Module 1 – Foundation

1. What is a HTTP?

HTTP(Hypertext Transfer Protocol) is fundamental protocol used for data communication on the World Wide Web.

It is the protocol used for client and server communication like web communication.

HTTP Works:-

1. Client sends a request to a server.
2. The Server processes the request and return a response.
3. The client render or user response.

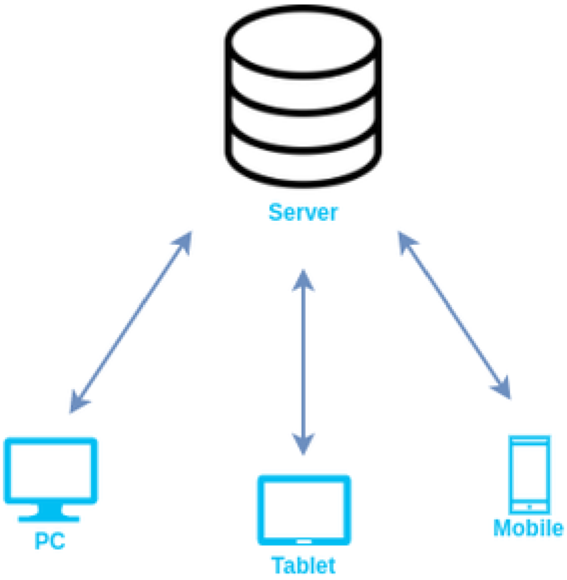
2. What is a Browsers? How they works?

A **web browser** is a **software application** that allows users to access, retrieve, and view content on the **World Wide Web.**

It displays web pages written in **HTML, CSS, JavaScript**, and other web technologies.

Examples: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari, Opera.

How Do Browsers Work?



- You enter a website address (URL) or click a link.
- The browser sends a request to the website's server.
- The server responds by sending the webpage's data (HTML, CSS, JavaScript, images, etc.).
- The browser processes this data and displays the webpage.
- You can interact with the page by clicking, scrolling, or entering information.

3. What is Domain Name?

A domain name is the human-readable address of a website that you type into a web browser to visit it. It serves as an easy-to-remember shortcut to the website's actual location on the internet, which is identified by an IP address (a string of numbers).

Example:

- Domain name: google.com
- Corresponding IP address: 142.250.190.14 (example only)

Structure of a Domain Name:

A typical domain name consists of:

- Subdomain (optional): www
- Second-level domain (SLD): google
- Top-level domain (TLD): .com

So in www.google.com:

- www is the subdomain
- google is the second-level domain
- .com is the top-level domain

4. What is hosting?

Hosting refers to the service of providing storage space and access for websites or applications on the internet.

When you create a website, all the files, images, and code that make it work need to be stored somewhere so that people around the world can access it at any time. Hosting companies provide this service by maintaining powerful computers called servers that store website data and keep it available online 24/7.

There are different types of hosting, including:

- Shared Hosting
- VPS Hosting
- Dedicated Hosting
- Cloud Hosting
- Managed Hosting

Module 2 – Fundamentals of World Wide Web

1. Difference between Web Designer and Web Developer

Feature	Web Designer	Web Developer
Main Focus	Visual design & UX	Functionality & performance
Tools Used	Figma, Adobe XD, Photoshop	VS Code, Git, Terminal, Browser Dev Tools
Skills	UI/UX, layout, branding	HTML, CSS, JavaScript, databases
Outcome	Design mockups and prototypes	Functional website

2. What is a W3C?

The World Wide Web Consortium (W3C) is the main international standards organization for the World Wide Web.

Standard	**Description**
HTML	Structure of web pages
CSS	Styling of web pages
WCAG	Web Content Accessibility Guidelines
DOM	Document Object Model
XML	Extensible Markup Language

3. What is Domain?

A **domain** is the **address** people use to visit a website on the internet.

For example:

- **google.com**
- **facebook.com**
- **example.org**

These are all domain names.

Parts of a Domain Name:

Take the domain **www.example.com**:

Part	**Meaning**
www	(optional) Subdomain
example	Second-level domain (your custom name)
.com	Top-level domain (TLD) like .com, .org, .in

4. What SEO?

SEO stands for Search Engine Optimization.

SEO or Search Engine Optimization is defined as the **process of improving (optimising) the visibility of a website/webpage on Search Engines**, such as Google, Bing, etc.

SEO (Search Engine Optimization) = making your website easier for Google (and people!) to find.

5. What is SDLC life cycle?

SDLC stands for **Software Development Life Cycle**.

It’s a step-by-step process used by software developers to **design, develop, test, and deploy software** effectively and efficiently.

SDLC Life Cycle (Simple Explanation)

Here are the **main stages** of the SDLC:

1.Planning:- "What are we building?"

- Think about what the software should do.
- Set goals, cost, and timeline.

2. Requirements Gathering:- "What do users need?"

- Talk to users and stakeholders.
- List all the things the software must do.

3. Design:- "How will it look and work?"

- Create a blueprint or plan for how the software will look and work (UI/UX, architecture).

4. Development:- "Let’s build it!"

- Write the actual code to build the software.

5. Testing:- *"Does it work?"*

- Check the software for bugs or problems and fix them.

6. Deployment:- "Launch it!"

- Launch the software for users to use.

7. Maintenance:- "Keep it working and updated."

- Update the software, fix bugs, and improve it over time.

Module 3 – Fundamentals of IT

1. Explain in your own words what a program is and how it functions.

A program is a set of instructions written to tell a computer what to do.

It performs a specific task like adding numbers, showing a message, or saving data.

It is written by people (called **programmers**) using special languages like **Python**, **Java**, or **C++**.

How Does a Program Function?

1. You **write the instructions** in a programming language.

2. The **computer reads** these steps one by one.

3. It **executes** them exactly as told — like doing math, storing information, or showing output.

4. A program may also **take input**, **process it**, and **give output**.

2. What is Programming?

Programming is the process of writing those instructions using a programming language (like Python, C++, or Java).

It means creating programs that solve problems or perform tasks.

3. What are the main differences between high-level and low-level programming languages?

- **High-Level Language**
  - Easy to read and write
  - Looks more like **English**
  - Works on **many types of computers**
  - **Examples:** Python, Java, C++, JavaScript
  - It is **Slower**

- **Low-Level Language**
  - Hard to read and write
  - Looks like **machine instructions**
  - Works only on **one type of computer**
  - **Examples:** Assembly, Machine code
  - It is **Faster**

4. Describe the roles of the client and server in web communication.

- The client is what the user uses, like a browser (Chrome, Firefox) or a mobile app.
- It sends a request to the server asking for something (like a webpage).
- It shows the server's response on the screen.

Server – Backend Side:

- The server is a computer where websites and data are stored.
- It receives requests from clients and sends back the correct information.
- It does the background work like fetching data or running logic.

Simple Example:

1. You type www.google.com in your browser (you are the client).
2. The browser sends a request to Google's server.
3. The server sends back the Google homepage.
4. Your browser shows it on your screen.

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

- Here the tcp’s full form(transmission control protocol) and ip(internet protocol).
- Tcp/ip model is the set of rules the define how data is transmitted over the internet.
- This model’s main function is the reliable communication between computers across different networks.
- This cummication process into 4 layers:-
- **1.application layer:-** interfaces with the user; handles software-related communication(protocol’s:-HTTP,FTP,DNS).
- **2.transport layer:-** Ensures reliable data transfer between devices; handles errors & sequencing(protocol’s:-TCP,UDP).
- **3.internet layer:-** Handles logical addressing and routing across networks(protocol’s:-IP,ICMP,ARP).
- **4.network access layer:-** Deals with the physical transmission of data over a network(protocol’s:-ethernet,MAC).

6. Explain Client Server Communication.

- ➢ Here the step by step explanation of client server communication.
    - ➢ **Client initiates communication** – It sends a request to the server.
    - ➢ **Server receives the request** – It checks the data, processes it, and prepares a response.
    - ➢ **Server sends a response** – This could be success data, error message, or requested info.
    - ➢ **Client receives and processes the response** – It displays or uses the information in the app/game.

7. How does broadband differ from fiber-optic internet?

➢ **Broadband:-**

- A general term for high-speed internet using various technologies (DSL, cable, satellite, etc.)
- It's transmit speed is 100 to 300 Mbps.
- Broadband uses coper wires.
- Some times in broadband faces high latency.
- Broadband affected distance, weather, and electromagnetic interference.
- Broadband's cost is very low.

➢ **Fiber-optic internet:-**

- A specific type of broadband that uses fiber-optic cables to transmit data as light signals.
- It's transmit speed is 1Gbps and more.
- Uses glass or plastic fibers that transmit light.
- In fiber-optic sometimes faces very low latency.
- Fiber-optic is highly reliable and less affected by external conditions.
- Fiber-optic is more expensive than broadband.

8. What are the differences between HTTP and HTTPS protocols?

➢ **HTTP:-**

- HTTP full fom is hyper text transper protocol.
- HTTP is sent data as plain text.
- That's means HTTP is not secure.
- HTTP port number is 80.

- HTTPS is used for basic websites or internal networks.
- In SEO(search engine optimization) HTTP priority is very low.

- **HTTPS:-**
  - HTTPS full form is hyper text markup language.
  - HTTPS sent encrypted data.
  - That means HTTPS is more secure than HTTP.
  - HTTPS port number is 443.
  - Data is safe from tampering.
  - HTTPS uses banking, e-commerce, login systems.
  - In SEO(search engine optimization) HTTPS priority is high.

9. What is the role of encryption in securing application, Software Applications and Its Types.

- Encryption is the process of converting data into a coded format so that only authorized users can read or access it.
- Role of encryption:-
  1. Protects sensitives data
     - Keeps passwords, personal info, payment data safe from hackers.
  2. Ensures privacy
     - Prevents third parties from reading user data during transmission.
  3. Data integrity

- Ensures the data is not altered while being sent or stored.

4. Secure communication
   - Used in HTTPS, messaging apps, email, etc.
5. compliance
   - Helps meet legal and security standards (like GDPR, HIPAA).

- **Software application and its types:-**

types	**description**	**Examples**
System software	Helps run computer hardware and system	Windows, Linux, macOS
Application software	For end-users to perform tasks	MS Word, Photoshop, Zoom
web applications	Runs in browser over internet	Gmail, Facebook, YouTube
Mobile applications	Designed for smartphones/tablets	WhatsApp, Instagram, Paytm
Enterprise software	For business use	ERP, CRM, SAP
Game software	For business use	PUBG, Minecraft, FIFA

10. What is the difference between system software and application software?

- ➢ **System software:-**
  - Manages hardware and system operations.
  - Works in the background.
  - Ex. Windows, Linux, macOS, Device Drivers.
  - Needed for running application software.
  - Usually comes pre-installed.

- ➢ **Application software:**-
  - Helps users perform specific tasks.
  - Directly used by the user.
  - EX. MS Word, WhatsApp, Photoshop.
  - Runs on top of system software.
  - Installed as per user needs.

11. What is the significance of modularity in software architecture?

- ➢ Modularity means dividing a software system into smaller, independent parts called modules, where each module performs a specific task or function.
- ➢ **Why modularity is important**

  1. Easy maintenance

  - Bugs or issues can be fixed in one module without affecting the whole system.

2. Reusability

- Modules can be reused in other projects or parts of the software.

3. improved development

- Different teams can work on different modules at the same time.

4. Better testing

- Each module can be tested independently (unit testing).

5. Scalability

- New features or modules can be added easily without disturbing existing ones.

6. Improved code management

- Makes the software easier to read, understand, and manage.

12. Why are layers important in software architecture?

- ➢ Layers in software architecture refer to organizing the system into separate levels, each with a specific role. This is called layered architecture.
- ➢ **Importance of layers:-**

1. Separation of concerns
    - Each layer handles a specific responsibility (e.g., UI, logic, data), reducing complexity.
2. improved maintainability

- Changes in one layer (like design) don’t affect others (like database).

3. Reusability
   - Layers can be reused across multiple applications or modules.
4. Scalability
   - Easier to upgrade or scale specific layers (like switching to a better database layer).

5 Testability

- Each layer can be tested independently (unit testing or integration testing).

6. security
   - Sensitive logic and data can be hidden in deeper layers, not exposed to the user.

13. Explain the importance of a development environment in software production.

➢ A development environment is a set of tools and resources that developers use to build, test, and debug software applications. It plays a crucial role in software production for several reasons.

1. **Efficient development:-**
   - A good development environment provides tools like code editors, compilers, debuggers, and version control.

- These tools help developers write, test, and fix code faster, increasing productivity.

2. **Debugging and testing:-**
    - Development environments offer debugging tools to identify and fix errors in the code.
    - They also support unit testing, which ensures that each part of the program works correctly.
3. **Consistency across teams:-**
    - A shared development environment ensures that all developers work with the same versions of tools and libraries.
    - This reduces errors caused by differences in setup between team members.
4. **Automation and Integration:-**
    - Many development environments support automation tools (e.g., build tools, continuous integration).
    - These help in automating repetitive tasks like building and testing the code, speeding up development cycles.
5. **Version Control Integration:-**
    - Most environments integrate with Git or other version control systems.
    - This allows tracking changes, collaborating with team members, and managing multiple versions of the software easily.
6. **Simulated environments for testing:-**
    - Developers can use local environments or virtual setups to simulate the production environment.

14. What is the difference between source code and machine code?

- **Source code:-**
  - The human-readable instructions written by a programmer using a programming language like C, Java, Python, etc.
  - Source code write in High-level languages (e.g., C++, Python) or low-level languages (Assembly).
  - Source code is readable by humans.
  - Ex. Print(“hello world”) python code.
  - Source code must be compile or interpreted to be run by computer.
  - Programmers can modify source code easily.

- Machine code:-
  - The binary code (0s and 1s) that the computer's processor understands and executes directly.
  - Machine code is write in binary.
  - Machine code is readable by computers.
  - Ex. 11110000 01100111
  - Generated by compiling or interpreting the source code.
  - Machine code is very difficult to modify by humans.

15. Why is version control important in software development?

- ➢ Version control is very important in software development because it helps developers manage changes to the code over time.

1.tracks every changes:-

- Version control systems (like Git) keep a record of all code changes.
- You can see who made changes, when, and why.
- If something breaks, you can easily go back to a previous working version.

2. Team collaboration:-

- In team projects, many developers work on the same codebase.
- Version control lets them work together without overwriting each other's work.
- It helps merge changes from multiple developers safely.

3. Safe experimentation:-

- You can create a branch to test new features without affecting the main code.
- If the experiment fails, just delete the branch—no harm done.

4. Easy bug fixing:-

- When a bug is found, version control helps you track down when the bug was introduced.

- This makes it easier to fix issues.

16. What are the benefits of using Github for students?

➢ Using GitHub offers many valuable benefits for students, especially those learning programming or working on projects.

1. Learn real world tools:-

- GitHub is used by professional developers all over the world.
- By using it early, students learn how software is built and managed in the real world.

2. Build a portfolio:-

- Students can upload their projects to GitHub.
- This becomes a public portfolio to show their skills to employers or internships.
- Recruiters often check GitHub profiles to see code quality and activity.

3. Collaboration and teamwork:-

- Students can work on group projects using GitHub.
- It helps manage code, track who did what, and avoid conflicts.
- Perfect for college assignments or hackathons.

4. Version control:-

- GitHub is based on Git, so students learn version control automatically.
- They can go back to earlier versions, recover lost code, or fix mistakes easily.

5. Free hosting for projects:-

- GitHub Pages lets students host websites for free (ideal for portfolios or project demos).
- Great for web development students to showcase live projects.

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

➢ **Open source software:-**

- Available to everyone.
- Users can modify and improve the code.
- Usually free.
- Open licenses (e.g., GPL, MIT).
- Community-based support.
- Open to inspection by anyone (bugs can be fixed faster).
- Linux, VLC Media Player, GIMP.

➢ **Priopprietary software:-**

- Not shared with the public.
- Only the company can modify it.
- Often paid or licensed.
- Restricted licenses (e.g., Microsoft EULA).
- Official customer support.
- Security is managed by the company only.
- Windows, MS Office, Adobe Photoshop.

18. How does GIT improve collaboration in a software development team?

➢ GIT improves collaboration in a software development team in several important ways:

1. **Version control:-**
   - Git keeps track of every change made to the codebase.
   - Developers can see who made what changes and when, which helps in tracking progress and understanding the project history.
2. **branching and merging:-**
   - Each developer can work on their own branch without affecting the main codebase.
   - Once the feature is complete, it can be merged into the main branch after review.
3. **Conflict management:-**
   - Git detects conflicts when two developers modify the same part of the code.
   - It helps teams resolve these merge conflicts efficiently, maintaining code integrity.
4. **Collaboration with remote repositories:-**
   - Git works with services like GitHub, GitLab, and Bitbucket.
   - Developers can push (upload) and pull (download) changes from a central repository, enabling remote collaboration.
5. **Code review and feedback:-**

- Tools like GitHub allow pull requests, where code can be reviewed and discussed before merging.
- This improves code quality and helps in team learning.

19. What is the role of application software in businesses?

➢ Here the role of application software in business.

1. **Automating Business Processes:**
    - Application software like ERP (Enterprise Resource Planning) and CRM (Customer Relationship Management).
2. **Data management and analysis:**
    - Software like Excel or database applications helps businesses:
    - Store, retrieve, and manage large amounts of data.
    - make data-driven decisions.
3. **Communication and Collaboration:**
    - Tools like Microsoft Teams, Slack, Zoom, and email clients allow team members
    - Share files
    - Hold meetings
    - Collaborate on documents in real-time
4. **Security and Compliance:**
    - Security software ensures data privacy and compliance with regulations (e.g., GDPR).
    - Antivirus, firewall, and encryption tools protect business data.

20. What are the main stages of the software development process?

- **1. Requirement analysis:-**
    - Understand what the client or user needs.
    - Gather and document all functional and non-functional requirements.

  2. System design:-
    - Plan the architecture of the software.
    - Design components like user interfaces, databases, and system modules.

  3. coding:-
    - Developers write the actual code based on the design documents.
    - Programming languages and tools are selected and used.

  4. Testing:-
    - Test the software for bugs, errors, and performance issues.
    - Includes unit testing, integration testing, system testing, and acceptance testing.

  5. Deployment:-
    - Release the software to the production environment.
    - It becomes available for end-users.

  6. Maintenance:-
    - Fix issues that arise after deployment.
    - Update the software with new features or improvements.