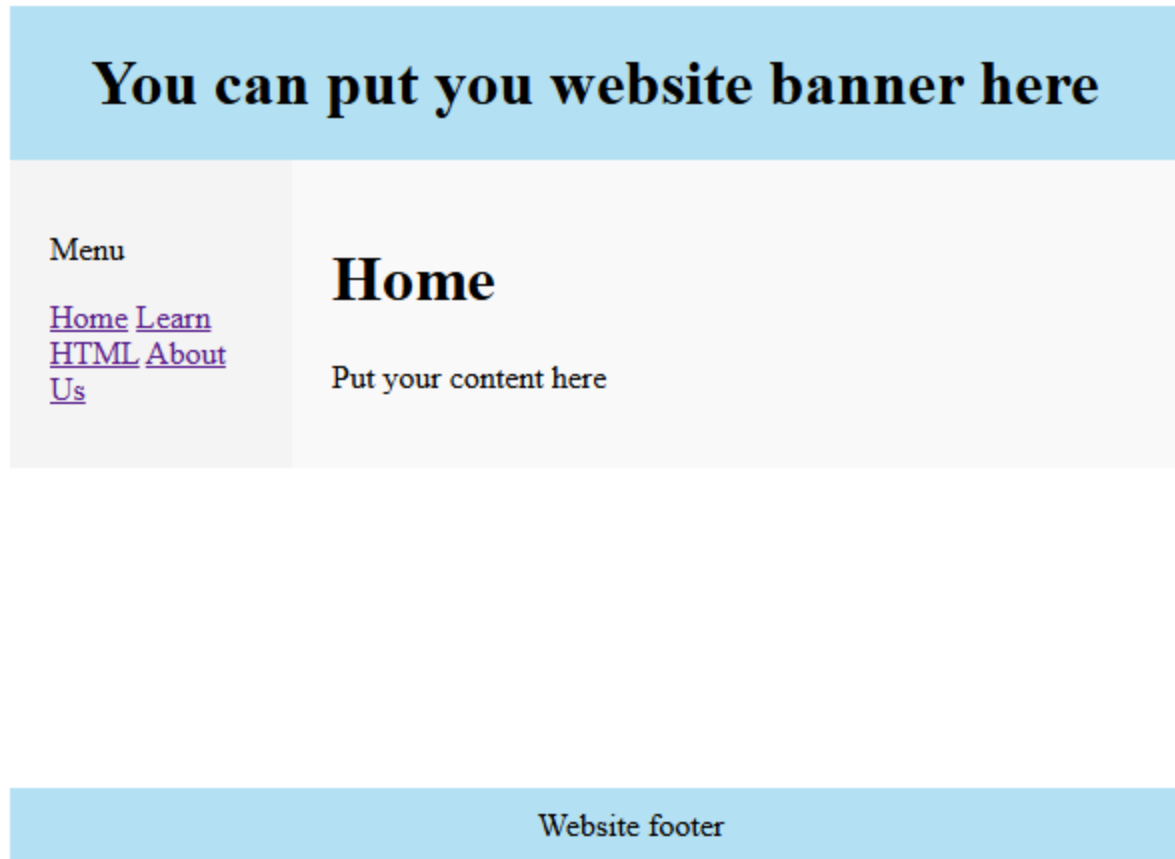


Figure 13: Web View of Website Layout 1

```

<!DOCTYPE html>
<html>

<head>
  <title>Layout structure of HTML</title>

  <link rel="stylesheet" type="text/css" href="style.css" media="screen" />
</head>

<body>
  <!--header segment-->
  <header>
    <div>
      <p>My website</p>
      <p>Layout</p>
    </div>
  </header>
  <section>
    <!--Menu Navigation segment-->
    <nav>
      <ul>
        <li>
          <a href="#">Home</a>
        </li>
        <li>
          <a href="#">Jobs</a>
        </li>
        <li>
          <a href="#">Support</a>
        </li>
        <li>
          <a href="#">About</a>
        </li>
        <li>
          <a href="#">Contact</a>
        </li>
      </ul>
    </nav>
  </section>
</body>
</html>

```

Figure 14: HTML Code for Website Layout 2


```

<!--Content segment-->
<article>
  <h1>Welcome to my first website</h1>
  <p>
    HTML layout refers to the way in which the content of a
    website is organized and structured. It makes the website
    easy to navigate. For example,As you can see we have various
    contents on the page like heading, footer, the home page, etc in a structured way.
  </p>
</article>
</section>
<!--Footer segment-->
<footer>
  <h2>Links</h2>
  <div>
    <a href="#"> About us </a>
    <a href="#"> Refund policy </a>
    <a href="#"> Terms of use </a>
    <a href="#"> Privacy policy </a>
    <a href="#"> FAQ's </a>
    <a href="#"> Affiliates </a>
    <a href="#"> Contact </a>
  </div>
  <div>
    <p>
      Copyrights © XXX
      Private Limited. All rights reserved.
    </p>
  </div>
</footer>
</body>
</html>

```

Figure 15: HTML Code for Content and Footer Segment of Website Layout 2

```

*{box-sizing: border-box;}
header {
  font-size: 25px;
  color: ■ whitesmoke;
  padding: 1px;
  text-align: center;
  background-color: ■ lightslategray;
}
nav {
  float: left;
  width: 20%;
  height: 350px;
  background: ■ lightgray;
  padding: 20px;
}
nav ul { padding: 1px;}
article {
  float: left;
  padding: 20px;
  width: 80%;
  background-color: ■ lightyellow;
  height: 350px;
}
footer {
  background-color: ■ lightslategray;
  padding: 10px;
  text-align: center;
  color: ■ white;
  padding-top: 20px;
  padding-bottom: 10px;
}
footer a {
  margin: 10px;
}
footer p {
  margin-top: 30px;
}

```

Figure 16: CSS for Website Layout 2



Figure 17: Web View of Website Layout 2

4.0. Self-Assessment Questions

1. What is HTML?

- a) HTML describes the structure of a webpage.
- b) HTML is the standard markup language mainly used to create web pages.
- c) All of the mentioned.

2.What is the correct syntax of doctype in HTML5?

- a) <doctype html>
- b) <doctype html!>
- c) <!doctype html>

3. Which of the following is used to read an HTML page and render it?

- a) Web server

b) Web network

c) Web browser

4. Which of the following tag is used for inserting the largest heading in HTML?

a) head

b) <h1>

c) <h6>

5. Which of these represent web design tools.

i. computer

ii. text editor

iii. Web browsers

iv. graphics editor

v. version control system

vi. flash drive

vii. Cloud computing

(a) i, ii, iii, vi, vii (b) i, ii, iii, iv (c) i, ii, iii, iv, v

6. Create a parent div tag with dimension 900px width 400 height and float three div tag inside it.

7. With your knowledge of HTML and CSS create a simple web layout containing banner, sidebar, main content and footer.

5.0. Tutor Marked Assignments

Design a simple web page for coffee shop.

Unit 3: Client-Side Programming

1.0.Introduction

Have you ever wondered how websites respond so quickly to your actions without having to reload the entire page? That's the magic of client-side programming. Let's dive into what it is and why it plays such a crucial role in web development. Client-side programming refers to code that runs on the user's computer, typically within a web browser. When you visit a website, your browser downloads HTML, CSS, and JavaScript files from the server. These files work together on your device to display the webpage and make it interactive.

The main goal of client-side programming is to enhance the user experience. It allows web pages to respond instantly to user actions, like clicking buttons or filling out forms, without needing to communicate with the server every time. This makes websites faster and more efficient. Client-side programming is a vital component of modern web development. It makes websites more interactive, responsive, and user-friendly. By learning client-side technologies, you can create richer experiences for web users and build dynamic applications that run smoothly in the browser. This unit presents client-side programming. It also entails java script programming and how to use it to program HTML page.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Give brief discussion about JavaScript.
- ii. Define and use JavaScript Variables.
- iii. Create JavaScript statement.
- iv. Define Functions.
- v. Use 5 DOM events.
- vi. Add a personalized welcome message.

3.0. Java Script Basics

JavaScript is a programming language that adds interactivity to your website (for example: games, responses when buttons are pressed or data entered in forms, dynamic styling, animation). JavaScript ("JS" for short) is a full-fledged dynamic programming language that, when applied to an HTML document, can provide dynamic interactivity on websites. It was invented by Brendan Eich, co-founder of the Mozilla project, the Mozilla Foundation, and the Mozilla Corporation.

JavaScript is incredibly versatile. You can start small, with carousels, image galleries, fluctuating layouts, and responses to button clicks. With more experience you'll be able to create games, animated 2D and 3D graphics, comprehensive database-driven apps, and much more! JavaScript itself is fairly compact yet very flexible. Developers have written a large variety of tools complementing the core JavaScript language, unlocking a vast amount of extra functionality with minimum effort. These include:

1. Application Programming Interfaces (APIs) built into web browsers, providing functionality like dynamically creating HTML and setting CSS styles, collecting and manipulating a video stream from the user's webcam, or generating 3D graphics and audio samples.
2. Third-party APIs to allow developers to incorporate functionality in their sites from other content providers, such as Twitter or Facebook.
3. Third-party frameworks and libraries you can apply to your HTML to allow you to rapidly build up sites and applications.

JavaScript Placement in HTML File

1. Script in `<head>...</head>` section.
2. Script in `<body>...</body>` section.
3. Script in `<body>...</body>` and `<head>...</head>` sections.
4. Script in an external file and then include in `<head>...</head>` section.

```
<script>
  // JavaScript code
</script>
```

Figure 18: JavaScript Tag

JavaScript in `<head>...</head>` section

```

<html>
<head>
  <script type = "text/javascript">
    function sayHello() {
      alert("Hello World")
    }
  </script>
</head>

<body>
  <input type = "button" onclick = "sayHello()" value = "Say Hello" />
</body>
</html>

```

Figure 19: JavaScript Placement in the Header Tag

JavaScript in <body>...</body> section

If you need a script to run as the page loads so that the script generates content in the page, then the script goes in the <body> portion of the document. In this case, you would not have any function defined using JavaScript. Take a look at the code in Figure 20.

```

<html>
<head>
</head>
<body>
  <script type = "text/javascript">
    document.write("Hello World")
  </script>
  <p>This is web page body </p>
</body>
</html>

```

Figure 20: JavaScript Placement in the Body Tag

JavaScript in <body> and <head> Sections

You can put your JavaScript code in <head> and <body> sections altogether as follows:

```
<html>
<head>
  <script type = "text/javascript">
    function sayHello() {
      alert("Hello World")
    }
  </script>
</head>

<body>
  <script type = "text/javascript">
    document.write("Hello World")
  </script>
  <input type = "button" onclick = "sayHello()" value = "Say Hello" />
</body>
</html>
```

Figure 21: JavaScript in Head and Body Section of HTML

JavaScript in External File

As you begin to work more extensively with JavaScript, you will likely find cases where you are reusing identical JavaScript code on multiple pages of a site. You are not restricted to be maintaining identical code in multiple HTML files. The script tag provides a mechanism to allow you to store JavaScript in an external file and then include it in your HTML files. To use JavaScript from an external file source, you need to write all your JavaScript source code in a simple text file with the extension ".js" and then include that file as shown below. For example, you can keep the following content in the filename.js file, and then you can use the sayHello function in your HTML file after including the filename.js file.

filename.js


```
function sayHello() {  
    alert("Hello World")  
}
```

Figure 22: JavaScript in an External File

Here is an example to show how you can include an external JavaScript file in your HTML code using the script tag and its src attribute. You may include the external script reference within the <head> or <body> tag as presented in Figure 23.

```
<html>  
<head>  
    <script type = "text/javascript" src = "filename.js" ></script>  
</head>  
<body>  
    ...  
</body>  
</html>
```

Figure 23: Linking External JavaScript to HTML

3.1 JavaScript Syntax

JavaScript syntax comprises a set of rules that define how to construct a JavaScript code. JavaScript can be implemented using JavaScript statements that are placed within the <script>...</script> HTML tags in a web page. You can place the <script> tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the <head> tags. The <script> tag alerts the browser program to start interpreting all the text between these tags as a script.

Your First JavaScript Code

Let us take a simple example to print out "Hello World". We call document.write method which writes a string into our HTML document. This method can be used to write text, HTML, or both. Take a look at the code in Figure 24.

```

<html>
<head>
  <title> Your first JavaScript program </title>
</head>
<body>
  <script language = "javascript" type = "text/javascript">
    document.write("Hello World!")
  </script>
</body>
</html>

```

Figure 24: JavaScript code to output text

Output:

Hello World!

JavaScript Literals

In Figure 25, 10 is a Number literal and 'Hello' is a string literal.

```

<html>
<body>
  <script>
    document.write(10); // Number Literal
    document.write("<br />"); // To add line-break
    document.write("Hello"); // String Literal
  </script>
</body>
</html>

```

Figure 25: JavaScript Literal

This code will produce the following result

Output:

```
10  
Hello
```

You can use the assignment operator (equal sign) to assign values to the variable. In the below code, variable a contains the numeric value, and variable b contains the text (string).

```
<html>  
<body>  
  <script>  
    let a = 5; // Variable Declaration  
    document.write(a); // Using variable  
    document.write("<br>");  
    let b = "One";  
    document.write(b);  
  </script>  
</body>  
</html>
```

Figure 26: Variable Declaration in JavaScript

This code will produce the following result

Output:

```
5  
One
```

Case Sensitivity

JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters. So, the identifiers Time and TIME will convey different meanings in JavaScript. In the code below, we compare the 'time' and 'Time' strings, which return false.

```

<html>
<body>
  <script>
    let a = "time";
    let b = "Time";
    document.write("a == b? " + (a == b));
  </script>
</body>
</html>

```

Figure 28: Case Sensitivity in JavaScript

This code will produce the following result

Output:

```
a == b? false
```

JavaScript Identifiers

In JavaScript, identifiers are the name of variables, functions, objects, etc. For example, p is an identifier in the below code.

```

<script>
  x = 20

  function test(){

  }
</script>

```

Figure 29: Example of Identifier in JavaScript

The 'test' is an identifier in the below code. Here are the rules which you should follow to define valid identifiers.

- i. Identifiers should always start with the alphabetical characters (A-Z, a-z), \$(dollar sign), or _ (underscore).
- ii. It shouldn't start with digits or hyphens.

- iii. The identifier can also contain digits except for the start of it.

Comments in JavaScript

JavaScript supports both C-style and C++-style comments, thus –

- i. Any text between a `//` and the end of a line is treated as a comment and is ignored by JavaScript.
- ii. Any text between the characters `/*` and `*/` is treated as a comment. This may span multiple lines.
- iii. JavaScript also recognizes the HTML comment opening sequence `<!--`. JavaScript treats this as a single-line comment, just as it does the `//` comment.
- iv. The HTML comment closing sequence `-->` is not recognized by JavaScript so it should be written as `//-->`.

```
<script>
  // This is a comment. It is similar to comments in C++
  /*
   * This is a multi-line comment in JavaScript
   * It is very similar to comments in C Programming
   */
</script>
```

Figure 30: JavaScript Comment

In this example, we have defined `var1` and `va2` and initialized them with number values. After that, we use the `*` operator to get the multiplication result of `var1` and `var2`.

Output Text in JavaScript

`document.write()`

In JavaScript, the simplest way to print "Hello World" is to use `document.write()` method. The `document.write()` method writes the content (a string of text) directly to the HTML document or web page. Let's look at the following:

```
<html>
<head>
  <script>
    document.write("Hello World");
  </script>
</head>
<body>
</body>
</html>
```

Figure 31: Output Text with document.write

We should write the document.write() within the <script> and </script> tags. We can place the <script> inside the <head> or <body> section.

alert() Function

We can use the window **alert()** method to print "Hello Word" in a dialogue box. The alert() is a window method that instructs the browser to display an alert box with the message. We can write alert without passing any message to it.

```
<html>
<head>
  <script>
    alert("Hello World");
  </script>
</head>
<body>
</body>
</html>
```

Figure 32: Output Text with alert function

console.log()

The `console.log()` is a very convenient method to print the message to web Console. It is very useful to debug the JavaScript codes in the browsers. Let's look at the simple application of the `console.log()` to print the "Hello World" to the web console.

```
<html>
<head>
  <script>
    console.log("Hello World");
  </script>
</head>
<body>
  <p> Please open the console before clicking "Edit & Run" button </p>
</body>
</html>
```

Figure 33: Output Text with `console.log()`

innerHTML

The `innerHTML` property of an HTML element defines the HTML content of the element. We can use this property to display the Hello World message. By changing the `innerHTML` property of the element, we can display the message in HTML document or webpage. To use `innerHTML` property of an element, we need to access that element first. We can use `document.getElementById()` method to access the element. Let's look at a complete example.

```

<html>
<head>
  <title>Using innerHTML property</title>
</head>
<body>
  <div id = "output"> </div>
  <script>
    document.getElementById("output").innerHTML = "Hello World";
  </script>
</body>
</html>

```

Figure 34: Output Text with getElementById

JavaScript Variables

JavaScript variables are used to store data that can be changed later on. These variables can be thought of as named containers. You can place data into these containers and then refer to the data simply by naming the container. Before you use a variable in a JavaScript program, you must declare it. In JavaScript, you can declare the variables in 4 ways .

- i. Without using any keywords.
- ii. Using the 'var' keyword.
- iii. Using the 'let' keyword.
- iv. Using the 'const' keyword.

You can follow the syntax below to declare the variables without using any keywords.

```

<script>
Age = 10;
Name ="John";

</script>

```

Figure 35: Declaration of Variable Without Keyword

Furthermore, you can use the var keyword to declare the variables as shown in Figure 36.

```
<script>
  var money;
  var name;
</script>
```

Figure 36: Declaration of Variable you Var Keyword

You can also declare multiple variables with the same **var** keyword as follows:

```
<script>
  var money, name;
</script>
```

Figure 37: Declaration of Multiple Variable with a var Keyword

Variable Initialization using the Assignment Operator Storing a value in a variable is called variable initialization. You can do variable initialization at the time of variable creation or at a later point in time when you need that variable. For instance, you might create a variable named money and assign the value 2000.50 to it later. For another variable, you can assign a value at the time of initialization as follows.

```
<script>
  var name = "Ali";
  var money;
  money = 2000.50;
</script>
```

Figure 38: JavaScript Variable Initialization

JavaScript Data Types

In JavaScript, the variable can hold the value of the dynamic data type. For example, you can store the value of number, string, boolean, object, etc. data type values in JavaScript variables.

```
<script>
  var num = 765; // Number
  var str = "Welcome"; // String
  var bool = false; // Boolean
</script>
```

Figure 39: JavaScript Data Type

JavaScript Object

In JavaScript, the object data type allows us to store the collection of the data in the key-value format. There are multiple ways to define the object, which we will see in the Objects chapter. Here, we will create an object using the object literals. In the example below, we used the '{}' (Object literals) to create an obj object. The object contains the 'animal' property with the string value, the 'legs' property with the number value, and the value of the 'color' variable is assigned to the 'hourseColor' property. The JSON.stringify() method converts the object to strings and shows it in the output.

```
<html>
<head>
  <title> JavaScript Object </title>
</head>
<body>
  <script>
    let color = "Brown";
    const obj = {
      animal: "Hourse",
      legs: 4,
      hourseColor: color
    }
    document.write("The given object is: " + JSON.stringify(obj) + "<br/>")
  </script>
</body>
</html>
```

Figure 40: JavaScript Object

JavaScript Array

In JavaScript, the array is a list of elements of the different data types. You can create an array using two square brackets '[]' and insert multiple comma separated values inside the array.

```
<html>
<head>
  <title> JavaScript Array </title>
</head>
<body>
  <script>
    const colors = ["Brown", "red", "pink", "Yellow", "Blue"];
    document.write("The given array is: " + colors + "<br/>");
  </script>
</body>
</html>
```

Figure 41: JavaScript Array

JavaScript Date

You can use the JavaScript Date object to manipulate the date. In the example below, we used the Date() constructor to create a date. In the output, you can see the current date and time according to your time zone.

```
<html>
<head>
  <title> JavaScript Date </title>
</head>
<body>
  <script>
    let date = new Date();
    document.write("The today's date and time is: " + date + "<br/>");
  </script>
</body>
</html>
```

Figure 42: JavaScript Date

3.2 JavaScript Function

A function in JavaScript is a group of reusable code that can be called anywhere in your program. It eliminates the need of writing the same code again and again. It helps programmers in writing modular codes. Functions allow a programmer to divide a big program into a number of small and manageable functions. Like any other advanced programming language, JavaScript also supports all the features necessary to write modular code using functions. You must have seen functions like `alert()` and `write()` in the earlier chapters. We were using these functions again and again, but they had been written in core JavaScript only once. JavaScript allows us to write our own functions as well. This section explains how to write your own functions in JavaScript.

Function Definition

Before we use a function, we need to define it. The most common way to define a function in JavaScript is by using the `function` keyword, followed by a unique function name, a list of parameters (that might be empty), and a statement block surrounded by curly braces. All statements you need to execute on the function call must be written inside the curly braces.

```
function functionName(parameter-list) {  
    statements  
}
```

Figure 43: JavaScript Function

The basic syntax to define the function in JavaScript is as follows. JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page. When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc. Developers can use these events to execute JavaScript coded responses, which cause buttons to close windows, messages to be displayed to users, data to be validated, and virtually any other type of response imaginable.

```
function greet(){  
    alert("Good morning")  
}
```

Figure 44: Example of JavaScript function definition

Function Expression

The Function expression in JavaScript allows you to define a function as an expression. The function expression is similar to the anonymous function declaration. The function expression can be assigned to a variable. The syntax of function expression in JavaScript is as follows

```
const varName = function (parameter-list) {  
    statements  
};
```

Figure 45: Defining function as Expression in JavaScript

In the example below, we have defined a JavaScript function using function expression and assigned it to a variable name myFunc.

```
const myFunc = function (x, y){  
    return x + y;  
};
```

Figure 46: JavaScript Function Declared as an Expression

Calling a Function

To invoke a function somewhere later in the script, you would simply need to write the name of that function with the parentheses () as shown in the following code. The below code shows the button in the output. When you click the button, it will execute the sayHello() function. The sayHello() function prints the "Hello there!" message in the output.

```
<html>  
<head>  
    <script type="text/javascript">  
        function sayHello() {  
            alert("Hello there!");  
        }  
    </script>  
</head>  
<body>  
    <p>Click the following button to call the function</p>  
    <form>  
        <input type="button" onclick="sayHello()" value="Say Hello">  
    </form>  
    <p> Use different text in the write method and then try... </p>  
</body>  
</html>
```

Figure 47: Declare and Invoke Function in JavaScript

Function Parameters

Till now, we have seen functions without parameters. But there is a facility to pass different parameters while calling a function. These passed parameters can be captured inside the function and any manipulation can be done over those parameters. A function can take multiple parameters

separated by comma. Try the following example. We have modified our sayHello function here. Now it takes two parameters.

```
<html>
  <head>
    <script type = "text/javascript">
      function sayHello(name, age) {
        document.write (name + " is " + age + " years old.");
      }
    </script>
  </head>
  <body>
    <p>Click the following button to call the function</p>
    <form>
      <input type = "button" onclick = "sayHello('Zara', 7)"
        |value = "Say Hello">
    </form>
    <p>Use different parameters inside the function and then try...</p>
  </body>
</html>
```

Figure 48: Function and Parameter in Javascript

The return Statement

A JavaScript function can have an optional **return** statement. This is required if you want to return a value from a function. This statement should be the last statement in a function. For example, you can pass two numbers in a function, and then you can expect the function to return their multiplication in your calling program. The code below defines a function that concatenates two parameters before returning the resultant in the calling program. Also, you may take a look that how it returns the value using the return statement.

```

<html>
<head>
  <script type="text/javascript">
    function concatenate(first, last) {
      var full;
      full = first + last;
      return full;
    }
    function secondFunction() {
      var result;
      result = concatenate('Zara ', 'Ali');
      alert(result);
    }
  </script>
</head>
<body>
  <p>Click the following button to call the function</p>
  <form>
    <input type="button" onclick="secondFunction()" value="Call Function">
  </form>
  <p>Use different parameters inside the function and then try...</p>
</body>
</html>

```

Figure 49: Return Value in a Function

JavaScript Variable Scope

The scope of a variable is the region of your program in which it is defined. JavaScript variables have only two scopes.

- i. Global Variables – A global variable has global scope which means it can be defined anywhere in your JavaScript code.
- ii. Local Variables – A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

Within the body of a function, a local variable takes precedence over a global variable with the same name. If you declare a local variable or function parameter with the same name as a global variable, you effectively hide the global variable. Take a look into the following example. In the example below, we have defined the variable named `myVar` outside the function and initialized it with the 'global' value. Also, we have defined the variable with the same identifier inside the `checkscope()` function and initialized it with the 'local' value. We print the `myVar` variable's value inside the function. So, the local variable takes preference over the global variable and prints the 'local' in the output.

```
<html>
<head>
  <title> JavaScript Variable Scope Example</title>
</head>
<body onload = checkscope();>
  <script>
    var myVar = "global";      // Declare a global variable
    function checkscope( ) {
      var myVar = "local";    // Declare a local variable
      document.write(myVar);
    }
  </script>
</body>
</html>
```

Figure 50: JavaScript Variable Scope

This produces the following result

Output:

local

In the example below, we have defined the variables without using the `var` keyword. The name variable contains the value of the string type, and the number variable contains the value of the

float data type. When we define the variables without using any keyword, JavaScript considers them global variables and can use them anywhere inside the code.

3.3 The DOM events

The DOM events are actions that can be performed on HTML elements. When an event occurs, it triggers a JavaScript function. This function can then be used to change the HTML element or perform other actions. Here are some examples of DOM events:

- i. Click – This event occurs when a user clicks on an HTML element.
- ii. Load – This event occurs when an HTML element is loaded.
- iii. Change – This event occurs when the value of an HTML element is changed.
- iv. Submit – This event occurs when an HTML form is submitted.

You can use the event handlers or `addEventListener()` method to listen to and react to the DOM events. The `addEventListener()` method takes two arguments: the name of the event and the function that you want to be called when the event occurs.

The DOM events are also referred as Document Object Model events. It is used to interact with the DOM elements and manipulate the DOM elements from JavaScript when any event occurs. Let's look at the below examples of DOM events.

The onclick Event Type

This is the most frequently used event type which occurs when a user clicks the left button of his mouse. You can put your validation, warning etc., against this event type.

```

<html>
<head> |
  <script>
    function sayHello() {
      alert("Hello World")
    }
  </script>
</head>
<body>
  <p>Click the following button and see result</p>
  <form>
    <input type = "button" onclick = "sayHello()" value = "Say Hello" />
  </form>
</body>
</html>

```

Figure 51: Onclick Event

The ondblclick Event Type

We use the 'ondblclick' event handler in the code below with the element. When users double click the button, it calls the changeColor() function. In the changeColor() function, we change the color of the text. So, the code will change the text's color when the user double-clicks the button.

```

<html>
<body>
  <h2 id = "text"> Hi Users! </h2>
  <button ondblclick="changeColor()"> Double click me! </button>
  <script>
    function changeColor() {
      document.getElementById("text").style.color = "red";
    }
  </script>
</body>
</html>

```

Figure 52: ondbclick Event

The onkeydown Event Type

We used the 'keydown' event in the code below with the <input> element. Whenever the user will press any key, it will call the customizeInput() function. In the customizeInput() function, we change the background color of the input and the input text to red.

```
<html>
<body>
  <p> Enter charater/s by pressing any key  </p>
  <input type = "text" onkeydown = "customizeInput()">
  <script>
    function customizeInput() {
      var ele = document.getElementsByTagName("INPUT")[0];
      ele.style.backgroundColor = "yellow";
      ele.style.color = "red";
    }
  </script>
</body>
```

Figure 53 : onkeydown Event

The onmouseenter and onmouseleave Events In the code below, we use the 'onmouseenter' and 'onmouseleave' event handlers to add a hover effect on the <div> element. When the mouse pointer enters the <div> element, it calls the changeRed() function to change the text color to red, and when the mouse pointer leaves the <div> element, it calls the changeBlack() function to change the text color to black again.

```

<html>
<body>
  <div id = "text" style = "font-size: 20px;" onmouseenter = "changeRed()"
  |onmouseleave = "changeBlack()"> Hover over the text. </div>
  <script>
    function changeRed() {
      document.getElementById("text").style.color = "red";
    }
    function changeBlack() {
      document.getElementById("text").style.color = "black";
    }
  </script>
</body>
</html>

```

Figure 54: onmouseenter Event

3.4 HTML 5 Standard DOM Events

The standard HTML 5 events are listed here for your reference. Here script indicates a Javascript function to be executed against that event.

Table 3: HTML 5 Standard DOM Events

Attribute	Description
Offline	Triggers when the document goes offline
Onabort	Triggers on an abort event
Onafterprint	Triggers after the document is printed
Onbeforeonload	Triggers before the document loads
Onbeforeprint	Triggers before the document is printed
Onblur	Triggers when the window loses focus

Oncanplay	Triggers when media can start play, but might has to stop for buffering
Oncanplaythrough	Triggers when media can be played to the end, without stopping for buffering
Onchange	Triggers when an element change
Onclick	Triggers on a mouse click

3.5 Operators

An operator is a mathematical symbol which produces a result based on two values (or variables). In the following table you can see some of the simplest operators, along with some examples to try out in the JavaScript console.

Table 4: Operators in JavaScript

Operator	Explanation	Symbol(s)	Example
add/concatenation	Used to add two numbers together, or glue two strings together.	+	6 + 9; "Hello " + "world!";
subtract, multiply, divide	These do what you'd expect them to do in basic math.	-, *, /	9 - 3; 8 * 2; // multiply in JS is an asterisk 9 / 3;
assignment operator	You've seen this already: it assigns a value to a variable.	=	var myVariable = 'Bob';
Identity operator	Does a test to see if two values are equal to one another, and returns a true/false (Boolean) result.	===	var myVariable = 3; myVariable === 4;
Negation, not equal	Returns the logically opposite value of what it precedes; it turns a true into a false, etc. When it is	!, !==	The basic expression is true, but the comparison

	used alongside the Equality operator, the negation operator tests whether two values are <i>not</i> equal.		returns false because we've negated it: <pre>var myVariable = 3; !(myVariable === 3);</pre> Here we are testing "is myVariable NOT equal to 3". This returns false because myVariable IS equal to 3. <pre>var myVariable = 3; myVariable !== 3;</pre>
--	--	--	---

4.0. Self-Assessment Questions

1. In JavaScript, what is a block of statement?

- (a) Conditional block
- (b) block that combines a number of statements into a single compound statement
- (c) both conditional block and a single statement

2. When interpreter encounters an empty statement, what will it do:

- (a) Shows a warning
- (b) Prompts to complete the statement
- (c) Ignores the statements

3. The "function" and " var" are known as:

- a) Keywords
- b) Data types
- c) Declaration statements

4. Define a variable and assign a number to each of the variables.

5. Use any three binary Operators on the two variable and create another variable to store the result

6. Which of these is an example of function in JavaScript.

- (a) function myFunction(a, b) {

```

        return a * b;
    }
(b) function myFunction(a, b) :
        return a * b;
(c) function myFunction(a, b) (
        return a * b;
    )

```

7. Write a javascript program to add personalized message to a web page.
8. Create a function to compute the average of two numbers.
9. Create a five function to trigger five DOM events on a html page.

5.0. Tutor Marked Assignments

1. Write a JavaScript program to compute the average of 5 numbers.
2. Upgrade the program in 1 to compute the average of N numbers.

Unit 4: Server-Side Programming

1.0.Introduction

Server-side programming might sound a bit technical, but it's actually a core part of how websites and web applications work. If you've ever filled out a form online, logged into a website, or made a purchase over the internet, you've interacted with server-side programming. In this introduction, we'll explore what server-side programming is all about, why it's important, and how you can start learning it. At its simplest, server-side programming is the code that runs on a web server, rather than on your personal computer (the client). When you visit a website, your browser sends a request to the server. The server processes this request, often involving database interactions or other server-side logic, and then sends back a response, usually in the form of HTML, CSS, and JavaScript, which your browser renders into the web page you see.

Server-side programming is a powerful skill that opens up a world of possibilities in web development. It might seem daunting at first, but by starting small and gradually building up your knowledge, you'll be able to create dynamic and interactive web applications. Remember to take advantage of the wealth of resources available online, and don't hesitate to reach out to the developer community for support.

This unit introduces students to server-side programming. It also entails Python programming and web development with Django framework.

2.0. Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss server-side programming.
- ii. Install python, setup virtual environment, and install Django.
- iii. Write 5 simple python output statement.
- iv. Write 4 conditional statement.
- v. Define 4 data structure and loop through.
- vi. Students should be able to develop web pages with Django.

3.0 Server Side Programming

Server-side programming is a fundamental aspect of modern web development, focusing on activities that occur on the server rather than on the client's device. It involves writing code that executes on the server, and managing the logic and data that power dynamic web applications. This server-side code interacts with databases, processes user input, and generates responses that are sent back to the client's browser. At the heart of server-side programming is the ability to create dynamic and interactive web pages. Unlike static content, which remains the same for every user, dynamic content can change based on user interactions, preferences, or other variables. This is essential for functionalities such as user authentication, personalized content, and real-time data updates. Several programming languages are commonly used for server-side development, each with its own ecosystem of frameworks and tools. Languages such as PHP, Python, Ruby, Java, and JavaScript (through Node.js) are popular choices. For instance, PHP is widely used with content management systems like WordPress, Python with frameworks like Django and Flask, and JavaScript with Node.js and Express.js. These languages enable developers to write scripts that handle complex operations on the server efficiently.

Security is a paramount concern in server-side programming. Servers often handle sensitive information, including personal data and financial details. Developers must implement robust security measures to protect against threats such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). This involves validating and sanitizing user input, using prepared statements for database queries, and properly managing user sessions. Performance

optimization is also essential in server-side programming. Efficient code and resource management ensure that applications can handle high volumes of traffic without significant delays. Techniques such as caching frequently accessed data, optimizing database queries, and using asynchronous processing can enhance performance and scalability.

Server-side programming works in tandem with client-side programming to deliver full-featured web applications. While server-side code handles data processing and business logic, client-side scripts (typically written in JavaScript) manage user interface interactions within the browser. This division of labour allows for efficient distribution of tasks between the server and the client. In the context of modern web applications, server-side programming is also integral to working with APIs (Application Programming Interfaces). Servers expose endpoints that clients can interact with to perform various operations. This is the foundation of RESTful web services, which allow different software systems to communicate over the web.

Server-side programming is a critical discipline that underpins the functionality of dynamic web applications. It encompasses a range of activities including handling user requests, processing data, interfacing with databases, ensuring security, and optimizing performance. Mastery of server-side programming is essential for developers aiming to build robust, secure, and efficient web applications.

3.1 Web Programming with Python

To begin programming in Python, it is essential to install the latest version of Python on your computer.

1. Download Python:

Visit the official Python website at <https://www.python.org/downloads/>.

Choose the appropriate installer for your operating system (Windows, macOS, or Linux).

2. Install Python:

Run the downloaded installer.

Follow the installation prompts.

On Windows, ensure that you check the option "Add Python to PATH" to run Python from the command line.

3. Verify the Installation:

Open the command prompt (Windows) or terminal (macOS/Linux).

Type ``python --version`` or ``python3 --version``.

The installed Python version should be displayed.

Writing Your First Python Program

With Python installed, you can write and execute your first Python script.

1. Open your visual code editor:

2. Write the Code:

```
print ("Hello, World!")
```

3. Save the File:

Save the file with a `.py`` extension, e.g., ``hello_world.py``.

4. Run the Program:

Navigate to the file's directory using the command line.

Execute the script by typing ``python hello_world.py`` or ``python3 hello_world.py`` as showed below



```
C:\Users\Your Name>python helloworld.py
```

Figure 55: Run Python Script

You should see the output: ``Hello, World!``

Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")  
Hello, World!
```

Figure 56: Execute Python Directly from the Command Line

Python Indentation

Indentation refers to the spaces at the beginning of a code line. Whereas in other programming languages the indentation in code is for readability only, the indentation in Python is very important. Python uses indentation to indicate a block of code.

```
if 5 > 2 :  
    print(" Five is greater than two")  
if 5 > 2 :  
    print("Five is greater than two")
```

Figure 57: Python Indentation

The number of spaces is up to you as a programmer, the most common use is four, but it has to be at least one. You have to use the same number of spaces in the same block of code, otherwise, Python will give you an error:

Comments

Python has commenting capability for the purpose of in-code documentation. Comments start with a #, and Python will render the rest of the line as a comment:

```
#This is a comment.  
print("Hello, World!")
```

Figure 58: Python Comment

Variables

Variables are containers for storing data values.

Creating Variables

Python has no command for declaring a variable. A variable is created the moment you first assign a value to it.

```
x = 5
y = "John"
print(x)
print(y)
```

Figure 59: Python Variable

Variables do not need to be declared with any particular *type*, and can even change type after they have been set.

Casting

If you want to specify the data type of a variable, this can be done with casting.

```
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
```

Figure 60: Type Casting

Get the Type

You can get the data type of a variable with the `type()` function.

```
x = 5
y = "John"
print(type(x))
print(type(y))
```

Figure 61: Type Function

Single and Double Quotes

String variables can be declared either by using single or double quotes:

```
x = "John"
# is the same as
x = 'John'
```

Figure 62: Single and Double Quote for String Variable Declaration

Case-Sensitive

Variable names are case-sensitive.

```
a = 4
A = "Sally"
#A will not overwrite a
```

Figure 63: Case-Sensitive

Variable Names

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume). Rules for Python variables: A variable name must start with a letter or the underscore character A variable name cannot start with a number A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _) Variable names are case-sensitive (age, Age and AGE are three different variables) A variable name cannot be any of the Python keywords.

```
myvar = "John"
my_var = "John"
_my_var = "John"
myVar = "John"
MYVAR = "John"
myvar2 = "John"
```

Figure 64: Variable Names

Many Values to Multiple Variables

Python allows you to assign values to multiple variables in one line:

```
x, y, z = "Orange", "Banana", "Cherry"
print(x)
print(y)
print(z)
```

Figure 65: Assigning Multiple Values to Variables

One Value to Multiple Variables

And you can assign the same value to multiple variables in one line:

```
x = y = z = "Orange"
print(x)
print(y)
print(z)
```

Figure 66: Assigning One Value Multiple Variable

Unpack a Collection

If you have a collection of values in a list, tuple etc. Python allows you to extract the values into variables. This is called unpacking.

```
fruits = ["apple", "banana", "cherry"]
x, y, z = fruits
print(x)
print(y)
print(z)
```

Figure 67: Unpacking a Collection

Output Variables

The Python print() function is often used to output variables. You can also use the + operator to output multiple variables. Several variables can be output using comma

```
x = "Python is awesome"
print(x)

#Seperate Variables by Comma
x = "Python"
y = "is"
z = "awesome"
print(x, y, z)

#Add Variables
x = "Python "
y = "is "
z = "awesome"
print(x + y + z)
```

Figure 68: Output Variable

3.2 Python Data Structure

Data Structures are a way of organizing data so that it can be accessed more efficiently depending upon the situation. In this section you will learn about python data structure such as array , list , tuple , set ,dictionary.

List

List items are ordered, changeable, and allow duplicate values. List items are indexed, the first item has index [0], the second item has index [1] etc. Lists are ordered, meaning that the items have a defined order, and that order will not change. If you add new items to a list, the new items will be placed at the end of the list. List can be modified, meaning that you change, add, and remove items in a list after it has been created. List items can be of any data type


```

# Create a List
mylist = ["apple", "banana", "cherry"]
print(mylist)

# Duplicate values in list:
thislist = ["apple", "banana", "cherry", "apple", "cherry"]
print(thislist)

#To determine how many items a list has, use the len() function:
thislist = ["apple", "banana", "cherry"]
print(len(thislist))

#List items can be of any data type:
list1 = ["apple", "banana", "cherry"]
list2 = [1, 5, 7, 9, 3]
list3 = [True, False, False]

#A list can contain different data types:
#A list with strings, integers and boolean values:
list1 = ["abc", 34, True, 40, "male"]

```

Figure 69: How to Create and Use a List

Access Items

List items are indexed and you can access them by referring to the index number. Negative indexing means starting from the end -1 refers to the last item, -2 refers to the second last item etc. You can specify a range of indexes by specifying where to start and where to end the range. When specifying a range, the return value will be a new list with the specified items. By leaving out the start value, the range will start at the first item: By leaving out the end value, the range will go on to the end of the list: Specify negative indexes if you want to start the search from the end of the list: To determine if a specified item is present in a list use the `in` keyword:

```

#Print the second item of the list:
thislist = ["apple", "banana", "cherry"]
print(thislist[1])

#Print the last item of the list:
thislist = ["apple", "banana", "cherry"]
print(thislist[-1])

#Return the third, fourth, and fifth item:
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:5])

#This example returns the items from the beginning to, but NOT including, "kiwi"
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[:4])

#This example returns the items from "cherry" to the end:

thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[2:])

#This example returns the items from "orange" (-4) to, but NOT including "mango" (-1):
thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
print(thislist[-4:-1])

#Check if "apple" is present in the list:
thislist = ["apple", "banana", "cherry"]
if "apple" in thislist:
    print("Yes, 'apple' is in the fruits list")

```

Figure 70: Accessing Items in a List

Loop Through a List

You can loop through the list items by using a for loop. You can also loop through the list items by referring to their index number. Use the `range()` and `len()` functions to create a suitable iterable. You can loop through the list of items by using a while loop. Use the `len()` function to determine the length of the list, then start at 0 and loop your way through the list items by referring to their indexes. Remember to increase the index by 1 after each iteration. You can loop through the list of items by using a while loop. Use the `len()` function to determine the length of the list, then start at 0 and loop your way through the list items by referring to their indexes. Remember to increase the index by 1 after each iteration. List Comprehension offers the shortest syntax for looping through lists:

```

#Print all items in the list, one by one
thislist = ["apple", "banana", "cherry"]
for x in thislist:
    print(x)

#Print all items by referring to their index number
thislist = ["apple", "banana", "cherry"]
for i in range(len(thislist)):
    print(thislist[i])

#Print all items, using a while loop to go through all the index numbers
thislist = ["apple", "banana", "cherry"]
i = 0
while i < len(thislist):
    print(thislist[i])
    i = i + 1

#A short hand for loop that will print all items in a list (List Comprehension )
thislist = ["apple", "banana", "cherry"]
[print(x) for x in thislist]

```

Figure 71: Loop through a List

Tuple

Tuples are used to store multiple items in a single variable. Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage. A tuple is a collection which is ordered and unchangeable. Tuples are written with round brackets. Tuple items are ordered, unchangeable, and allow duplicate values. Tuple items are indexed, the first item has an index [0], the second item has an index [1] etc. To determine how many items a tuple has, use the len() function. To create a tuple with only one item, you have to add a comma after the item, otherwise Python will not recognize it as a tuple. Tuple can contain different data types.

```

#Create a tuple
thistuple = ("apple", "banana", "cherry")
print(thistuple)

#Tuple allow duplicate values
thistuple = ("apple", "banana", "cherry", "apple", "cherry")
print(thistuple)

#Print number of items in a tuple
thistuple = ("apple", "banana", "cherry")
print(len(thistuple))

#Tuple with one item
thistuple = ("apple",)
print(type(thistuple))

#NOT a tuple
thistuple = ("apple")
print(type(thistuple))

#Tuple can contain different data type
tuple1 = ("abc", 34, True, 40, "male")

```

Figure 72: Declaring a Tuple

Access Tuple Items

You can access tuple items by referring to the index number, inside square brackets. Negative indexing means starting from the end. -1 refers to the last item, -2 refers to the second last item etc. You can specify a range of indexes by specifying where to start and where to end the range. When specifying a range, the return value will be a new tuple with the specified items. By leaving out the end value, the range will go on to the end of the tuple:

```

#Print the second item in the tuple
thistuple = ("apple", "banana", "cherry")
print(thistuple[1])

#negative index
thistuple = ("apple", "banana", "cherry")
print(thistuple[-1])

#Return the third, fourth, and fifth item:
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[2:5])

#This example returns the items from the beginning to, but NOT included, "kiwi":
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[:4])

#This example returns the items from "cherry" and to the end:
thistuple = ("apple", "banana", "cherry", "orange", "kiwi", "melon", "mango")
print(thistuple[2:])

```

Figure 73: Accessing Items in a Tuple

Change Tuple Values

Once a tuple is created, you cannot change its values. Tuples are **unchangeable**, or **immutable** as it also is called. But there is a workaround. You can convert the tuple into a list, change the list, and convert the list back into a tuple.

Add Items

Since tuples are immutable, they do not have a built-in `append()` method, but there are other ways to add items to a tuple.

1. **Convert into a list:** Just like the workaround for *changing* a tuple, you can convert it into a list, add your item(s), and convert it back into a tuple.

```

#Convert the tuple into a list to be able to change it:
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)

print(x)

#Convert the tuple into a list, add "orange", and convert it back into a tuple:
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)

#Create a new tuple with the value "orange", and add that tuple:
thistuple = ("apple", "banana", "cherry")
y = ("orange",)
thistuple += y

print(thistuple)

```

Figure 74: Change Value of a Tuple

Loop Through a Tuple

You can loop through the tuple items by using a for loop. Loop Through the Index Numbers. You can also loop through the tuple items by referring to their index number. Use the range() and len() functions to create a suitable iterable.

```

#Convert the tuple into a list to be able to change it:
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)

print(x)
|

#Convert the tuple into a list, add "orange", and convert it back into a tuple:
thistuple = ("apple", "banana", "cherry")
y = list(thistuple)
y.append("orange")
thistuple = tuple(y)

#Create a new tuple with the value "orange", and add that tuple:
thistuple = ("apple", "banana", "cherry")
y = ("orange",)
thistuple += y

print(thistuple)

#Iterate through the items and print the values:
thistuple = ("apple", "banana", "cherry")
for x in thistuple:
|   print(x)

#Print all items by referring to their index number:
thistuple = ("apple", "banana", "cherry")
for i in range(len(thistuple)):
|   print(thistuple[i])

#Print all items, using a while loop to go through all the index numbers:
thistuple = ("apple", "banana", "cherry")
i = 0
while i < len(thistuple):
|   print(thistuple[i])
|   i = i + 1

```

Figure 75: Loop Through a Tuple

Set

Sets are used to store multiple items in a single variable. Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Tuple, and Dictionary, all with different qualities and usage. A set is a collection which is unordered, unchangeable*, and unindexed.

Set Items

Set items are unordered, unchangeable, and do not allow duplicate values. Unordered means that the items in a set do not have a defined order. Set items can appear in a different order every time you use them, and cannot be referred to by index or key. Set items are unchangeable, meaning that we cannot change the items after the set has been created.

```
#Duplicate value will be ignored
thisset = {"apple", "banana", "cherry", "apple"}

print(thisset)

#True and 1 is considered the same value:
thisset = {"apple", "banana", "cherry", True, 1, 2}

print(thisset)

#False and 0 are considered the same value
thisset = {"apple", "banana", "cherry", False, True, 0}

print(thisset)
```

Figure 76: Creating a set

Access Items

You cannot access items in a set by referring to an index or a key. But you can loop through the set items using a for loop, or ask if a specified value is present in a set, by using the in keyword. Loop through the set, and print the values.


```
#Loop through the set, and print the values
thisset = {"apple", "banana", "cherry"}

for x in thisset:
    print(x)

#Check if "banana" is present in the set:
thisset = {"apple", "banana", "cherry"}

print("banana" in thisset)

#Check if "banana" is NOT present in the set:
thisset = {"apple", "banana", "cherry"}

print("banana" not in thisset)
```

Figure 76: Loop through Set

Dictionary

Dictionaries are used to store data values in key: value pairs. A dictionary is a collection which is ordered*, changeable and does not allow duplicates. Dictionaries are written with curly brackets, and have keys and values: Dictionary items are ordered, changeable, and do not allow duplicates. Dictionary items are presented in key:value pairs, and can be referred to by using the key name. Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created. Dictionaries cannot have two items with the same key: The values in dictionary items can be of any data type.

```

#create a dictionary
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}

print(thisdict)

# Print the "brand" value of the dictionary:
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
print(thisdict["brand"])

#Duplicate values will overwrite existing values:
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964,
    "year": 2020
}
print(thisdict)

```

Figure 77: Creating a Dictionary

Accessing Items

You can access the items of a dictionary by referring to its key name, inside square brackets. There is also a method called `get()` that will give you the same result. The `keys()` method will return a list of all the keys in the dictionary. The list of the keys is a *view* of the dictionary, meaning that any changes done to the dictionary will be reflected in the keys list. The `values()` method will return a list of all the values in the dictionary. The list of the values is a *view* of the dictionary, meaning that any changes done to the dictionary will be reflected in the values list. The `items()` method will return each item in a dictionary, as tuples in a list.

```

#Get the value of the "model" key:
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = thisdict["model"]

#Get the value of the "model" key:
x = thisdict.get("model")

#Get a list of the keys:
x = thisdict.keys()

#Add a new item to the original dictionary, and see that the keys list gets updated as well:
car = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}

x = car.keys()

print(x) #before the change

car["color"] = "white"

print(x) #after the change

```

Figure 78: Accessing Items in a Dictionary

Loop Through a Dictionary

You can loop through a dictionary by using a for loop. When looping through a dictionary, the return values are the *keys* of the dictionary, but there are methods to return the *values* as well.

```
#Print all key names in the dictionary, one by one:
for x in thisdict:
    print(x)

#Print all values in the dictionary, one by one:
for x in thisdict:
    print(thisdict[x])

#You can also use the values() method to return values of a dictionary:
for x in thisdict.values():
    print(x)

#You can use the keys() method to return the keys of a dictionary:
for x in thisdict.keys():
    print(x)

#Loop through both keys and values, by using the items() method:
for x, y in thisdict.items():
    print(x, y)
```

Figure 79: Loop through a Dictionary

3.3 Function

A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result.

Creating a Function

In Python a function is defined using the def keyword:

Calling a Function

To call a function, use the function name followed by parenthesis.

```
def my_function():  
    print("Hello from a function")  
  
# invoke a function  
  
my_function()
```

Figure 80: Calling a Function

Arguments

Information can be passed into functions as arguments. Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma. The following example has a function with one argument (fname). When the function is called, we pass along a first name, which is used inside the function to print the full name:

```
def my_function(fname):  
    print("Your name is : "+fname)  
  
my_function("Tunde")  
my_function("Adam")  
my_function("Ruth")
```

Figure 81: Passing Argument into a Function

Number of Arguments

By default, a function must be called with the correct number of arguments. Meaning that if your function expects 2 arguments, you have to call the function with 2 arguments, not more, and not less. If you do not know how many arguments will be passed into your function, add a * before the parameter name in the function definition. This way the function will receive a *tuple* of arguments and can access the items accordingly:

```
# Funtion with two argument
def my_function(fname, lname):
    print(fname + " " + lname)

my_function("Adam", "Solomon")

#Function with arbitrary argument
def my_function(*kids):
    print("The youngest child is " + kids[2])

my_function("John", "Peter", "George")
```

Figure 82: Function with Arbitrary Function

Global Variables

Variables that are created outside of a function (as in all of the examples in the previous pages) are known as global variables. Global variables can be used by everyone, both inside of functions and outside. If you create a variable with the same name inside a function, this variable will be local, and can only be used inside the function. The global variable with the same name will remain as it was, global and with the original value. Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function. To create a global variable inside a function, you can use the global keyword.

```

# Create a variable outside of a function, and use it inside the function
x = "awesome"
def myfunc():
    print("Python is " + x)

myfunc()

#Create a variable inside a function, with the same name as the global variable
x = "awesome"
def myfunc():
    x = "fantastic"
    print("Python is " + x)

myfunc()

print("Python is " + x)

#If you use the global keyword, the variable belongs to the global scope:

def myfunc():
    global x
    x = "fantastic"

myfunc()

print("Python is " + x)

#To change the value of a global variable inside a function, refer to the variable by using the global keyword:
x = "awesome"

def myfunc():
    global x
    x = "fantastic"

myfunc()

print("Python is " + x)

```

Figure 83: Variable Scope

Operators In python

Operators are special symbols or keywords that perform operations on one or more operands (variables or values). Understanding these operators is crucial for effective programming in Python. Table 5,6, and 7 presents important python operators.

Table 5: Arithmetic Operators in Python

S/N	Arithmetic Operator	Symbol	Example
1	Addition	+	x + y

2	Subtraction	-	x - y
3	Multiplication	*	x * y
4	Division	/	x/y
5	Modulus	%	x % y
6	Exponential	**	x ** y
7	Floor Division	//	x // y

Table 6: Comparison Operator in Python

S/N	Comparison Operator	Symbol	Example
1	Equal	==	x= y
2	Not Equal	!=	x !=y
3	Greater than	>	x > y
4	Less than	<	x < y
5	Greater than or Equal to	>=	x >= y
6	Less than or Equal to	<=	x <= y

Table 7: Logical Operators

S/N	Comparison Operator	Symbol	
1	Logical AND	And	x > 0 and y < 15
2	Logical OR	Or	x > 0 or y < 15

3	Logical NOT	Not	<pre>is_raining = True if not is_raining: print("It's not raining!")</pre>
---	-------------	-----	--

Conditional Statement

Conditional Statements are statements in python that provide a choice for the control flow based on a condition. It means that the control flow of the python program will be decided based on the outcome of the condition. Syntax and example are presented in Table 8.

Table 8: Conditional Statement in Python

S/N	Conditional Statements	Syntax	Example
	If Statements	<pre>If Statements: if condition: # Execute this block if condition is true</pre>	<pre>age = 18 if age >= 18: print("You are an adult.")</pre>
	If...Else Statements:	<pre>if condition: # Execute if true else: # Execute if false</pre>	<pre>score = 75 if score >= 60: print("Pass") else: print("Fail")</pre>
	Elif Statements	<pre>if condition1: # Execute if condition1 is true elif condition2:</pre>	<pre>grade = 85 if grade >= 90: print("A") elif grade >= 80:</pre>

		<pre># Execute if condition2 is true else: # Execute if none of the above are true</pre>	<pre>print("B") elif grade >= 70: print("C") else: print("D")</pre>
	For Loops	<pre>for variable in iterable: # Code block to execute</pre>	<pre>for i in range(5): print(i) # Output: 0 1 2 3 4 While Loops: while condition: # Code block to execute</pre>
	Try...Except Blocks:	<pre>Try...Except Blocks: try: # Code that may cause an error except ExceptionType: # Code to execute if an error occurs</pre>	<pre>try: result = 10 / 0 except ZeroDivisionError: print("Cannot divide by zero.")</pre>

3.2. Function

Apart from the standard Python function, user can define their own function.

Syntax

```
def function_name(parameters):
```

```
# Code block  
return result
```

Calling a Function

```
function_name(arguments)
```

Example

```
def add_numbers(a, b):  
    return a + b  
  
result = add_numbers(5, 7)  
print(result) # Output: 12
```

3.3. Introduction to Django Python Web Framework

Among the various web frameworks available for Python, Django stands out as a powerful and comprehensive tool that simplifies the development process. This tutorial aims to introduce beginners to Python web development using Django, guiding them through the fundamental concepts and steps required to build a basic web application. Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. Developed by experienced developers, it takes care of much of the hassle of web development, allowing developers to focus on writing their applications without needing to reinvent the wheel. Django follows the Model-View-Template (MVT) architectural pattern, which promotes the separation of concerns and enhances scalability and maintainability.

Model-View-Template Architecture: Separates data representation, user interface, and business logic.

Object-Relational Mapping (ORM): Facilitates interaction with the database without writing SQL queries.

URL Routing: Maps URLs to views using a clear and concise syntax.

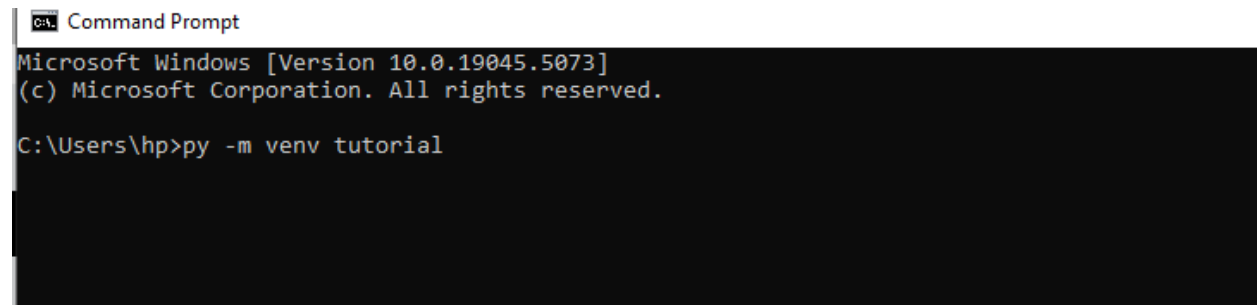
Template System: Uses Django's templating language to create dynamic HTML pages.

Built-in Admin Interface: Provides an automatically generated administrative panel.

Security Features: Includes protection against common attacks like SQL injection and cross-site scripting.

Create Virtual Environment

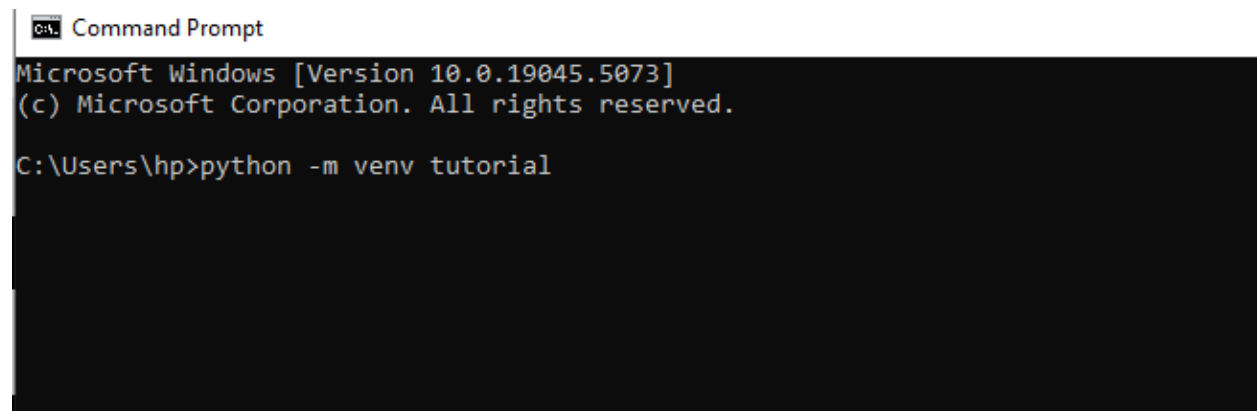
It is suggested to have a dedicated virtual environment for each Django project, and one way to manage a virtual environment is venv, which is included in Python. The name of the virtual environment is your choice, in this example, we will call it tutorial. Type the following in the command prompt, remember to navigate to where you want to create your project:



```
Command Prompt
Microsoft Windows [Version 10.0.19045.5073]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>py -m venv tutorial
```

Figure 84: Create a Virtual Environment on Windows OS



```
Command Prompt
Microsoft Windows [Version 10.0.19045.5073]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>python -m venv tutorial
```

Figure 85: Create a Virtual Environment on MacOS

This will set up a virtual environment, and create a folder named "tutorial" with subfolders and files as shown below. Navigate to the directory you specified on your computer to view the created virtual environment folder.

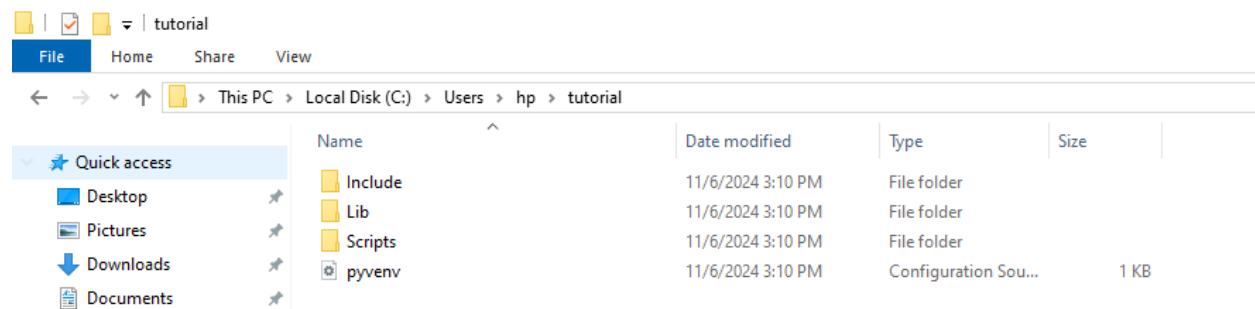


Figure 86: The Virtual environment folder

Then you have to activate the environment, by typing this command: by typing this command "tutorial\scripts\activate.bat" for Windows or tutorial/bin/activate for Linux/MacOS. Once the environment is activated, you will see this result in the command prompt.

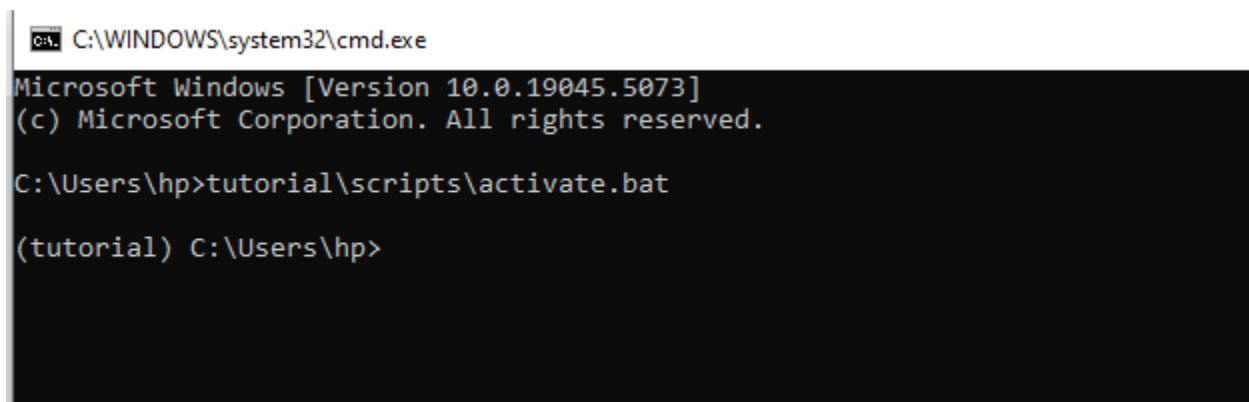


Figure 87: Activate Virtual Environment

Now, that we have created a virtual environment, we are ready to install Django. Using this command for **py -m pip install Django** for Windows OS or **python -m pip install Django** for Linux/MacOS.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.5073]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>py -m pip install Django
```

Figure 88: Django Installation Command

This will give a result like Figure 85.

```
Collecting Django
  Downloading Django-4.0.3-py3-none-any.whl (8.0 MB)
    | 8.0 MB 2.2 MB/s
Collecting sqlparse>=0.2.2
  Using cached sqlparse-0.4.2-py3-none-any.whl (42 kB)
Collecting asgiref<4,>=3.4.1
  Downloading asgiref-3.5.0-py3-none-any.whl (22 kB)
Collecting tzdata; sys_platform == "win32"
  Downloading tzdata-2021.5-py2.py3-none-any.whl (339 kB)
    | 339 kB 6.4 MB/s
Installing collected packages: sqlparse, asgiref, tzdata, Django
Successfully installed Django-4.0.3 asgiref-3.5.0 sqlparse-0.4.2 tzdata-2021.5
WARNING: You are using pip version 20.2.3; however, version 22.3 is available.
You should consider upgrading via the 'C:\Users\Your Name\myworld\Scripts\python.exe -m pip install --upgrade pip' command.
```

Figure 85: Django Installation

Creating a Project on Django

Once you have come up with a suitable name for your Django project, navigate to where in the file system you want to store the code (in the virtual environment), and run this command in the command prompt:

```
(tutorial) C:\Users\hp>django-admin startproject my_first_project
(tutorial) C:\Users\hp>
```

Figure 86: Creating Django Project

Django creates a `my_first_project` folder on your computer, with this content:

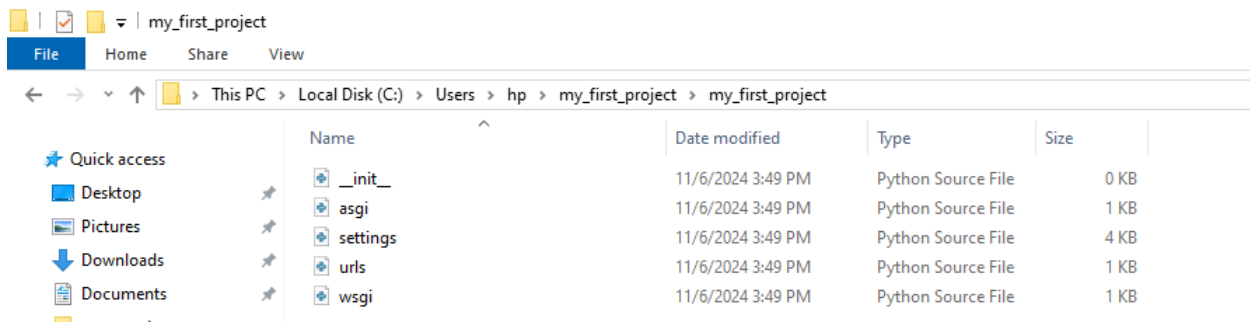


Figure 87: Django Project Folder

These are all files and folders with a specific meaning, you will learn about some of them later in this tutorial, but for now, it is more important to know that this is the location of your project and that you can start building applications in it. Now that you have a Django project, you can run it, and see what it looks like in a browser. Navigate to the `/my_first_project` folder and execute "py manage.py run server" in the command prompt :

```
(tutorial) C:\Users\hp\my_first_project>py manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
November 06, 2024 - 16:02:25
Django version 5.1.3, using settings 'my_first_project.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

[06/Nov/2024 16:06:18] "GET / HTTP/1.1" 200 12068
Not Found: /favicon.ico
[06/Nov/2024 16:06:18] "GET /favicon.ico HTTP/1.1" 404 2218
```

Figure 88: Lunch Django on Browser

Type <http://127.0.0.1:8000/> on your browser. This will result in the page displayed in Figure 89.

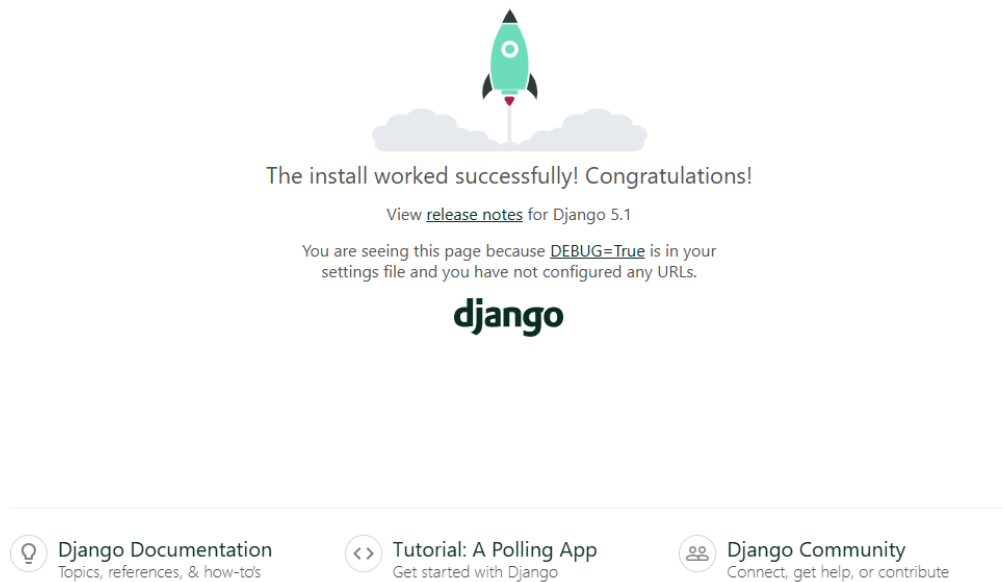


Figure 89: Django Default Home page

Creating an APP

Now that we have a Django project, the next step is to make an app for your project. You cannot have a web page created with Django without an app. I will name my app my_APP. Start by navigating to the my_first_project folder, and run the command in Figure 90. If the server is still running, and you are not able to write commands, press [CTRL] [BREAK], or [CTRL] [C] to stop the server and you should be back in the virtual environment.

```
(tutorial) C:\Users\hp\my_first_project>py manage.py startapp my_APP
(tutorial) C:\Users\hp\my_first_project>
```

Figure 90: Creating an APP

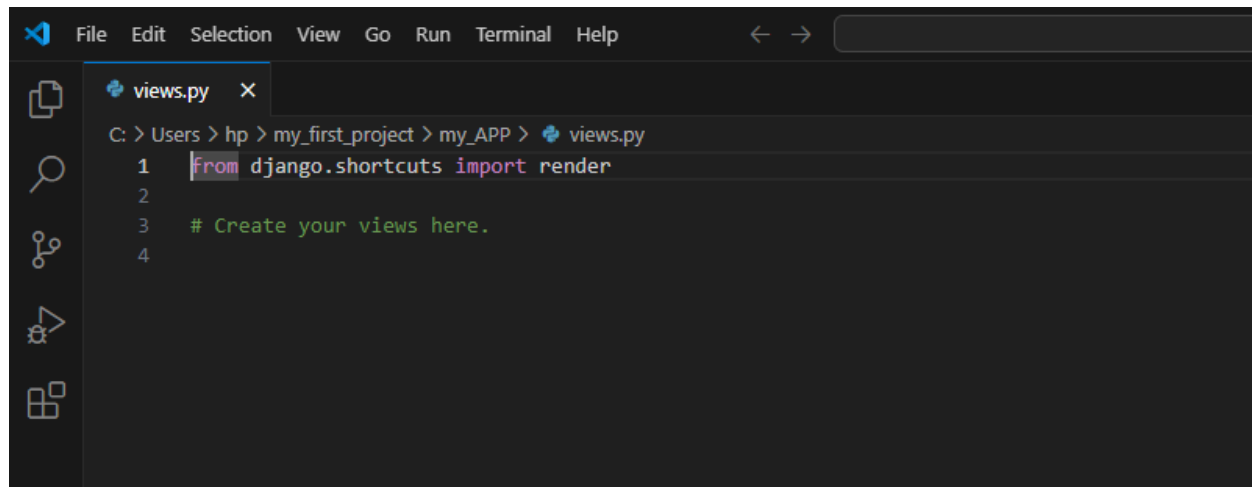
Django creates a new folder named my_APP in the project, with this content:

View				
PC > Local Disk (C:) > Users > hp > my_first_project > my_APP				
Name	Date modified	Type	Size	
migrations	11/6/2024 4:17 PM	File folder		
__init__	11/6/2024 4:17 PM	Python Source File	0 KB	
admin	11/6/2024 4:17 PM	Python Source File	1 KB	
apps	11/6/2024 4:17 PM	Python Source File	1 KB	
models	11/6/2024 4:17 PM	Python Source File	1 KB	
tests	11/6/2024 4:17 PM	Python Source File	1 KB	
views	11/6/2024 4:17 PM	Python Source File	1 KB	

Figure 91: Django APP Folder

Django Views

Django views are Python functions that take HTTP requests and return HTTP responses, like HTML documents. A web page that uses Django is full of views with different tasks and missions. Views are usually put in a file called views.py located in your app's folder. There is a views.py in your my_APP folder that looks like this:



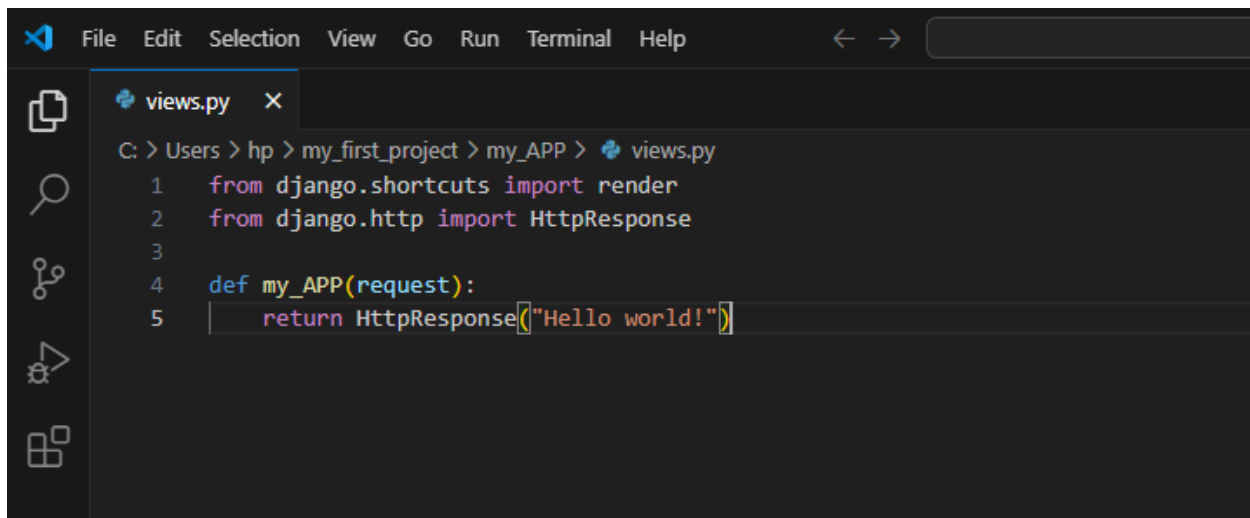
```

File Edit Selection View Go Run Terminal Help
views.py x
C: > Users > hp > my_first_project > my_APP > views.py
1 from django.shortcuts import render
2
3 # Create your views here.
4

```

Figure 92: Django View File

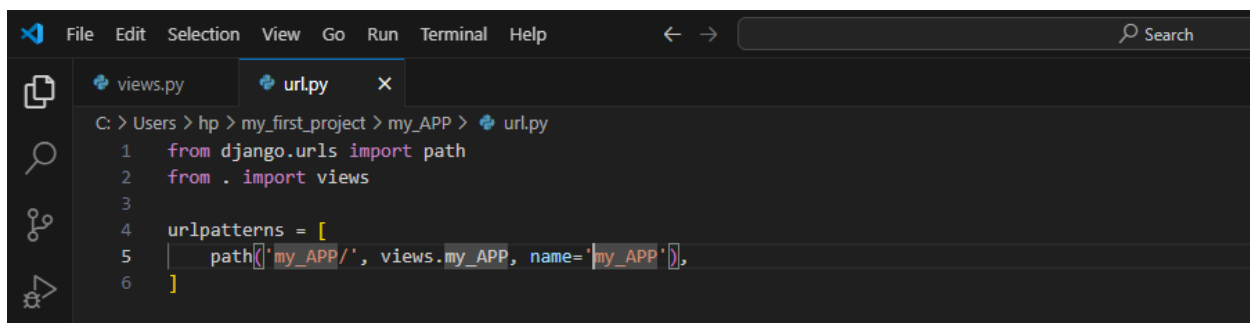
Replace the content with this

A screenshot of the Visual Studio Code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The left sidebar shows icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The main editor window has a tab for 'views.py'. The file path is 'C: > Users > hp > my_first_project > my_APP > views.py'. The code in the file is:

```
1 from django.shortcuts import render
2 from django.http import HttpResponse
3
4 def my_APP(request):
5     return HttpResponse("Hello world!")
```

Figure 93: Output Hello World on the Home page

Create a file named `urls.py` in the same folder as the `views.py` file, and type this code in it.

A screenshot of the Visual Studio Code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The left sidebar shows icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The main editor window has two tabs: 'views.py' and 'url.py'. The file path for 'url.py' is 'C: > Users > hp > my_first_project > my_APP > url.py'. The code in the file is:

```
1 from django.urls import path
2 from . import views
3
4 urlpatterns = [
5     path('my_APP/', views.my_APP, name='my_APP'),
6 ]
```

Figure 94: Create Urls.py File

The `urls.py` file you just created is specific to the `my_APP` application. We have to do some routing in the root directory `my_first_project` as well. There is a file called `urls.py` in the `my_first_project` folder, open that file and add the `include` module in the import statement, and also add a `path()` function in the `urlpatterns[]` list, with arguments that will route users that come in via `127.0.0.1:8000/`. Then your file will look like this.

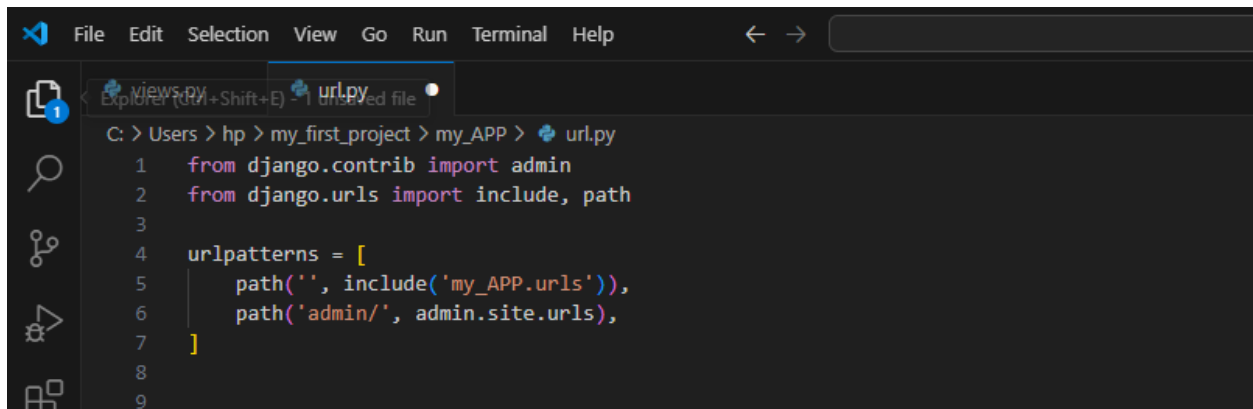


Figure 95: Modify URL File

If the server is not running, navigate to the / my_first_project folder and execute this command in the command prompt: `py manage.py run server`.

Output:

Hello world!

Open the HTML file and insert the following:

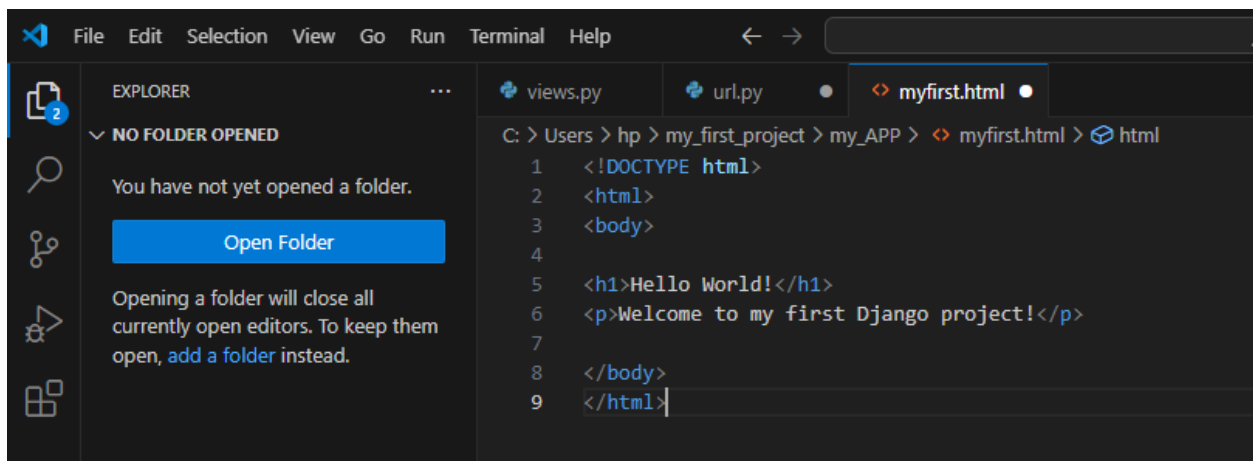


Figure 96: Modifying HTML File for the APP

Modify the View

Open the views.py file and replace the my_APP's view with this.

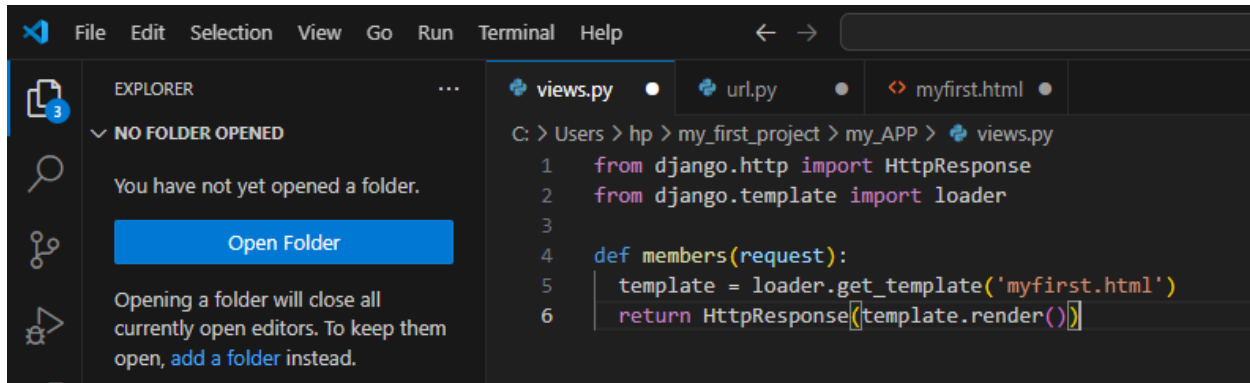


Figure 97: Modifying View For the APP

Change Settings

To be able to work with more complicated stuff than "Hello World!", We have to tell Django that a new app has been created. This is done in the settings.py file in the my_first_project. Look up the INSTALLED_APPS[] list and add the my_APP app like this:

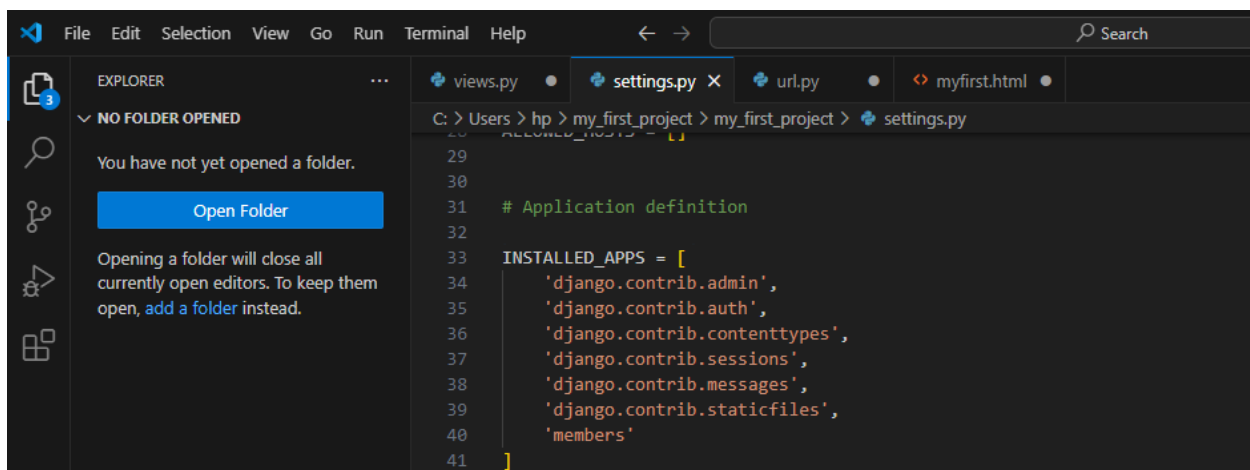


Figure 98: Change Settings

Then run py manage.py migrate which will produce this output

Output:

```
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK
```

Output:

Run py manage.py runserver

Hello World!

Welcome to my first Django project!

4.0. Self-Assessment Questions

1. Which of this is an output statement in Python?

- (i) print ("Hello world")
- (ii) print (10)
- (iii) def (print)
- a) i,ii b) i,iii c)iii

2. Which type of Programming does Python support?

- a) object-oriented programming.
- b) functional programming.
- c) All of the above.

3. Is Python case-sensitive when dealing with identifiers?

- a) no
- b) yes
- c) machine dependent

4. Which of the following is the correct extension of the Python file?

- a). python
- b) .pl
- c) .py

6. Write 5 python output statements.

7. Define 4 data structure in python and loop through it element.

8. Describe server-side programming.

9. Create a new virtual environment and install Django.

10. Using Django HTML and CSS create a website that displays a company's information.

11. Write 4 conditional statements in python using different control flow.

5.0. Tutor Marked Assignments

- 1. Explain what server-side programming is.
- 2. Differentiate between server-side and client-side programming.

Module 4: Intranet and Extranet

- Unit 1:** Intranet
- Unit 2:** Extranet
- Unit 3:** Network Security

Unit 1: Intranet

1.0.Introduction

In today's interconnected world, efficient communication and information sharing are vital for organizational success. As businesses expand and collaborate with various stakeholders, the need for secure and effective networking solutions becomes paramount. Intranets and extranets are two such solutions that organizations employ to enhance internal operations and external collaborations. Have you ever wondered how organizations share information internally and with trusted external partners? Understanding intranets and extranets is key to grasping how businesses communicate and collaborate in today's digital world.

An intranet is a private network that's accessible only to an organization's employees or members. Think of it as an internal website that helps staff communicate, collaborate, and access important resources securely within the company. It's designed to streamline workflows, improve communication, and enhance productivity by providing a centralized platform for information sharing. For example, an employee might use the intranet to find company policies, access training materials, or collaborate with colleagues on projects. Since it's closed off from the public internet, sensitive information remains secure within the organization. For instance, a supplier might use

the extranet to check inventory levels or place orders directly with the company. This connectivity streamlines business processes and fosters stronger relationships with external stakeholders. Intranets and extranets play vital roles in modern organizations by enhancing communication, collaboration, and efficiency. Intranets and extranets are powerful tools that help organizations communicate better, collaborate more effectively, and build stronger relationships both internally and externally. By leveraging these networks, businesses can enhance productivity, reduce costs, and maintain a competitive edge in today's fast-paced digital environment. In this unit, you will learn more about the intranet.

2.0. Learning Outcomes

At the end of this unit ,you should be able to:

- i. Define intranet
- ii. Itemize 5 features of the intranet.
- iii. Discuss 4 benefits of intranet.

3.0. Definition of Intranet

An intranet is a private network that is accessible only to an organization's employees. It utilizes internet protocols (TCP/IP) and network connectivity to facilitate communication, collaboration, and information sharing within the organization. Intranets are designed to streamline internal processes, improve productivity, and foster a unified corporate culture by providing a centralized platform for resources and applications.

Features of Intranets

Intranets serve as an internal hub for organizational activities. Key features include:

Internal Communication Tools: Platforms such as email systems, instant messaging, and internal forums enable efficient communication among employees.

Document Management Systems: Centralized repositories allow for the secure storage, retrieval, and management of documents, policies, and procedures.

Collaborative Workspaces: Team members can work together on projects through shared calendars, task lists, and project management tools.

Employee Directories: Comprehensive databases containing employee contact information facilitate easy connectivity across departments.

Corporate News and Announcements: Updates on company events, announcements, and news are disseminated organization-wide.

By leveraging these features, intranets enhance operational efficiency, reduce redundancies, and support knowledge management initiatives.

3.1. Benefits of Intranets

The implementation of an intranet offers several advantages:

Improved Communication: Streamlines the flow of information, reducing misunderstandings and delays.

Increased Productivity: Easy access to information and tools enhances employee efficiency.

Cost Savings: Reduces the need for physical documents and minimizes printing and distribution costs.

Enhanced Collaboration: Fosters teamwork through shared platforms and resources.

Knowledge Management: Captures and disseminates organizational knowledge effectively.

4.0. Self-Assessment Questions

1. Intranet can be defined as

(a) a private network that is accessible only to an organization's employees. It utilizes internet protocols (TCP/IP) and network connectivity to facilitate communication, collaboration, and information sharing within the organization

(b) a website

(c)a blog

2. The intranet is accessible

- (a) To the public
- (b) To only people within the company
- (c) To the administrator.

3. Key features of the intranet include

(a) Internal Communication Tools

(b) Network

(c) Data

6. The benefits of intranet include

(a) Improved Communication

(b) Increased Productivity

(c) All of the above

4. Which of this combination best describes the features of the intranet?

- i. Internal Communication Tools
- ii. Document Management Systems
- iii. Collaborative Workspaces
- iv. Employee Directories
- v. Corporate News and Announcements
- vi. External link to other organizations

(a) I,ii(b)i,ii,iii,iv,v (c)I,ii,iii,vi

5.0. Tutor Marked Assignments

- i. List and explain the features of the intranet
- ii. How can an intranet increase the productivity of an organization.

Unit 2: Extranet

1.0.Introduction

Have you ever wondered how businesses securely share information with their partners or suppliers over the Internet? That's where extranets come into play. An extranet is like having a private section of the internet that's specially designed for a company and its trusted external collaborators. It allows organizations to extend a portion of their internal network to external users in a secure and controlled manner. An extranet is a private network that leverages Internet technology to share part of a business's information or operations with suppliers, vendors, partners, customers, or other businesses. It's essentially an extension of a company's intranet—the internal network accessible only by the organization's employees—but it's accessible to authorized external users. This setup enables different organizations to work together and share information securely.

Extranets use standard internet protocols, making them accessible through regular web browsers. However, unlike public websites, extranets require users to authenticate themselves, usually with a username and password. Security measures like encryption and firewalls are put in place to protect sensitive information. This ensures that only authorized individuals can access specific data, applications, or services provided by the extranet. Extranets play a vital role in modern business operations by bridging the gap between organizations and their external partners. They provide a secure and efficient way to collaborate, share information, and streamline processes. By leveraging an extranet, companies can enhance their partnerships, improve productivity, and maintain a competitive edge in their industry.

This unit introduces the extranet to the student. It also presents features and functions of extranet benefits of the extranet and a comparison between the intranet and the extranet.

2.0.Learning Outcomes

At the end of this unit students should be able to:

- i. Define Extranet.
- ii. State 4 features of extranet.
- iii. State 4 benefits of extranet.
- iv. State 4 differences between Intranet and Extranet.

3.0. Definition of Extranet

An extranet extends the concept of an intranet by allowing controlled access to external users such as business partners, suppliers, vendors, or clients. It is a private network that facilitates secure communication and collaboration between an organization and its external stakeholders. Extranets use internet protocols but implement stringent security measures to protect sensitive information.

Features of Extranets

Extranets provide a platform for extended enterprise collaboration. Key features include:

Secure Access Controls: Authentication mechanisms ensure that only authorized external users can access the network.

Shared Applications and Data: External stakeholders can access relevant applications, databases, and shared documents necessary for collaboration.

Collaboration Tools: Similar to intranets, extranets offer tools for communication and project management with external parties.

Supply Chain Management: Organizations can share inventory levels, order statuses, and forecasts with suppliers and partners.

Customer Relationship Management (CRM): Clients can access support systems, track orders, and view account information.

By providing these features, extranets streamline external business processes, improve partner relations, and enhance customer service.

3.1. Benefits of Extranets

Extranets offer several significant benefits:

Strengthened Partnerships: Facilitates closer collaboration with partners and suppliers.

Improved Efficiency: Streamlines business processes and reduces transaction times.

Enhanced Customer Engagement: Provides clients with direct access to information and services.

Cost Reduction: Decreases operational costs through automation and reduced manual intervention.

Competitive Advantage: Offers superior service levels and responsiveness to stakeholders.

3.2. Comparison Between Intranet and Extranet

While both intranets and extranets utilize Internet technologies and serve as private networks, they differ primarily in their user accessibility and purpose.

Access Control

Intranet: Access is restricted to internal employees within the organization.

Extranet: Access is extended to authorized external users alongside internal employees.

Purpose

Intranet: Focuses on enhancing internal communication, collaboration, and information sharing.

Extranet: Aims to facilitate collaboration and communication with external stakeholders.

Security Measures

Intranet: Employs standard security protocols to protect internal information.

Extranet: Requires advanced security measures due to external access, including robust authentication, encryption, and firewall configurations.

Network Boundaries

Intranet: Operates within the organization's internal network infrastructure.

Extranet: Extends beyond internal networks to connect with external networks securely.

Security Considerations

Security is a critical aspect of both intranets and extranets. Protecting sensitive information from unauthorized access, breaches, and cyber threats is essential.

Security Measures for Intranets

User Authentication: Ensures that only authorized employees can access the network.

Access Controls: Role-based permissions limit access to sensitive information based on employee roles.

Firewalls: Protect the internal network from external threats.

Regular Updates and Patches: Keeping software up-to-date to mitigate vulnerabilities.

Security Measures for Extranets

Robust Authentication Mechanisms: Multi-factor authentication to verify external users.

Encryption: Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocols encrypt data transmission.

Virtual Private Networks (VPNs): Secure remote connections between external users and the organization's network.

Intrusion Detection Systems (IDS): Monitor network traffic for suspicious activities.

Challenges Associated with Intranets and Extranets

Despite their benefits, intranets and extranets present several challenges:

Implementation Costs: Initial setup and ongoing maintenance can be costly.

Technical Complexity: Requires specialized IT expertise to manage and support the networks.

User Adoption: Employees and external users may resist adopting new systems without proper training.

Security Risks: Potential vulnerabilities can lead to data breaches and unauthorized access. Organizations must address these challenges through strategic planning, investment in technology, training, and continuous monitoring.

Intranets and extranets are integral components of an organization's IT infrastructure, facilitating efficient communication, collaboration, and information sharing. Intranets improve internal operations by connecting employees and centralizing resources. Extranets extend these benefits to external stakeholders, enhancing business relationships and operational efficiency. Understanding the distinctions, benefits, and challenges of intranets and extranets enables organizations to leverage these tools effectively for competitive advantage.

4.0. Self-Assessment Questions

1. An extranet extends the concept of an intranet by allowing controlled_____to external users.

(a)access (b) denial (c) flow

2. Extranets use _____ but implement stringent security measures to protect sensitive information.

(a) internet protocols (b) network (c) rules

3. Which of the following is a feature of extranets?

(a)Secure access controls (b)firewall (c)Antivirus

4. Which of these best describes the features of an extranet?

i. Secure Access Controls

ii. Shared Applications and Data

iii. Collaboration Tools

iv. Supply Chain Management

v. CRM

vi. Website

(a) i,ii,iii,iv,v (b)I,ii,iii,iv,v,vi (c)) i,ii,iv,v

6. State 4 benefits of extranet.

7. State 5 differences between Intranet and Extranet.

5.0. Tutor Marked Assignments

Distinguish between intranet and extranet based on security measures.

Module 5: Exploring the Internet and Network Security

Unit 1 Search Engines

Unit 2 Network Security

Unit 1 Application of Internet

Introduction

In today's digital age, the internet plays a vital role in our daily lives. From catching up on the news to connecting with friends and family, it's hard to imagine a day without it. As of 2023, over 5 billion people worldwide have access to the Internet, underscoring its pervasive influence on daily life. However, this widespread connectivity also introduces significant security challenges that must be addressed to protect users and organizations from cyber threats. As we become more dependent on the internet, understanding internet security becomes increasingly important. Let's take a closer look at internet exploration and how we can protect ourselves while navigating the digital world. This unit presents the application of the internet which includes a discussion on search engines, the role of search engines in internet exploration, Privacy, Ethical Considerations and future trends of search engines.

1. Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss search engine
- ii. Itemize 5 examples of search engines.
- iii. State 5 features of search engines
- iv. Discuss the role of search engines in internet exploration.
- v. Itemize 4 privacy and ethical considerations pertaining to search engines.
- vi. Itemize 4 future trends in search engines.

3.0 Search Engine

A search engine is a software system designed to perform web searches, enabling users to find information on the World Wide Web. The primary functions of a search engine include crawling, indexing, and retrieving relevant results based on user queries. Search engines utilize automated programs known as spiders or crawlers to traverse the web. These crawlers follow links from one page to another, discovering new and updated content to add to the search engine's index.

Common Search Engines

Several dominant search engines facilitate the exploration of the Internet, each with unique features and market shares.

- i. **Google Search:** Launched in 1998 by Larry Page and Sergey Brin, Google is the most widely used search engine globally. It is renowned for its sophisticated algorithms and vast index of web pages.
- ii. **Bing:** Developed by Microsoft, Bing offers interactive search features and integrates with Microsoft's suite of products. It emphasizes visual search capabilities and rewards users through the Microsoft Rewards program.
- iii. **Yahoo! Search:** Once a leading search engine, Yahoo! now sources its search results from Bing but maintains its own user interface and additional services like Yahoo! Answers and Yahoo! News.
- iv. **Baidu:** Dominant in China's market due to the country's Internet regulations, Baidu provides search services primarily in the Chinese language and offers features tailored to Chinese users.
- v. **DuckDuckGo:** Focused on user privacy, DuckDuckGo does not track user activity or personalize search results, appealing to users concerned about data privacy.

Features of Search Engines

Modern search engines offer a range of functionalities to enhance user experience and improve search efficiency.

- i. **Autocomplete and Suggestions:** Predictive text features help users refine their queries by suggesting popular search terms.

- ii. **Voice Search:** With the proliferation of smart devices, voice-activated search allows users to perform searches through spoken commands.
- iii. **Image and Video Search:** Advanced algorithms enable users to search for visual content, including reverse image searches where users can find information related to a specific image.
- iv. **Local Search Results:** Search engines provide geographically relevant results, essential for finding local businesses and services.
- v. **Personalization:** By analyzing user behaviour and preferences, search engines tailor results to individual users, enhancing relevance.

3.3. The Role of Search Engines in Internet Exploration

Search engines are integral to how users interact with the Internet, impacting various aspects of digital life.

1. Information Accessibility: They democratize access to information, making a vast array of content readily available to anyone with Internet access.

2. Education and Research: Students and professionals rely on search engines for academic research, accessing scholarly articles, and staying informed on developments in their fields.

3. E-Commerce: Consumers use search engines to find products and services, compare prices, and read reviews, influencing purchasing decisions.

4. Digital Marketing: Businesses optimize their online presence through Search Engine Optimization (SEO) and Search Engine Marketing (SEM) to increase visibility and reach target audiences.

3.4. Privacy and Ethical Considerations

While search engines offer significant benefits, they also present concerns regarding privacy and ethics.

Data Collection and Privacy: Search engines collect vast amounts of user data, raising concerns about how this information is used and protected.

Algorithmic Bias: There is a risk of bias in search algorithms, which can inadvertently reinforce stereotypes or exclude certain information, affecting the objectivity of search results.

Filter Bubbles: Personalized search results can create echo chambers where users are only exposed to information that aligns with their existing beliefs.

Censorship and Information Control: In some regions, government policies may influence search engine results, leading to censorship and restricted access to information.

3.5. Future Trends in Search Engine Technology

Advancements in technology continue to shape the evolution of search engines.

- i. **Artificial Intelligence (AI) and Machine Learning:** AI enhances search algorithms, enabling a better understanding of user intent and more accurate results.
- ii. **Natural Language Processing (NLP):** Improved NLP allows search engines to comprehend complex queries and deliver more nuanced answers.
- iii. **Visual and Voice Search Advancements:** As technology improves, visual and voice search are becoming more prevalent, necessitating adjustments in how content is optimized for these formats.
- iv. **Blockchain Technology:** Blockchain has the potential to decentralize search engines, offering increased privacy and security for users.
- v. **Ethical AI Implementation:** There is a growing emphasis on developing ethical AI practices to mitigate biases and ensure fairness in search results.

Search engines are fundamental tools for navigating the vast expanse of the Internet. They enable users to access information efficiently, support academic and professional research, and drive economic activity through e-commerce and digital marketing. As technology advances, search engines will continue to evolve, offering improved functionalities and facing new challenges in privacy and ethics. It is essential to balance innovation with responsible practices to ensure that search engines remain valuable resources for global Internet exploration.

4.0. Self-Assessment Questions

1. Which of these is true about search engines?

(a) Search engines enable recovery of lost items on the internet.

(b) The primary functions of a search engine include crawling, indexing, and retrieving relevant results based on user queries.

(c) Chrome and Firefox are examples of search engines.

1. Which of these best represents an example of a search engine?

(a) Google Search, Bing, Yahoo Search, Baidu, DuckDuckGo

(b) Google Search, Bing, Yahoo Search, Baidu, Yahoo mail

(c) Gmail, Bing, Yahoo Search, Baidu, DuckDuckGo

2. The following are features of search engines.

(a) Autocomplete (b) Voice search (c) browser

3. Discuss 3 roles of search engines in internet exploration.

4. Itemize 4 privacy and ethical considerations about search engines.

5. Itemize 4 future trends in search engines.

6. Which of the following list best describe privacy and ethical considerations In search engines?

(a) Data Collection and Privacy, Algorithmic Bias, Filter Bubbles, Censorship and Information Control.

(b) Data Collection and Privacy, browser, access, Censorship and Information Control.

(c) Data Collection and Privacy, Data Loss, Data Leakage, Censorship and Information Control.

5.0. Tutor Marked Assignments

1. Discuss features of the 3 latest search engines that you know.

Unit3: Network Security

1.0 Introduction

Imagine a world where the internet has no network security. Every piece of information sent across the web is exposed for anyone to see. Personal emails, bank details, and private messages are all out in the open. Without the protections we have today, using the internet would be a risky

endeavour. Cybercriminals would have an easier time accessing sensitive data. They could intercept your passwords, read your confidential emails, and even steal your identity. Online banking and shopping would become dangerous activities. People might think twice before entering their credit card information on any website.

Viruses and malware would spread unchecked. Without firewalls and antivirus programs, computers and devices would be vulnerable to attacks. Hackers could take control of systems, leading to data loss or corruption. The overall functionality of the internet could be compromised due to rampant malicious activities. Privacy would be a significant concern. Without network security, there's no guarantee that your conversations or browsing history are private. Anyone could monitor your online activities, leading to potential misuse of information. This lack of privacy could hinder free expression and the sharing of ideas online.

Businesses would suffer as well. Companies rely on network security to protect their proprietary information and customer data. Without it, they risk losing valuable assets and facing legal consequences. Trust in online services would diminish, affecting e-commerce and cloud-based operations. The lack of network security would also impact critical infrastructure. Services like electricity grids, transportation systems, and healthcare networks could be susceptible to attacks. This vulnerability could lead to disruptions in essential services that society relies on daily. Overall, the internet without network security would be a perilous place. The lack of protection would affect individuals, businesses, and governments alike. It's clear that network security is crucial for maintaining the integrity and usability of the internet we use today.

In this unit, students will learn about safeguarding information and maintaining the integrity of networks. It also entails common network threats and mitigation and emerging trends in network security.

2.0 Learning Outcomes

At the end of this unit, you should be able to:

- i. Discuss network security.
- ii. Itemize 4 importance of network security.
- iii. Discuss 5 common threats to network security.

- iv. Discuss 4 network security measures and best practices.
- v. Students should be able to list common network security threats.
- vi. discuss methods of mitigating network security threats
- vii. Itemize 3 emerging trends in network security.
- viii. Discuss 4 challenges of network security.

Network Security

Network security encompasses a range of strategies, technologies, and practices designed to protect the integrity, confidentiality, and accessibility of computer networks and their data. In an increasingly interconnected world, where cyber threats are prevalent and sophisticated, establishing robust network security measures has become imperative for organizations of all sizes. Effective network security is not merely a technological concern but a comprehensive approach that includes policies, procedures, and user awareness. Organizations must remain vigilant and proactive in their efforts to safeguard their digital assets. By implementing robust network security measures and staying informed about the latest threats and trends, they can protect themselves against the ever-present risks in the digital landscape.

3.0 Importance of Network Security

safeguarding information and maintaining the integrity of networks are pivotal concerns for organizations and individuals alike. Network security involves the systematic approach to protecting networking infrastructure from unauthorized access, misuse, malfunction, modification, destruction, or improper disclosure. Implementing robust network security measures is essential to ensure data confidentiality, integrity, and availability. The rapid evolution of technology and the increasing reliance on computer networks for critical operations have amplified the importance of network security. Key reasons for prioritizing network security include:

1. Protection of Sensitive Information: Networks often carry confidential data such as personal information, financial records, and intellectual property. Unauthorized access to this data can lead to severe consequences, including identity theft and financial loss.

2. Prevention of Unauthorized Access: Effective network security controls prevent unauthorized users from accessing network resources, thereby protecting systems from potential threats.

3. Ensuring Business Continuity: Security breaches can disrupt operations, leading to downtime and revenue loss. Robust network security helps maintain uninterrupted services.

4. Compliance with Regulations: Many industries are subject to regulations that mandate the protection of data, such as the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA).

5. Protection Against Cyber Threats: Cyber threats are increasingly sophisticated, with attackers employing advanced techniques to infiltrate networks. Network security measures protect against various forms of cyber-attacks.

3.1. Common Threats to Network Security

Understanding the landscape of potential threats is crucial for developing effective security strategies. Common threats include:

Malware: Malicious software designed to damage or disable computers and networks. Examples include viruses, worms, and Trojan horses.

Phishing Attacks: Deceptive attempts to obtain sensitive information by masquerading as trustworthy entities in electronic communication.

Denial-of-Service (DoS) Attacks: Attacks intended to shut down a machine or network, making it inaccessible to intended users.

Man-in-the-Middle (MitM) Attacks: An attacker secretly intercepts and possibly alters the communication between two parties who believe they are directly communicating with each other.

Insider Threats: Risks posed by individuals within the organization who have authorized access but misuse their privileges.

Advanced Persistent Threats (APTs): Prolonged and targeted cyber-attacks in which an intruder gains access to a network and remains undetected for an extended period.

SQL Injection: Insertion of malicious SQL statements into an entry field for execution, allowing attackers to tamper with databases.

3.2 Network Security Measures

Implementing a multi-layered defense strategy is essential for effective network security. Key measures include:

Firewalls: Hardware or software systems that control incoming and outgoing network traffic based on predetermined security rules, forming a barrier between trusted and untrusted networks.

Intrusion Detection and Prevention Systems (IDPS): Monitor network traffic for suspicious activity and issue alerts or take action to prevent intrusions.

Encryption: Protects data confidentiality by converting information into a secure format during transmission.

Virtual Private Networks (VPNs): Secure connections over public networks, allowing remote users to access the network securely.

Access Control: Restricts access to network resources through authentication and authorization mechanisms, ensuring users have appropriate permissions.

Antivirus and Anti-Malware Software: Detects, prevents, and removes malicious software from devices and networks.

Security Policies and Procedures: Establishing clear guidelines for users, including acceptable use policies, incident response plans, and regular security training.

Regular Updates and Patch Management: Keeping software and systems up-to-date to protect against known vulnerabilities.

Network Segmentation: Dividing the network into smaller segments to limit the spread of cyber-attacks.

Monitoring and Logging: Continuous monitoring of network activity to detect anomalies and maintain logs for forensic analysis.

Multi-Factor Authentication (MFA): Requires multiple methods of verification before granting access to resources, reducing the risk of unauthorized access.

Security Information and Event Management (SIEM): Aggregates and analyzes security data from various sources for real-time analysis and threat detection.

Next-Generation Firewalls (NGFWs): Offer advanced features beyond traditional firewalls, such as application awareness and intrusion prevention.

Artificial Intelligence and Machine Learning: Utilized for proactive threat detection by identifying patterns and anomalies indicative of cyber threats.

Endpoint Detection and Response (EDR): Monitors endpoint devices to detect and respond to cyber threats in real time.

Challenges in Network Security

Despite the availability of advanced security measures, organizations face challenges:

Evolving Threat Landscape: Cyber threats are constantly evolving, requiring organizations to stay updated on the latest security trends and threats.

Resource Constraints: Implementing comprehensive security measures can be costly, and organizations may have limited budgets for security initiatives.

Human Error: Employees may inadvertently compromise security through negligence or lack of awareness.

Complexity of Security Solutions: Implementing and managing complex security systems requires specialized expertise.

Balancing Security and Usability: Stricter security measures can impede user experience, necessitating a balance between security and usability.

3.4 Emerging Trends in Network Security

Staying abreast of emerging trends is essential for proactive security management:

Zero Trust Architecture: A security model that assumes no implicit trust and requires continuous verification for access to resources.

Cloud Security: With the increasing adoption of cloud services, securing data and applications in the cloud has become a priority.

Internet of Things (IoT) Security: Protecting a growing number of connected devices that can be exploited if not secured properly.

Blockchain Security Solutions: Leveraging blockchain technology for secure transactions and data integrity.

Security Automation and Orchestration: Automating security processes to improve response times and reduce human error.

Network security is a critical aspect of modern information technology infrastructure. As cyber threats become more sophisticated, it is imperative for organizations to implement comprehensive security strategies that encompass technological solutions, policies, and user education. By adopting a proactive approach to network security, organizations can protect their assets, ensure compliance with regulations, and maintain the trust of their stakeholders.

4.0. Self-Assessment Questions

1. Which of these is not true about network security?

(a) Network security encompasses a range of strategies, technologies, and practices designed to protect the integrity, confidentiality, and accessibility of computer networks and their data

(b) network security regulates the network.

(c) By implementing robust network security measures and staying informed about the latest threats and trends, organizations can protect themselves against the ever-present risks in the digital landscape

2. Itemize 4 importance of network security.

3. list common network security threats.

4. Which of the following best describes network security threat measures?

(a) Firewalls, IDPS, VPNs, Access Control, Antivirus and Anti-Malware Software.

(b) Firewalls, IDPS, VPNs, Gmail, AOL

(c) Firewalls, IDPS, VPNs, Access Control, Antivirus and Anti-Malware Software.

5. Itemize 3 emerging trends in network security.

7. Discuss 4 challenges of network security.

5.0. Tutor Marked Assignments

1. What is the primary purpose of a firewall in network security?
2. Which layer of the OSI model do firewalls primarily operate? State the reason for your answer.
3. What does the term “phishing” refer to in the context of network security?

References

- Aouedi, O., Vu, T. H., Sacco, A., Nguyen, D. C., Piamrat, K., Marchetto, G., & Pham, Q. V. (2024). A survey on intelligent Internet of Things: applications, security, privacy, and future directions. *IEEE Communications Surveys & Tutorials*.
- Chen, Q., Li, D., & Wang, L. (2024). Network Security in the Internet of Things (IoT) Era. *Journal of Industrial Engineering and Applied Science*, 2(4), 36-41.
- Mu, X., & Antwi-Afari, M. F. (2024). The applications of Internet of Things (IoT) in industrial management: a science mapping review. *International Journal of Production Research*, 62(5), 1928-1952.
- Rustamovich, A. I. (2024). Basic Concepts Related To Internet Capabilities And Internet Usage. *Web of Discoveries: Journal of Analysis and Inventions*, 2(5), 141-146.
- Zhang, X., Fu, X., Xue, Y., Chang, X., & Bai, X. (2024). A review on basic theory and technology of agricultural energy internet. *IET Renewable Power Generation*, 18(7), 1318-1331.
- Zhang, Y., Hu, S., & Chen, L. (2024). Internet technology adoption and firm energy efficiency: Evidence from China. *Technological Forecasting and Social Change*, 201, 123214.