Module 1 – Overview of IT Industry

Q. Write a simple "Hello World" program in two different programming languages of your choice. Compare the structure and syntax. **Ans-**

1. Python Program:
```
# Hello World in Python Print
("Hello, World!")
```
2. C Program:
```
// Hello World in C #include<stdio.h>
int  main()  {  printf("Hello,  World!");
return
            0;
             }
```

Q. Research and create a diagram of how data is transmitted from a client to a server over the internet.
Ans-

Client (Browser / App)

|

| Request: "GET /data"

↓

[TCP/IP Stack]

↓

[Router / Internet]

↓

[TCP/IP Stack]

↓

Server (e.g., Web Server)

|

| Response: "Here is your data"

↑

[TCP/IP Stack]

↑

[Router / Internet]

↑

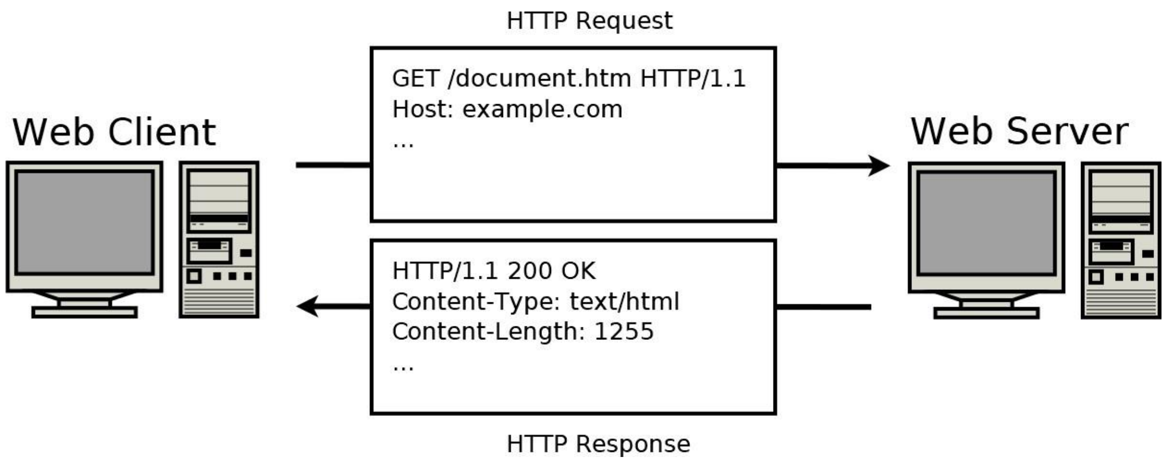
[TCP/IP Stack]

↑

Client receives data

Q. Design a simple HTTP client-server communication in any language.

Ans-



```
GET /document.htm HTTP/1.1
Host: example.com
...
```

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1255
...
```

Q. Research different types of internet connections (e.g., broadband, fiber, satellite) and list their pros and cons.

Ans-

1.Digital Subscriber Line(DSL) Pros:

- Widely available
- Allows internet and phone use at the same time
- Affordable for basic users Cons:
- Speed depends on distance from service provider

- Slower compared to modern options like fiber

2. Cable Internet
Pros:
- Faster than DSL
- Suitable for streaming and gaming • Uses existing TV cable lines Cons:
- Shared bandwidth can cause speed drops during peak hours
- Limited availability in rural areas

3. Fiber Optic Pros:
- Very high speed (up to 1 Gbps or more)
- Low latency and highly reliable
- Great for heavy users (streaming, gaming, work-from-home) Cons:
- Limited availability in some regions
- Installation may be expensive

4. Satellite Internet Pros:
- Available in remote and rural areas • Doesn't require cable or phone lines Cons:
- High latency (delay), not good for gaming or video calls
- Weather can affect signal quality
- Data caps and slower speeds

5. Wireless Internet (Mobile Data / Wi-Fi) Pros:
- Convenient and portable
- Easy to set up
- Useful for smartphones and hotspots Cons:
- Speed and reliability depend on signal strength
- May have data limits or be costly

6. Broadband over Power Lines (BPL)
Pros:
- Uses existing electrical infrastructure
- Easy access where other services are unavailable Cons:
- Not widely available
- Interference issues can occur

Q. Simulate HTTP and FTP requests using command line tools (e.g., curl).

Ans-

1. Simulating an HTTP Request Using curl
   Command:

   ```
   curl http://example.com
   ```

   Explanation:

   - This command sends an HTTP GET request to the server at example.com.
   - The server responds with the HTML content of the page.
   - Useful for testing websites or APIs.

2. Simulating an FTP Request Using curl Command
   (to download a file):

   ```
   curl ftp://ftp.example.com/file.txt --user username:password
   ```

   Explanation:

   - Connects to an FTP server.
   - Logs in with provided username and password.
   - Downloads the file file.txt from the FTP server.

Q. Identify and explain three common application security vulnerabilities. Suggest possible solutions.

Ans-

1. SQL Injection
   - Problem: Hacker tricks the app to get into the database.
   - Fix: Check and clean user input.
2. XSS (Cross-Site Scripting)
   - Problem: Hacker puts bad code in a website that runs on other people's screens.
   - Fix: Don't show user input directly. Clean it first.
3. Weak Login System
   - Problem: Easy passwords or no security checks.
   - Fix: Use strong passwords and add OTP or 2-step login.

Q. Identify and classify 5 applications you use daily as either system software or application software.

Ans-

- Google Chrome – Application Software
- Microsoft Word – Application Software
- Windows 10 – System Software
- VLC Media Player – Application Software
- Antivirus (like Quick Heal) – System Software

Q. Design a basic three-tier software architecture diagram for a web application.

Ans-

```
                    [ User / Browser ]
                            |
                 -------------------------
         | 1. Presentation Layer | → HTML, CSS, JS
                 -------------------------
                            |
                 -------------------------
   | 2. Application Layer | → Backend logic (PHP, Python)
                 -------------------------
                            |
                 -------------------------
        | 3. Data Layer       | → Database (MySQL, etc.)
                 -------------------------
```

Q. Create a case study on the functionality of the presentation, business logic, and data access layers of a given software system.

Ans-

1. Presentation Layer (Frontend / UI)

Role: This is what the user interacts with.

- User browses restaurants and food items
- Adds food to cart
- Enters delivery details
- Makes payment

  Technologies Used: HTML, CSS, JavaScript, React, Flutter (for mobile)

2. Business Logic Layer (Application Layer)

Role: This handles all decision-making and rules.

- Processes order and verify payment
- Applies discounts and taxes
- Matches user with nearby delivery agents
- Calculates estimated delivery time

  Technologies Used: Node.js, Java, PHP, Python

3. Data Access Layer (Database Layer) Role: Deals with storing and retrieving data.

- Saves user profiles, orders, and payment info
- Fetches list of restaurants and menus
- Tracks real-time delivery status
- Stores feedback and reviews

  Technologies Used: MySQL, MongoDB, PostgreSQL

Q. Explore different types of software environments (development, testing, production). Set up a basic environment in a virtual machine.

Ans-

Types of Software Environments:

1. Development Environment **o** Used by developers to write and build code **o** Contains IDEs, compilers, and debugging tools

   o Example: VS Code, Python, XAMPP
2. Testing Environment **o** Used by QA (testers) to test features **o** Isolated from development and production **o** Includes tools for

automated/manual testing **o** Example: Selenium, Postman, JUnit

3. Production Environment **o** The live environment where real users access the application **o** Must be stable, secure, and monitored

o Example: Hosted web server (Apache, Nginx), Cloud (AWS, Azure) Basic Virtual Machine Setup (Example using VirtualBox):

1. Install VirtualBox or VMware
2. Create a new virtual machine **o** Choose OS (e.g., Ubuntu or Windows) **o** Allocate RAM and disk space
3. Install a development stack **o** Example for web development:
   - Install Apache, MySQL, PHP (or use XAMPP)
   - Install code editor (e.g., VS Code)
4. Test a basic web page or script **o** Create a hello.php file **o** Run it in the browser from localhost

Q. Write and upload your first source code file to Github.

Ans-

1. Write a Simple Code File

   Create a simple file named hello.py:

   # hello.py print("Hello,

   GitHub!")

2. Create a Repository on GitHub
   - Go to https://github.com

• Click New Repository

• Name it (e.g., first-code)

• Add a description (optional)

• Choose Public

• Click Create repository

3. Upload the Code Using Git (Command Line) Open terminal or Git :

```
git  init  git  add
hello.py
git commit -m "Add hello.py" git branch
-M main
git  remote  add  origin  https://github.com/your-username/first-
code.git
git push -u origin main
```

Q. Create a Github repository and document how to commit and push code changes.

Ans-

Step 1: Create a GitHub Repository:

1. Go to https://github.com

2. Click on “New” to create a new repository

3. Enter a repository name (e.g., my-first-repo)

4. (Optional) Add a description

5. Choose Public or Private

6. Click Create repository

Step 2: Prepare Your Project Locally:

Create a folder and add a file (e.g., main.py):

python #

main.py

print("This is my first commit!")

Step 3: Use Git to Commit and Push Code:

Open Git or Terminal, then run:

```
git init                                # Initialize Git in the folder
git add .                               # Stage all files
git commit -m "Initial commit"                # Commit changes
with a message
git branch -M main                      # Rename default branch
to main git remote add origin https://github.com/your-
username/my-
first-repo.git
git push -u origin main  # Push changes to GitHub Replace your-
```

username with your actual GitHub username.

Summary:

- You created a GitHub repository
- Committed code using Git
- Pushed it to GitHub successfully

Q. Create a student account on Github and collaborate on a small project with a classmate.

Ans-

Tasks to Perform:

1. Create a GitHub account by visiting https://github.com.

2. Set up your profile with your real name and profile photo.

3. Create a new repository named collab-project.

4. Add a README.md file describing the project.

5. Invite your classmate as a collaborator via repository settings.

6. Both team members should commit at least one file each.

7. Explore features like:

o Issues **o** Pull

requests **o** Commit

history Tools Required:

- GitHub account
- Web browser
- Basic internet connection

Q. Create a list of software you use regularly and classify them into the following categories: system, application, and utility software.

Ans-

1.System Software

These manage and control computer hardware and serve as a platform for application software.

- Windows 11 / macOS / Linux – Operating systems
- Android / iOS – Mobile operating systems
- Device Drivers – Graphics card driver, printer driver

2.Application Software

These are programs designed to perform specific tasks for the user.

- Microsoft Word – Word processing

- Google Chrome / Microsoft Edge – Web browsing
- Zoom / Microsoft Teams – Video conferencing
- Spotify / YouTube – Media streaming
- Photoshop / Canva – Photo editing
- MS Excel – Spreadsheet work
- WhatsApp / Gmail – Communication

3.Software

These help maintain, protect, and optimize the computer system.

- Antivirus (e.g., Windows Defender, Avast) – Protects from malware
- WinRAR / 7-Zip – File compression
- CCleaner – System cleaning and optimization
- Backup Tools (e.g., Google Drive, OneDrive) – Data backup
- Disk Cleanup / Disk Defragmenter – System maintenance

Q. Follow a GIT tutorial to practice cloning, branching, and merging repositories.

Ans-

Tasks to Perform:

1. Cloning a Repository:

   o Use git clone to download a remote repository to your local machine.

   o Example: code git clone https://github.com/username/repository-name.git
2. Creating a Branch:

   o Create a new branch to add features without affecting the main code. **o** Example: code git checkout -b feature-branch

3. Making Changes: **o** Edit files, commit the changes using git commit, and push to
   the new branch.
4. Merging Branches:
   - Switch to the main branch and merge the feature branch into it.
   - Example: code-- git checkout main git merge feature-branch
5. Resolve Merge Conflicts (if any):
   - Practice conflict resolution when Git highlights file conflicts.

Q. Write a report on the various types of application software and how they improve productivity.

Ans-

Suggested Report Structure:

1. Word Processing Software:
   - Example: Microsoft Word, Google Docs
   - Productivity Impact: Helps create, edit, format, and print text documents quickly and professionally.
2. Spreadsheet Software:
   - Example: Microsoft Excel, Google Sheets
   - Productivity Impact: Allows data analysis, calculations, chart generation, and financial modeling.
3. Presentation Software:
   - Example: PowerPoint, Canva
   - Productivity Impact: Enables professionals to communicate ideas effectively with visual support.
4. Database Management Software (DBMS):
   - Example: Microsoft Access, MySQL
   - Productivity Impact: Organizes and retrieves structured data efficiently, saving time and effort.
5. Multimedia Software:
   - Example: Adobe Photoshop, VLC Media Player

- Productivity Impact: Facilitates content creation, editing, and consumption (videos, graphics, audio).

6. Web Browsers:
    - Example: Google Chrome, Mozilla Firefox
    - Productivity Impact: Provides access to information, tools, and web applications instantly.
7. Communication Software:
    - Example: Zoom, Microsoft Teams, Slack
    - Productivity Impact: Enables instant messaging, video conferencing, and team collaboration.

Q. Create a flowchart representing the Software Development Life Cycle (SDLC).

Ans-

Phases of SDLC to Include:

- 1. Requirement Analysis
- 2. System Design
- 3. Implementation (Coding)
- 4. Testing
- 5. Deployment
- 6. Maintenance

Sample Flowchart Structure : csharp

[Start]

↓

[Requirement Analysis]

↓

[System Design]

↓

[Implementation]

↓

[Testing]

↓

[Deployment]

↓

[Maintenance]

↓

[End]

You can also include decision points (e.g., after testing: "Is software bug-free?" → Yes → Deploy / No → Return to Coding)

Q. Write a requirement specification for a simple library management system.

Ans-

Sample Requirement Specification Document:

1. Introduction:
   - Purpose: To manage books, members, and borrowing activities in a digital format.
   - Scope: The system will allow librarians to add/remove books, register members, issue/return books, and generate reports.
2. Functional Requirements:
   - The system shall allow the librarian to: **o** Add, delete, and update book records.
     - **o** Register and manage members.
     - **o** Issue books to members. **o** Return books from members. **o** Generate overdue fine reports.
   - The system shall display:
     - **o** Available and borrowed books. **o** Member transaction history.
     - **o** Due date alerts.
3. Non-Functional Requirements:
   - Usability: User-friendly UI for easy navigation.

- Reliability: System should handle simultaneous users and maintain data consistency.
- Security: Login credentials required for librarian and staff access.
- Performance: The system should perform all operations within 2 seconds.
- Portability: Should work on web browsers and desktop platforms.

4. Assumptions:

- Users have basic computer literacy.
- Database is regularly backed up.

Q. Perform a functional analysis for an online shopping system.

Ans-

Functional Requirements of Online Shopping System:

1. User Registration & Login:

- Users must be able to register and securely log in.
- Forgot password and user authentication features included.

2. Product Browsing and Search:

- Users can browse by category, search for products using keywords, and filter results.

3. Shopping Cart:

- Users can add/remove products, view totals, and update quantities.

4. Checkout and Payment:

- System calculates total price with taxes and shipping.
- Supports payment gateways like UPI, Credit/Debit Cards, Net Banking.

5. Order Management:

- Users can view order history, current status (shipped, delivered), and cancel orders.

6. Admin Functionalities:

- Add/update/delete product listings •

  Manage inventory, users, and process orders

7. Feedback and Reviews:

- Customers can leave product ratings and reviews.

Optional Functional Block Diagram:

A diagram showing the flow between:

User → Product Search → Cart → Checkout → Payment → Order Confirmation

Q. Design a basic system architecture for a food delivery app.

Ans-

Tasks to Perform:

1. Identify the main system components and user roles.

2. Design a basic architecture diagram.

3. Describe the role of each component and how data flows through the system.

Architecture Components:

1. Frontend (User Interface):
   - Customer App: Browse restaurants, place orders, track delivery.
   - Restaurant Panel: Accept/prepare orders, update status.
   - Delivery App: Accept delivery tasks, update real-time location.
2. Backend (Application Server):
   - Handles:

     o Order placement logic **o**

     Authentication and user data **o**

     Payment integration

o Notification system (push/SMS/email) **o** Order status updates

3. Database Layer: Stores:
    - User data (login, address, orders)
    - Restaurant menus and availability
    - Payment history and reviews
    - Delivery logs
4. Payment Gateway API:
    - Securely processes transactions via UPI, cards, wallets, etc.
5. Real-Time Tracking System:
    - Uses GPS and mapping APIs (e.g., Google Maps)
    - Tracks delivery location • Shows ETA to customers
6. Notification System:
    - Sends order confirmations, delivery status, offers, etc.

Q. Develop test cases for a simple calculator program.

Ans-

Tasks to Perform:

1. Identify the calculator functions to be tested.

2. Define input values, expected output, and conditions.

3. Organize test cases into a test case table.

Calculator Functionalities to Test:

- Addition (+)

• Subtraction (-)

• Multiplication (*)

• Division (/)

• Handling of invalid inputs

• Division by zero

Q. Document a real-world case where a software application required critical maintenance.

Ans-

Tasks to Perform:

1. Research a known software maintenance case.

2. Describe the problem, its cause, and the maintenance performed.

3. Summarize the outcome and lessons learned.

Case Study:

1. Background:

   WhatsApp, the popular messaging application owned by Meta, faced a global outage on 25th October 2022. Users were unable to send or receive messages for over two hours.

2. Problem Description:

   • Messages were stuck on the “clock” icon.

   • Groups and private chats were unresponsive.

   • Web version also failed to connect.

   • The issue impacted millions of users worldwide.

3. Cause:

   • Internal server configuration changes triggered a major communication breakdown between WhatsApp servers.

   • Load balancing failed due to improper update deployment.

4. Maintenance Actions Taken:

   • The engineering team rolled back the latest deployment.

   • Reconfigured server communication modules.

- Conducted an emergency round of system health checks and network traffic balancing.

5. Outcome:
   - Services were gradually restored within 2.5 hours.
   - Meta issued a public apology and promised enhanced monitoring.
   - Internal deployment processes were revised to include stricter testing phases.

Q. Create a DFD for a hospital management system.

Ans-

Tasks to Perform:

1. Identify key processes and external entities in the hospital system.

2. Create a Level 0 DFD (Context Diagram).

3. Expand into a Level 1 DFD showing detailed interactions.

Level 0 - DFD (Context Diagram) :

External Entities:

- Patient
- Doctor
- Receptionist
- Admin Processes:
- Hospital Management System Data Flows:
- Patient provides registration details
- Doctor provides diagnosis
- Receptionist schedules appointments
- Admin manages records

Code---

[Patient] → (HMS) ← [Doctor]

[Receptionist] → (Hospital Management System) ← [Admin]

Level 1 - DFD (Detailed Process Breakdown) :

Processes:

1. Patient Registration
2. Appointment Scheduling
3. Medical Diagnosis
4. Billing and Discharge
5. Report Generation Data Stores:

- Patient Records
- Appointment Database
- Billing Info
- Medical History Example Flow:

scss

code—

[Patient] → (1. Patient Registration) → [Patient Records]

[Receptionist] → (2. Appointment Scheduling) → [Appointment DB]

[Doctor] → (3. Medical Diagnosis) ↔ [Medical History] (HMS)

→ (4. Billing & Discharge) → [Billing Info]

Q. Build a simple desktop calculator application using a GUI library.

Ans-

Tasks to Perform:

1. Design a calculator GUI with buttons for digits (0-9), operations (+, –, ×, ÷), clear, and equals.
2. Implement logic to handle button clicks and perform operations.
3. Display results and handle invalid inputs (e.g., division by zero).

Suggested Tech Stack :

- Language: Python (Recommended)
- GUI Library: Tkinter Design Notes :
- Use frames to organize buttons into rows
- Validate inputs and handle edge cases
- UI should be responsive and user-friendly Tools Required :
- Python 3.x
- Tkinter (comes built-in with Python)
- Code editor (VS Code / PyCharm / IDLE) Learning Outcome :

After completing this lab, students will:

- Understand GUI event handling and layout design
- Be able to create interactive desktop apps
- Learn how to integrate logic with GUI controls

Q. Draw a flowchart representing the logic of a basic online registration system.

Ans-

Tasks to Perform :

1. Identify the sequence of steps a user follows in an online registration form.
2. Define decision points such as validation and duplication check.
3. Draw a flowchart using standard flowchart symbols.

Flowchart Logic Description :

1. Start

2. Display Registration Form

3. User Inputs Details

4. Validate Required Fields **o** If Invalid → Show Error → Go to Step 3 **o** If Valid → Proceed

5. Check If User Already Exists **o** If Yes → Show "User Exists" Message → End **o** If No → Proceed

6. Store User Data in Database

7. Show Registration Success Message

8. End

Flowchart (Text Representation):

code

```
[Start]
↓
[Display Registration Form]
↓
[User Enters Details]
↓
[Validate Inputs]
↓
┌──────────────┐
| Inputs Valid? |
└──────┬───────┘
|Yes No
↓  ↙
[Check If User Exists] ← [Show Error]
```

```
↓
┌──────────────┐
│ User Exists? │
└──────┬───────┘
│Yes No
↓ ↙
[Show Exists Msg] ← [Store in Database]
↓
[End] ← [Show Success]
```