ASSIGNMENT

Module 1:-

Foundation

THEORY EXERCISE:-

• What is a HTTP?

- HTTP stands for *HyperText Transfer Protocol*. It's the foundational language of the web—basically, it's how your browser talks to websites.
- HTTP is like a *messenger* between your phone or computer and a website.
- You ask to visit a site (like clicking a link), and HTTP goes to the website and brings back the stuff—like text, pictures, or videos—so you can see it.
- It's just a way for devices to ask for things and get them back.

Kind of like saying: "Hey, can I have this?" and the website says: "Sure! Here it is."

• What is a Browsers? How they works?

- A browser is like a window that lets you look out into the internet. It's an app (like Chrome, Safari, Firefox, or Edge) that helps you visit websites.

Here's how it works, step by step:

- You type a website name (like google.com) or click a link.
- The browser sends a request to that website's server.
- The server sends back all the stuff you need—like words, images, videos.
- The browser puts it all together and *shows you the page*.

• What is Domain Name?

- A domain name is like the *name or address* of a house but for a website , like www.example.com.

Example:-www.example.com

- **www**: This stands for **World Wide Web** (optional, but common).
- **Example:** This is the **unique name** of website.
- **.com:** This is the **top-level domain(TLD)**.Other common ones include.org .net,.edu.

- **What is hosting?**
  - **Hosting** is like renting space on the internet.

  Imagine This:

  - You made a project on your computer. But only **you** can see it.
  - If you want **everyone** to see it, you need to **upload it** to the **internet**.

  That's what **hosting** does — it makes your website **visible to the world**.

Module 2:-

Fundamentals of World Wide Web

THEORY EXERCISE:

Difference between Web Designer and Web Developer

Aspect	**Web Designer**	**Web Developer**
Main Focus	Looks and feel of the website	Functionality and structure of the website

What They Do	Designs layout, colors, fonts, and images	Builds the actual website using coding
Tools Used	Photoshop, Figma, Adobe XD, Sketch	HTML, CSS, JavaScript, PHP, Python, etc.
Skills Needed	Creativity, graphic design, UX/UI knowledge	Programming, logic, and problem-solving
End Product	A visual mockup or design of the website	A fully functional website
Example Task	Create a beautiful homepage design	Code that makes a contact form work

- A Web Designer makes the website look nice.
- A Web Developer makes the website actually work.
- Designer makes the design.

- **What is a W3C?**
  - **W3C** stands for **World Wide Web Consortium**.
  - W3C is like a rulebook maker for the web.
- **What is Domain?**
  - A domain is the name of a website that you type in the browser to visit it.

- A **domain** (or **domain name**) is like the address of a house but for websites:

Common Domain Extensions:-

- **.com-**commercial
- **.org-**organization
- **.net**-network
- **.in,.uk,.us-**country-specifc
- **.edu**-education

- **What SEO?**

- SEO stands for Search Engine Optimization.
- it's the practice of making your website easier for search engines (like Google) to find, understand, and rank highly in their results when people search for things.

- **What is SDLC life cycle?**

- The Software Development Life Cycle (SDLC).
- SDLC is the step-by-step process used to plan, build, test, and deliver software.

Phase	What Happens
Requirements	Identify and record what users want.
Feasibility	Confirm it's doable with time, money, and tech.
Design	Plan the software's structure and user flow.
Coding	Write the actual code.
Testing	Find and fix bugs.
Deployment	Launch the software to users.
Maintenance	Keep it updated and improve over time.

Module 3
Fundamentals of IT

- **What is a Program?**

- A program (noun) is executable software that runs on a computer—like web browsers, word processors, games, etc.

What is Programming?

- Programming means giving step-by-step

instructions to a computer so it can do a specific task—like showing a message, solving a math problem, or building a website.

Role	Client	Server
Who is it?	The user's device (like a phone or computer)	A powerful computer that stores websites and data
What it does?	Sends requests to the server (like asking for a webpage)	Receives requests and sends back the correct information
Example	Your browser asking to open a website	The website’s computer sending the webpage to your browser
Starts the conversation?	Yes – the client always starts it	No , the server just waits for requests
Sends or receives?	Sends requests, receives responses	Receives requests, sends responses

- **Describe the roles of the client and server in web communication**.

  - **Explain the function of the TCP/IP model and its layers.**

- TCP/IP helps computers send and receive data over the internet.
- Each layer has a specific job, just like steps in sending a package.

- **Explain Client Server Communication:-**

  - It is how your device (the **client**) and a website's computer (the **server**) talk to each other over the internet.
  - **Client asks**, Server **responds**.
  - This is how the internet works behind the scenes.

Client (You):-

- The **client** is usually your **phone, computer, or browser**.
- It sends a request like asking to open a webpage.

Server (Website's Computer):-

- The **server** is a computer that **stores websites, apps, or data**.
- It receives the request and sends back the correct information.

How They Communicate (Step-by-Step):

1. You open your browser and type a website name.
2. The browser (client) sends a request to the   server.
3. The server receives the request.
4. The server sends back the webpage.
5. You see the website on your screen.

Layer	**What Does**	**Example**
1. Application Layer	Where you use the internet (web, email, etc.)	Web browser, YouTube, Gmail
2. Transport Layer	Makes sure the message is **sent fully and correctly**	Like checking if every word is sent
3. Internet Layer	**Chooses the best path** for the data to travel across the network	Like choosing the best road for a trip
4. Network Access Layer	**Sends the data through cables or Wi-Fi**	Like putting the letter in the mailbox

- **How does broadband differ from fiber-optic internet?**

  Broadband:- Broadband is a fast internet connection that lets you use the internet without blocking phone calls. It can use different types of cables like telephone or TV wires.

Fiber-option internet:-Fiber-optic internet is a very fast type of broadband that uses light through tiny glass cables to send data. It gives faster and more reliable internet.

- **What are the differences between HTTP and HTTPS protocols?**
- **HTTP:-HyperText Transfer Protocol**

  HTTP is a rule used by web browsers and websites **to** send and receive data over the internet.

- A way to load websites, but not secure.
- **HTTPS (HyperText Transfer Protocol Secure)**

  HTTPS is the secure version of HTTP.
  It also sends and receives data, but it protects your information using encryption, so hackers can't see it.

- A safe way to load websites that protects your data.

- **What is the role of encryption in securing application, Software Applications and Its Types**

  - Encryption is like locking your data with a secret code so no one else can read it.

What Are Software Applications?

  - Software applications (apps) are programs that help you do tasks on a phone or computer**.**

Type	**What It Means (Easy)**	**Examples**
1. Web Apps	Used through the internet in a browser	Gmail, Google Docs, YouTube
2. Mobile Apps	Run on phones and tablets	WhatsApp, Instagram, TikTok
3. Desktop Apps	Installed on a computer	MS Word, Paint, VLC Media Player
4. System Software	Helps the computer run properly	Windows, macOS, Linux
5. Utility Software	Does special jobs like cleaning or protecting	Antivirus, Disk Cleaner, Backup tool

- **What is the difference between system software and application software?**

System Software:-

- System software helps the computer run and manage hardware.
- Example: Windows, macOS

Application software:-

- Application software helps you do tasks like writing, chatting, or playing.
- Example: MS Word, WhatsApp

Feature	System Software	Application Software
Use	Runs the computer	Helps you do tasks
Works for	The computer	The user (you)
Runs	In the background	When you open it
Examples	Windows, macOS, Android	MS Word, WhatsApp, Chrome

- **What is the significance of modularity in software architecture?**
  - Modularity means breaking a big software program into smaller, separate parts called modules.
  - Modularity makes software easier to build, manage, fix, and grow by dividing it into smaller parts.

- **Why are layers important in software architecture?**

  Layers are important in software architecture because they help **organize the system** into different parts, each with its own job.

- **Common layers in software Architecture:-**

  Presentation Layer – This is what the user sees and interacts with (like the furniture).

  Business Logic Layer – This handles the rules and processes (like plumbing and electricity).

  Data Layer – This manages how data is stored and retrieved (like the foundation).

- **Explain the importance of a development environment in software production.**
  - A **development environment** is where programmers **build and test software** before it goes live.
  - A development environment helps make software **better, safer, and faster to build**.

- **What is the difference between source code and machine code?**
  - **Source code** is the code that programmers write using languages like Python, Java, or C++. It's easy for humans to read.
  - **Source code** = Human language for computers

- **Machine code** is the code that computers understand — made of 0s and 1s (binary).
- **Machine code** = Computer language (binary)

- **Why is version control important in software development?**

- **Version** control helps keep track of changes in code over time**.** version control makes coding safer, easier, and better for teamwork.

- **Common Version Control Tools:**
- • **Git** (most popular)
- • **SVN (Subversion)**
- • **Mercuria**

important:-

- **Undo mistakes** – Go back to older versions if something breaks.
- **Work together** – Many people can work on the same code without conflicts.
- **Track changes** – See who changed what and why.
- **Backup** – Keeps your code safe.

- **What are the benefits of using Github for students?**

- GitHub is a place to store, share, and work on code online**.** GitHub helps students learn, code better, and share their work.

Benefits for students:-

- Collaboration
- Version Control
- Showcase Projects
- Learning Tool
- Real-World Skills
- Free Hosting for Websites
- Documentation
- Networking
- Access to Resource

- **What are the differences between open-source and proprietary software?**

- **Open-source** software is free to use, and anyone can see, change, open and flexible or share the code.

- **Proprietary software** is owned by a company, closed and controlled and you can't see or change the code.

- **Open-source and proprietary differences:-**

Feature	**Open-Source**	**Proprietary**
Cost	Usually free	Usually paid
Code access	Open to everyone	Closed/secret
Customization	Can be changed	Can't be changed
Example	Linux, Firefox	Windows, Microsoft Word

- **How does GIT improve collaboration in a software development team?**

- Git helps teams work on the same code without getting in each other's way. Git makes teamwork easier, safer, and more organized in coding projects.

- **Git improves teamwork:-**

- **Tracks changes** – Everyone's edits are saved and organized.
- **Merges work** – Combines changes from different people easily.
- **Avoids conflicts** – Helps manage who changed what and when.
- **Backups** – Keeps code safe if something goes wrong.

- **What is the role of application software in businesses?**

- Application software helps businesses do their daily tasks like writing, calculating, selling, or managing data.
- Application software is a program that helps you do specific tasks on a computer.

- **What are the main stages of the software development process?**

- The software development process is a step-by-step method used to create, test, and deliver software

1. **Planning** – Decide what the software should do.
2. **Design** – Plan how it will look and work.
3. **Development** – Write the actual code.
4. **Testing** – Check for bugs and fix them.
5. **Deployment** – Release the software to users.
6. **Maintenance** – Keep it updated and working well.

- **Why is the requirement analysis phase critical in software development?**

- **Requirement analysis** helps understand what the software needs to do**.** requirement analysis makes sure **the right software is built** the right way**.**

- **Requirement analysis is like reading the recipe before cooking**. If you miss this step, you might make the wrong dish or **forget important ingredients.**

- **What is the role of software analysis in the development process?**

- software analysis helps build the right software in a smart and organized way.
- Software analysis helps understand and plan what the software should do.

- **Understand the problem** :– Know what the user really needs.
- **Break down tasks** :– Divide the work into smaller parts.
- **Set clear goals** :– Make sure everyone knows what to build.
- **Find risks early:** – Spot problems before coding starts.
- **Explanation**
- **Role:-** Helps identify **what the user or client want**
- **What are the key elements of system design?**

- system design is the blueprint that helps build clear, working, and safe software.
- system design helps create a clear plan to build effective, organized, and safe software.
- System design is the process of planning how a software system will work and how its parts will fit together.

Key elements of System Design:-

1. **Architecture** :– The overall structure of the system.
2. **Modules**:– Smaller parts with specific jobs.
3. **Data Flow** – How information moves through the system.
4. **User Interface (UI)** – How users see and use the system.
5. **Security**:– Protecting data and access.
6. **Performance:** – Making sure the system runs fast and well.

- **Why is software testing important?**

- Software testing is important because it helps make sure the software works correctly.

Finds mistakes – It helps find errors or bugs before the software is used by people.

Saves money – Fixing problems early is cheaper than fixing them later.

Keeps users happy – Good software works smoothly and makes users satisfied.

Keeps data safe – Testing helps protect against hackers and errors that might lose or leak information.

Makes sure everything works – It checks if the software does what it’s supposed to do.

- **What types of software maintenance are there?**

  - Software maintenance is the process of changing, updating, or improving software after it has been released to keep it working well**.**

1. Corrective Maintenance

- **Fixing problems:-**
  This type fixes bugs or errors found after the software is used.

  Example: If a button doesn't work, it's corrected.

  2. Adaptive Maintenance

  - **Adjusting to changes**:-
    This happens when the software needs to work with new systems or technology.

    Example: Updating software to work on a new version of Windows.

3. Perfective Maintenance

Making it better:-

This means improving the software to work faster or look nicer, even if there are no bugs.

Example*:* Adding a new feature or changing the design to make it easier to use.

4. Preventive Maintenance

Avoiding future problems:-

This is done to clean up the code or improve it so problems don't happen later.

Example: Removing old parts of the code that could cause trouble later.

- **What are the key differences between web and desktop applications?**

  - **Web apps** work in a browser, need internet, no install.

- **Desktop apps** need to be installed, work offline, and run on your device.

• **Point**	• **Web Application**	• **Desktop Application**
• Where it runs	• In a web browser (like Chrome, Firefox)	• On a computer (installed on your device)
• Internet needed?	• Yes, usually needs the internet	• No, works without internet (after install)
• Installation	• No install needed	• Must be installed
• Updates	• Done automatically online	• User needs to download updates
• Access	• Can be used from any device with internet	• Can only be used on that one device
• Speed	• Can be slower, depends on internet	• Often faster, uses local system resources
• Examples	• Gmail, YouTube, Google Docs	• MS Word, Photoshop, VLC Media Player

- **What are the advantages of using web applications over desktop applications?**
- Web applications offer several benefits, especially in terms of accessibility, maintenance, and cost.
- Web apps are easier to access, update, and use on any device because they are online-based.
- No installation is needed, and maintenance is simpler for both users and developers

- **What role does UI/UX design play in application development?**

  - UI/UX design makes an app look good and easy to use.
    It helps users enjoy the app and use it without confusion.
  - **UI (User Interface)** = How the app **looks** (buttons, colors, layout)
  - **UX (User Experience)** = How the app **feels to use** (easy, fast, smooth).
  - UI (User Interface) and UX (User Experience) design are critical for creating applications that are functional, user-friendly, and visually appealing

- **What are the differences between native and hybrid mobile Apps?**

Native apps: Built just for one platform, fast, and use full phone features.

Hybrid apps: One app for many platforms, cheaper and quicker, but may be a bit slower.

Feature	**Native App**	**Hybrid App**
Built for	One platform only (like Android *or* iOS)	Multiple platforms (Android **and** iOS)
Performance	Very **fast and smooth**	**Slower** than native sometimes
Access to phone features	Full access (camera, GPS, etc.)	Limited access, may need plugins
User experience	Best look and feel	Not always as smooth
Development time	Takes **more time** (separate apps)	**Faster** (one app for both systems)
Cost	**More expensive**	**Cheaper** to build
Examples	Instagram (originally native), WhatsApp	Facebook, Instagram (some parts hybrid)

- **What is the significance of DFDs in system analysis?**

  - Data Flow Diagrams (DFDs) are visual tools used in system analysis to represent the flow of data and how it is processed within a system. They play a crucial role in understanding and designing systems.
  - DFDs are like **maps** for a system, showing how **data moves** from one place to another and how it gets processed, helping everyone involved **understand** and **build** the system better.

- **What are the pros and cons of desktop applications compared to web applications?**

  - **Pros (Advantages) – meaning: good things or benefits**

  These are the positive points about desktop applications:-

1. Faster Performance:
   Desktop apps usually work faster because they use your computer's power directly.
2. Work Offline:
   You don't need internet to use most desktop apps.
3. More Features:
   They often have more advanced tools and settings.
4. Better Security:
   Since the data is stored on your computer, it can be more private.

  - **Cons (Disadvantages) – meaning: bad things or problems**

  These are the negative points about desktop applications:-

1. Need to Install:-You have to download and install the app on each device.
2. Updates Are Manual-:You need to update it yourself, unlike web apps which update automatically.
3. Only on One Device:-You can use it only on the device where it's installed.
4. Can Use More Storage:-It takes space on your hard drive.

- **How do flowcharts help in programming and system design?**
- A **flowchart** is a **diagram** that shows the steps of a process using arrows and shapes. It looks like a map of what the program or system will do.
- **Helps Plan:**
  It helps you **think and plan** what your program or system should do.
- **Easy to Understand:**
  It’s easier to **understand** a flowchart than reading long code.
- **Shows Steps Clearly:**
  It shows the steps in order – what happens first, next, and last.
- **Find Mistakes Early:**
  You can find and fix problems **before** writing any code.
- **Good for Sharing Ideas:**
  It helps you **explain your ideas** to others easily.