Module 1 – SE -Overview of IT Industry

(1)program:

**1. In Computing / Technology**

A **program** is a set of instructions written in a programming language that a computer can execute to perform a specific task.

* Example: A calculator app, a video game, or a word processor.
* Programming languages used: Python, JavaScript, C++, Java, etc.

**2. In Broadcasting (TV/Radio)**

A **program** is a scheduled show or segment.

* Example: A news program, a comedy show, or a sports broadcast.

**3. In Events / Planning**

A **program** can also refer to an organized plan or schedule of activities.

* Example: A wedding program, a conference agenda, or a school curriculum.

(2) Explain in your own words what a program is and how it functions.

**program** is like a recipe for a computer. Just like a recipe tells a cook step-by-step how to make a meal, a program tells a computer step-by-step what to do.

For example, if you want a computer to show your name on the screen, you write a program that says, "Hey computer, show this name!" The computer follows those step

FUNCTIONS:

**1. You write the program**

* This is like writing a list of instructions (called **code**).
* Example:

python

CopyEdit

print("Hello!")

**2. You run the program**

* You tell the computer to start doing what the instructions say.

**3. The computer reads your instructions**

* The computer goes through your code **line by line**.
* It follows the steps **exactly** how you wrote them.

**4. The computer does the job**

* For example, if the instruction says to show "Hello!", the computer will display that on the screen.

**5. It finishes when all steps are done**

* When there are no more instructions, the program ends.

(3) What is Programming?

**Programming** is the way we **talk to a computer** and tell it what to do.

Just like you give instructions to a person (like "turn on the light" or "make tea"), you give instructions to a computer using **code**.

We use **programming languages** like:

* Python
* JavaScript
* C
* Java

(4) What are the key steps involved in the programming process?

 **Understand the Problem**  
First, you need to know exactly what you want the computer to do. Think clearly about the problem you want to solve.

 **Plan the Solution**  
Make a simple plan or list of steps that will solve the problem. This is like making a recipe before cooking.

 **Write the Code**  
Use a programming language (like Python or JavaScript) to write the instructions for the computer based on your plan.

 **Test the Program**  
Run your program to check if it works correctly. See if it does what you want it to do.

 **Fix Errors (Debugging)**  
If the program doesn’t work properly or shows mistakes, find and fix those problems.

 **Improve and Maintain**  
After it works, you can make the program better or add new features. You also keep fixing it if something stops working later.

(6) What are the main differences between high-level and low-level programming languages?

| **Feature** | **High-Level Languages** | **Low-Level Languages** |
| --- | --- | --- |
| **What it looks like** | Uses words and symbols similar to English | Uses numbers and simple codes (0s and 1s) |
| **Easy to read/write** | Yes, easy for humans to understand | No, very hard for humans to read and write |
| **Close to computer** | Far from the hardware | Very close to the hardware (CPU) |
| **Portability** | Can run on many types of computers easily | Works only on a specific type of computer |
| **Speed** | Slower because it needs to be translated | Faster because computer understands it directly |
| **Examples** | Python, Java, C++, JavaScript | Machine language, Assembly language |
| **Use** | Writing apps, websites, games, etc. | Writing system software, drivers, hardware control |

(7) World Wide Web & How Internet Works.

**What is the World Wide Web (WWW)?**

* The **World Wide Web** is a huge collection of **webpages** that you can see on the internet.
* These webpages have text, pictures, videos, and links to other pages.
* When you open your browser (like Chrome or Firefox) and type a website address (like [www.google.com](http://www.google.com)), you are using the World Wide Web.
* Think of the Web as a giant library full of information that you can explore.

**What is the Internet?**

* The **Internet** is the big network that connects millions of computers and devices all over the world.
* It lets these computers talk to each other and share information.
* The Web (World Wide Web) is just one of the things that works on the Internet — like email, video calls, online games, and more.

**⚙️ How Does the Internet Work?**

1. **Your Device Connects**  
   Your computer, phone, or tablet connects to the Internet through a company like your internet provider (ISP).
2. **Requesting Information**  
   When you type a website address or click a link, your device sends a request over the Internet to the server that stores that website.
3. **Servers**  
   Servers are special computers that store websites and send their information back when asked.
4. **Sending Data Back**  
   The server sends the website data back to your device over the Internet.
5. **Displaying the Webpage**  
   Your browser takes the data and shows the webpage on your screen so you can read or watch it.

(8) : Research and create a diagram of how data is transmitted from a client to a serverover the internet.

[Client Device]

|

| (1) Request (broken into packets)

|

[Your Router / Modem]

|

| (2) Packets travel through Internet

|

[Internet Routers & Switches] ----> [Multiple networks & paths]

|

| (3) Packets arrive at

|

[Server]

|

| (4) Server processes request and sends response packets

|

[Internet Routers & Switches]

|

| (5) Packets travel back

|

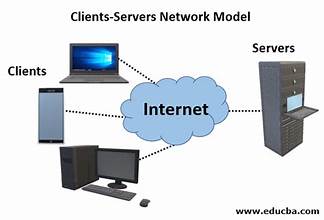
[Your Router / Modem]

|

| (6) Response packets received and reassembled

|

[Client Device displays the content]



(9) Describe the roles of the client and server in web communication.

**1. Client**

* The **client** is usually your device (computer, phone, tablet) or the web browser (like Chrome, Firefox).
* Its job is to **ask for information** or services from the server.
* For example, when you type a website address or click a link, your client sends a request to the server.
* It **displays the information** it gets back from the server so you can see the website, video, or app.

**2. Server**

* The **server** is a powerful computer that stores websites, apps, or data.
* Its job is to **listen for requests** from clients and then **send back the requested information**.
* For example, when a client asks for a webpage, the server finds it and sends the webpage data back to the client.
* It keeps running 24/7 so clients can always connect to it.

(10) Network Layers on Client and Server.

**What are Network Layers?**

* Network layers are like steps or levels that help computers (clients and servers) communicate over the internet.
* Each layer has a special job, and together they make sure data travels smoothly from one device to another.

**🏢 Network Layers on Client and Server (Basic Model)**

Both client and server use the same layers to send and receive data.

Here are the main layers (based on the OSI model simplified):

1. **Application Layer**
   * This is where programs like your browser or web server work.
   * It creates and reads the data you want to send or receive (like web pages, emails).
2. **Transport Layer**
   * It breaks data into small pieces called packets.
   * It also makes sure data arrives correctly and in order.
3. **Network Layer**
   * It finds the best path for data to travel across networks (like a GPS for data).
   * It handles addressing (like IP addresses).
4. **Data Link Layer**
   * It makes sure data is sent correctly over the physical connection (like your Wi-Fi or Ethernet cable).
5. **Physical Layer**
   * This is the actual hardware — wires, cables, wireless signals — that carry the data.

(11) Design a simple HTTP client-server communication in any language.(c language)

* HTTP SERVER IN C:

// simple\_http\_server.c

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8080

#define BUFFER\_SIZE 1024

int main() {

int server\_fd, new\_socket;

struct sockaddr\_in address;

int addr\_len = sizeof(address);

char buffer[BUFFER\_SIZE] = {0};

// Create socket file descriptor

if ((server\_fd = socket(AF\_INET, SOCK\_STREAM, 0)) == 0) {

perror("socket failed");

exit(EXIT\_FAILURE);

}

// Prepare the sockaddr\_in structure

address.sin\_family = AF\_INET;

address.sin\_addr.s\_addr = INADDR\_ANY; // Listen on all interfaces

address.sin\_port = htons(PORT);

// Bind the socket to the port

if (bind(server\_fd, (struct sockaddr \*)&address, sizeof(address)) < 0) {

perror("bind failed");

exit(EXIT\_FAILURE);

}

// Listen for incoming connections

if (listen(server\_fd, 3) < 0) {

perror("listen");

exit(EXIT\_FAILURE);

}

printf("Server listening on port %d...\n", PORT);

while(1) {

// Accept incoming connection

if ((new\_socket = accept(server\_fd, (struct sockaddr \*)&address, (socklen\_t\*)&addr\_len)) < 0) {

perror("accept");

exit(EXIT\_FAILURE);

}

// Read client request

read(new\_socket, buffer, BUFFER\_SIZE);

printf("Received request:\n%s\n", buffer);

// Prepare HTTP response

char \*response = "HTTP/1.1 200 OK\nContent-Type: text/html\n\n"

"<html><body><h1>Hello from C HTTP Server!</h1></body></html>";

// Send response

write(new\_socket, response, strlen(response));

// Close connection

close(new\_socket);

}

return 0;

}

* HTTP CLINT IN C

// simple\_http\_client.c

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <unistd.h>

#include <arpa/inet.h>

#define PORT 8080

#define BUFFER\_SIZE 1024

int main() {

int sock = 0;

struct sockaddr\_in serv\_addr;

char buffer[BUFFER\_SIZE] = {0};

char \*http\_get = "GET / HTTP/1.1\r\nHost: localhost\r\n\r\n";

// Create socket

if ((sock = socket(AF\_INET, SOCK\_STREAM, 0)) < 0) {

printf("\n Socket creation error \n");

return -1;

}

serv\_addr.sin\_family = AF\_INET;

serv\_addr.sin\_port = htons(PORT);

// Convert IPv4 and IPv6 addresses from text to binary form

if(inet\_pton(AF\_INET, "127.0.0.1", &serv\_addr.sin\_addr) <= 0) {

printf("\nInvalid address/ Address not supported \n");

return -1;

}

// Connect to the server

if (connect(sock, (struct sockaddr \*)&serv\_addr, sizeof(serv\_addr)) < 0) {

printf("\nConnection Failed \n");

return -1;

}

// Send HTTP GET request

send(sock, http\_get, strlen(http\_get), 0);

printf("HTTP GET request sent\n");

// Read the server response

int valread = read(sock, buffer, BUFFER\_SIZE);

printf("Server response:\n%s\n", buffer);

// Close the socket

close(sock);

return 0;

}

(12) Explain the function of the TCP/IP model and its layers.

**What is the TCP/IP Model?**

* The **TCP/IP model** is like a set of rules that help computers communicate over the Internet.
* It breaks down communication into smaller parts called **layers**, where each layer has a special job.
* It makes sure data is sent, received, and understood correctly between devices like your computer and a website’s server.

**📚 The 4 Main Layers of the TCP/IP Model:**

1. **Application Layer**
   * This is the top layer where you use programs like your web browser, email, or apps.
   * It prepares the data you want to send (like a webpage request) and also understands the data you receive.
2. **Transport Layer**
   * This layer breaks the data into smaller pieces called **packets** for sending.
   * It makes sure these packets arrive safely and in the right order.
   * TCP (Transmission Control Protocol) works here to make communication reliable.
3. **Internet Layer**
   * This layer decides how to send packets across networks to reach the right destination.
   * It uses addresses (like IP addresses) to guide the data to the correct device.
4. **Network Access Layer (Link Layer)**
   * This is the bottom layer that deals with the actual hardware (Wi-Fi, cables) that sends and receives the data on your local network.

z

(13) Explain Client Server Communication

**What is Client-Server Communication?**

* It’s the way your device (the **client**) and another powerful computer (the **server**) talk to each other over the internet or a network.
* The **client** asks for something (like a webpage, file, or video).
* The **server** listens for these requests, finds the information, and sends it back to the client.

**How Does It Work? (Step-by-step)**

1. **Client sends a request**
   * For example, you open your browser and type a website address. Your browser (client) sends a request to the server asking for that website.
2. **Server receives the request**
   * The server gets the request and looks for the webpage or data you asked for.
3. **Server sends back a response**
   * The server sends the webpage, image, or video back to your browser.
4. **Client shows the information**
   * Your browser receives the data and displays the website or plays the video for you.

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

**1. Broadband (DSL / Cable)**

**Description:**  
Broadband is a high-speed internet connection that includes DSL (uses phone lines) and Cable (uses TV cables).

**✅ Pros:**

* Faster than dial-up
* Reliable for home use
* Available in most areas
* Supports gaming

**❌ Cons:**

* Speed may vary depending on location
* DSL is slower than other modern options

**🌐 2. Fiber-Optic Internet**

**Description:**  
Uses light signals through fiber-glass cables. It’s the fastest type of internet connection.

**✅ Pros:**

* Extremely high speed
* Very reliable and stable
* Great for gaming, video calls, large file downloads

**❌ Cons:**

* Expensive setup
* Not available everywhere (mostly in cities)

**🌐 3. Satellite Internet**

**Description:**  
Connects via satellites orbiting the Earth. Used in remote and rural areas.

**✅ Pros:**

* Works in remote areas
* No need for cables or landlines

**❌ Cons:**

* Slower speede
* Affected by weather (rain, snow)
* More expensive

**🌐 4. Mobile Internet (3G, 4G, 5G)**

**Description:**  
Internet through mobile networks using SIM cards. Used in phones or portable hotspots.

**✅ Pros:**

* Portable and wireless
* 4G and 5G offer good speeds
* Good for travel and outdoor use

**❌ Cons:**

* Depends on signal strength
* Data limits or extra charges
* Can be unstable in poor network areas

(15) How does broadband differ from fiber-optic internet?

| **Feature** | **Broadband (DSL/Cable)** | **Fiber-Optic Internet** |
| --- | --- | --- |
| Speed | Medium | Very fast |
| Uses | Phone/TV cables | Glass fiber cables |
| Signal Type | Electric | Light |
| Availability | Widely available | Limited (mostly cities) |
| Cost | Cheaper | Slightly more expensive |
| Best for | Normal use | Heavy use (gaming, streaming) |

(16)difference between http and https:

| **Feature** | **HTTP (HyperText Transfer Protocol)** | **HTTPS (HTTP Secure)** |
| --- | --- | --- |
| **Meaning** | Protocol to transfer web pages | Secure version of HTTP |
| **Security** | Not secure – data is sent **without encryption** | Secure – data is **encrypted** using SSL/TLS |
| **Data Protection** | Data can be intercepted or read by others | Data is encrypted, so it’s protected from hackers |
| **URL Starts With** | http:// | https:// |
| **Use Case** | Browsing non-sensitive websites | Browsing websites with sensitive info like passwords, payments |
| **Speed** | Slightly faster since no encryption | Slightly slower due to encryption process |
| **Trust Indicator** | No special icon in browser | Shows a padlock icon in browser address bar |
| **Port Used** | Default port 80 | Default port 443 |

(17) Identify and explain three common application security vulnerabilities. Suggest possible solutions.

**1. SQL Injection (SQLi)**

**What it is:**

* This happens when an attacker tricks the application into running harmful database commands.
* For example, by typing something dangerous into a login box, they can get access to private data like user info or passwords.

**Example:**  
A user enters:  
' OR '1'='1  
This might fool the system into logging them in without a real password.

**Simple Solution:**

* Use **Prepared Statements** or **Parameterized Queries** to safely handle data.
* Think of it like putting data into locked boxes before sending it to the database.

**2. Cross-Site Scripting (XSS)**

**What it is:**

* This allows attackers to insert malicious scripts (like JavaScript) into websites that other users visit.
* The script might steal information, like login cookies or personal data.

**Example:**  
If a website shows user comments without filtering, a hacker could post:

html

CopyEdit

<script>alert('Hacked!');</script>

Then, others who see that comment get the alert (or worse, get hacked).

**Simple Solution:**

* Always **sanitize** (clean) and **escape** user input before displaying it.
* Use libraries or frameworks that automatically handle this, like React or Angular.

**3. Broken Authentication**

**What it is:**

* This means the system does a poor job of checking who the user is.
* Attackers might guess passwords, reuse stolen credentials, or exploit weak login systems.

**Example:**  
If a website allows unlimited password guesses or uses simple passwords, it’s easy to break in.

**Simple Solution:**

* Use **strong passwords** and enforce **multi-factor authentication (MFA)**.
* Limit login attempts and store passwords securely (using hashing like bcrypt).

(18) What is the role of encryption in securing applications Software Applications and Its Types.

**1. Role of Encryption in Securing Applications**

**Encryption** means turning readable data into a secret code so that only authorized people can read it.

**Why is it important?**

* **Protects sensitive data:** Like passwords, credit card numbers, personal info.
* **Keeps data safe in transit:** When data moves over the internet (like in HTTPS), encryption stops hackers from stealing it.
* **Secures stored data:** Even if someone steals the database, encrypted data is unreadable without the key.
* **Builds user trust:** People feel safer using apps that protect their information.

**How it works?**

* Data is converted into unreadable form (ciphertext).
* Only someone with the secret key can decrypt it back to readable form.

**2. Software Applications and Its Types**

**Software Application** is a program designed to help users perform specific tasks.

**Types of Software Applications**

| **Type** | **Description** | **Examples** |
| --- | --- | --- |
| **Desktop Applications** | Installed on a computer, used locally | Microsoft Word, Photoshop, VLC Media Player |
| **Web Applications** | Run in web browsers over the internet | Gmail, Facebook, Google Docs |
| **Mobile Applications** | Designed for smartphones/tablets | WhatsApp, Instagram, Uber |
| **Enterprise Applications** | Large scale software for businesses | SAP, Oracle ERP, Salesforce |
| **Embedded Software** | Built into hardware devices | Software in smart TVs, cars, microwaves |

(19) Identify and classify 5 applications you use daily as either system software or application software.

| **Application** | **Type** | **Explanation** |
| --- | --- | --- |
| **Google Chrome (or any browser)** | Application Software | Used for browsing the internet. Helps users interact with websites. |
| **Microsoft Word (or Google Docs)** | Application Software | Used to create and edit documents. Built for specific tasks. |
| **Windows / macOS / Linux** | System Software | The operating system that runs the computer and manages all other programs. |
| **Antivirus Software** | System Software | Protects the system from viruses and malware. Works in the background. |
| **WhatsApp (or any messaging app)** | Application Software | Allows users to send messages and media. Meant for direct user interaction. |

(20) What is the difference between system software and application software?

**System Software**

**What it is:**

* It's the software that **runs the computer** and **helps it work properly**.
* It manages the hardware (like CPU, memory, etc.) and lets other software run.

**Examples:**

* Windows, macOS, Linux (Operating Systems)
* Device drivers
* Antivirus programs

**Think of it like:**

* The **foundation** or **engine** of a car — it makes everything else work.

**📱 Application Software**

**What it is:**

* It's the software that helps **you do specific tasks**.
* You use it to write, browse, play games, send messages, etc.

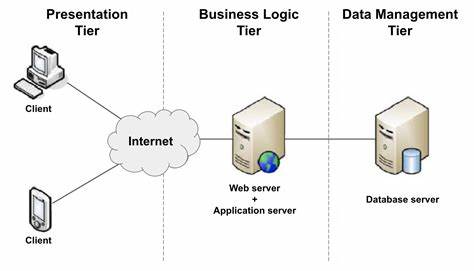
**Examples:**

* Microsoft Word (for writing)
* Google Chrome (for browsing)
* WhatsApp (for chatting)
* Spotify (for listening to music)

**Think of it like:**

* The **features inside the car**, like the radio or GPS — things you actually use and interact with.

(21) : Design a basic three-tier software architecture diagram for a web application.



(22) What is the significance of modularity in software architecture?

**Modularity** means breaking a software system into **smaller, independent parts** called **modules**, where each module handles a specific part of the functionality.

(23)

**1. Presentation Layer (User Interface)**

**Functionality:**

* Displays the book catalog to users in a user-friendly way.
* Shows details such as book title, author, price, and reviews.
* Allows users to search for books, add items to the cart, and place orders.
* Handles user inputs like clicking buttons or filling forms.

**2. Business Logic Layer (Application Layer)**

**Functionality:**

* Implements the rules and workflows of the bookstore.
* Processes user requests from the Presentation Layer.
* Validates data, such as checking if the book is in stock before adding to cart.
* Calculates total price including taxes and discounts.
* Handles user authentication and order processing.

**3. Data Access Layer**

**Functionality:**

* Manages communication with the database.
* Retrieves book information, user details, and order history from the database.
* Updates database when new orders are placed or stock levels change.

(24) Why are layers important in software architecture?

**Separation of Responsibility:**  
Each layer is responsible for a specific task.  
For example, one layer interacts with the user, another layer handles the logic, and the third layer stores the data.

**Maintainability:**  
When a bug or error occurs, you can easily fix it in that specific layer without affecting the other layers.

**Easy to Test:**  
You can test each layer separately, which helps to find errors quickly.

**Better Teamwork:**  
Different developers can work on different layers without blocking each other.

**Reusability:**  
Logic or functions written in one layer can be reused in other projects as well.

**Organized & Flexible System:**  
The software becomes easier to understand and is flexible to change.

(25)

### Types of Software Environments

1. **Development Environment**

* Where programmers write and create the software.
* Includes tools like code editors (e.g., VS Code), compilers, and debuggers.
* Used for building new features and fixing bugs.

1. **Testing Environment**

* Where the software is tested to find errors or issues before release.
* Helps ensure the software works correctly and securely.
* Mimics the production environment but is safe to test without affecting real users.

1. **Production Environment**

* The live environment where the software is available to end-users.
* Must be stable, secure, and always available.
* Changes here affect real users, so updates are done carefully.

### How to Set Up a Basic Environment in a Virtual Machine (VM)

A **virtual machine** lets you run a separate computer inside your real computer — perfect for creating isolated software environments.

#### Steps to Set Up:

1. **Install Virtual Machine Software:**  
   Download and install software like [VirtualBox](https://www.virtualbox.org/) or VMware.
2. **Create a New Virtual Machine:**
   * Open the VM software and create a new VM.
   * Choose an operating system (e.g., Windows, Linux).
3. **Install the Operating System:**
   * Use an ISO file to install the OS inside the VM.
   * Follow the installation steps as on a normal computer.
4. **Set Up Development Tools:**
   * Install software like code editors, databases, or any other tools needed.
   * This will be your **development environment** inside the VM.
5. **Optionally Set Up Testing Environment:**
   * Create another VM or use the same VM with separate configurations.
   * Install testing tools like Selenium, Postman, or testing frameworks.
6. **Use the VM as Your Software Environment:**
   * You can work inside this VM without affecting your main computer.
   * Safe to try new software, test bugs, or deploy small apps.

(26) Explain the importance of a development environment in software production.

1. **Safe Space to Build Software:**  
   The development environment is like a playground for programmers where they can write and test their code without affecting the real users or live system.
2. **Helps Find and Fix Bugs Early:**  
   Developers can try out new features and find mistakes early in the development stage before the software is released.
3. **Allows Collaboration:**  
   Multiple developers can work together, share code, and test their work before it goes live.
4. **Speeds Up Development:**  
   With the right tools and settings, developers can write, test, and improve code faster.
5. **Keeps Production Stable:**  
   By using a separate environment for development, the live system (production) remains stable and uninterrupted.

(27)

| **Aspect** | **Source Code** | **Machine Code** |
| --- | --- | --- |
| **What is it?** | Human-readable instructions written by programmers in a programming language (like Python, Java, C++). | Low-level code made up of binary numbers (0s and 1s) that the computer's processor understands directly. |
| **Readability** | Easy for humans to read and understand. | Hard for humans to read; meant for computers only. |
| **Purpose** | To write and develop software. | To execute the software on the computer hardware. |
| **How created?** | Written by programmers. | Created by translating (compiling or interpreting) the source code. |
| **Example** | print("Hello World") | 101010101001001... (binary code) |

(28)

### Importance of Version Control:

1. **Tracks Changes Over Time**
   * Every time you change the code, it's saved.
   * You can go back to any previous version if something goes wrong.
2. **Team Collaboration**
   * Many developers can work on the same project at the same time without overwriting each other’s code.
3. **Error Recovery**
   * If a new change breaks the project, you can easily undo it and return to the working version.
4. **Experiment Safely**
   * You can try out new features in a separate branch without affecting the main project.
5. **Clear History**
   * You can see who made what change and why (with messages), which helps during debugging or updates.
6. **Better Project Management**
   * Helps manage big projects by organizing code versions, releases, and updates clearly.

(29) : What are the benefits of using Github for students?

1. **Learn Real-World Tools**
   * GitHub is used by professional developers, so students learn how real-world projects are managed.
2. **Free Student Pack**
   * GitHub gives students access to many **free tools, software, and courses** that normally cost money.
3. **Build a Portfolio**
   * You can store your coding projects and share them with teachers or future employers as your **digital resume**.
4. **Work in Teams**
   * Students can work together on the same project, which builds teamwork and collaboration skills.
5. **Track Your Progress**
   * GitHub keeps a history of all your code changes, so you can see how your skills improve over time.
6. **Open Source Learning**
   * You can explore and contribute to real open-source projects and learn from others' code.
7. **Easy Project Sharing**
   * Share your code with classmates or teachers easily with just a link.
8. **Get Noticed by Companies**
   * Recruiters often look at GitHub profiles when hiring developers.

(30)types of software

### 1. ****System Software****

This type of software helps the computer to run and manage itself.  
It acts like the "manager" of the computer.

**Examples:**

* Operating System (Windows, Linux, macOS)
* Device Drivers
* Utility Programs (Antivirus, Disk Cleanup)

### 2. ****Application Software****

This software helps users do specific tasks.  
It runs **on top** of system software.

**Examples:**

* MS Word (for writing)
* Web Browsers (Chrome, Firefox)
* Games
* Media Players (VLC)

(31) Create a list of software you use regularly and classify them into the followingcategories: system, application, and utility software.

### ****1. System Software****

This category includes the foundational software that manages hardware and enables other software to run.

* **Windows / macOS / Linux** – Operating Systems
* **Device Drivers** – Such as printer or graphics drivers
* **Firmware** – BIOS/UEFI software for basic hardware control
* **System Libraries** – Like .dll files in Windows or .so files in Linux

### ****2. Application Software****

These are end-user programs designed to perform specific tasks.

* **Microsoft Word / Google Docs** – Word processing
* **Excel / Google Sheets** – Spreadsheets
* **Chrome / Firefox / Safari** – Web browsers
* **Zoom / Microsoft Teams** – Video conferencing
* **Adobe Photoshop / GIMP** – Image editing
* **Spotify / VLC Media Player** – Media playback
* **Slack / Discord** – Communication
* **Visual Studio Code / PyCharm** – Code editors/IDEs

### ****3. Utility Software****

Utility programs help manage, maintain, and control computer resources.

* **Antivirus Software (e.g., Windows Defender, Avast)** – Security
* **CCleaner** – System cleanup
* **7-Zip / WinRAR** – File compression and extraction
* **Task Manager / Activity Monitor** – System monitoring
* **Backup and Restore tools (e.g., Time Machine, Acronis)** – Data backup
* **Disk Management Tools (e.g., Disk Utility, Partition Manager)** – Disk handling
* **Clipboard Manager / Snipping Tool** – Productivity utilities

(32) What are the differences between open-source and proprietary software?

| **Point** | **Open-Source Software** | **Proprietary Software** |
| --- | --- | --- |
| **Meaning** | Free to use, see, and change the code | Owned by a company, code is hidden |
| **Cost** | Mostly free | Usually paid or licensed |
| **Change Allowed?** | Yes, anyone can modify the software | No, only the company can make changes |
| **Support** | Community helps (forums, online groups) | Company gives official support |
| **Examples** | Linux, VLC Player, Mozilla Firefox, LibreOffice | Windows, MS Office, Adobe Photoshop, macOS |

(33) How does GIT improve collaboration in a software development team?

 **Letting everyone work on their own copy** of the project

 **Saving every change**, so nothing is lost

 **Merging everyone's work** into one final version

 **Tracking who did what and when**

 **Avoiding mistakes** by warning if two people change the same thing

(34) : Write a report on the various types of application software and how they improve productivity.

* **Word Processors:** Like Microsoft Word, used to write documents.
* **Spreadsheets:** Like Excel, used to work with numbers and do calculations.
* **Presentation Software:** Like PowerPoint, used to make slideshows.
* **Database Software:** Used to store and organize large amounts of data.
* **Communication Software:** Like WhatsApp or Zoom, used to talk and work with others.
* **Graphic Design Software:** Like Photoshop, used to create and edit pictures.
* **Word Processing Software**
  + Helps you write and edit documents quickly
  + Offers tools like spell check and templates, saving time
  + Makes your work look professional without much effort
* **Spreadsheet Software**
  + Automatically does calculations for you
  + Organizes data clearly so you can understand it fast
  + Creates charts and graphs that help in decision-making
* **Presentation Software**
  + Lets you make attractive slides quickly
  + Uses templates so you don’t have to start from scratch
  + Helps explain ideas clearly in meetings or classes
* **Database Software**
  + Stores lots of information in an organized way
  + Helps you find information quickly
  + Reduces mistakes by managing data accurately
* **Communication Software**
  + Allows instant chatting and video calls
  + Helps teams work together even if they are far apart
  + Speeds up decision-making by improving communication
* **Graphic Design Software**
  + Makes creating pictures and designs faster and easier
  + Helps produce professional-quality work
  + Saves time on marketing and creative projects

(35) What is the role of application software in businesses?

### What is the Role of Application Software in Businesses?

Application software helps businesses **do their work better and faster**. It is like a tool that makes different jobs easier to complete.

### How Application Software Helps Businesses:

1. **Helps with Daily Tasks:**  
   Businesses use software like word processors and spreadsheets to write letters, create bills, and manage money easily.
2. **Improves Communication:**  
   Tools like email, chat apps, and video calls help employees talk and share information quickly, even if they are far away.
3. **Organizes Information:**  
   Database software keeps all the business information safe and organized so it’s easy to find when needed.
4. **Helps in Decision Making:**  
   Businesses use software to analyze data and create reports, which helps managers make smart choices.
5. **Increases Productivity:**  
   By automating boring tasks, application software saves time and lets employees focus on important work.
6. **Supports Marketing and Sales:**  
   Graphic design and presentation software help businesses create advertisements and presentations to attract customers.

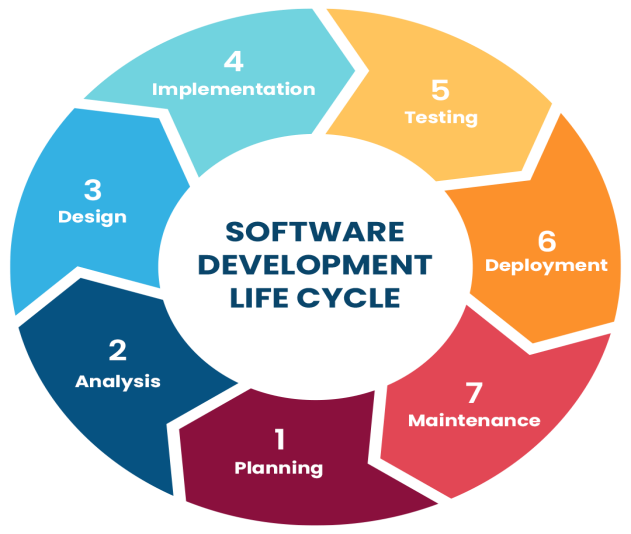
(36) Software Development Process

The **Software Development Process** is a step-by-step way to create software (computer programs). It helps developers plan, build, test, and deliver software that works well and meets users’ needs.

### Main Steps in the Software Development Process:

1. **Requirement Gathering**
   * Understand what the users want the software to do.
   * Talk to users and write down their needs.
2. **Planning**
   * Decide how to build the software.
   * Make a plan with tasks and timelines.
3. **Design**
   * Create a blueprint of the software.
   * Decide how it will look and work.
4. **Coding (Development)**
   * Write the actual program code.
   * Developers build the software using programming languages.
5. **Testing**
   * Check if the software works correctly.
   * Find and fix bugs (mistakes).
6. **Deployment**
   * Install the software so users can start using it.
7. **Maintenance**
   * Fix problems found later.
   * Update the software to add new features or improve it.

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



(38) What are the main stages of the software development process?

1. **Requirement Gathering**  
   Understanding what the users need the software to do.
2. **Planning**  
   Creating a plan on how to build the software, including tasks and schedule.
3. **Design**  
   Making a blueprint for the software’s structure and appearance.
4. **Coding (Development)**  
   Writing the actual code to build the software.
5. **Testing**  
   Checking the software for errors and making sure it works correctly.
6. **Deployment**  
   Installing and releasing the software for users to use.
7. **Maintenance**  
   Fixing problems after release and updating the software as needed.

(39) Software Requirement

**software Requirements** are the detailed descriptions of what a software program should do and how it should work. They tell developers exactly what the users want from the software.

Two type:

Functional

Non functional

(40)

Library Management System Requirements

│

├── Functional Requirements

│ ├── Book Management

│ │ ├── Add Book

│ │ ├── Update Book

│ │ ├── Delete Book

│ │ └── Search Book

│ ├── User Management

│ │ ├── Register User

│ │ ├── Update User Details

│ │ └── Delete User

│ ├── Borrowing Management

│ │ ├── Borrow Book

│ │ ├── Return Book

│ │ ├── Track Borrowing

│ │ └── Prevent Overdue Borrowing

│ └── Reports

│ └── Generate Reports

│

├── Non-Functional Requirements

│ ├── Ease of Use

│ ├── Performance (Fast Response)

│ ├── Data Security

│ └── Multi-user Support

│

├── Constraints

│ ├── Single Library Branch

│ └── Local Network (No Internet Required)

│

└── Assumptions

├── Unique User IDs

├── Unique Book ISBN

└── Librarian Manages Records

(41) Why is the requirement analysis phase critical in software development?

* **Understanding What Users Want:**  
  It helps the developers know exactly what the users need from the software.
* **Avoids Mistakes:**  
  If requirements are not clear, developers might build the wrong thing, wasting time and effort.
* **Helps Plan Better:**  
  Clear requirements make it easier to plan how to build the software step-by-step.
* **Saves Time and Money:**  
  Finding problems early by understanding requirements prevents costly fixes later.
* **Improves Quality:**  
  When the software is made according to clear requirements, it works better and users are happier.

(42)

**Software Analysis** is the process of **understanding what the software should do** before we start building it.

It answers questions like:

* What is the problem?
* What does the user want?
* What features should the software have?

(43) Perform a functional analysis for an online shopping system

### ****Customer Side (User)****

Imagine you are shopping online like on Amazon or Flipkart:

1. **Create Account** – You sign up with your name, email, and password.
2. **Login/Logout** – You login to your account and logout when done.
3. **Browse Products** – You see different categories like clothes, mobiles, etc.
4. **Search Products** – You search for "shoes" and see a list of shoes.
5. **View Product Details** – You click on a shoe and see its price, size, reviews, and images.
6. **Add to Cart** – You like it and add it to your cart.
7. **Change Quantity or Remove** – You decide to buy 2 or remove it later.
8. **Checkout** – You go to your cart and click "Buy Now".
9. **Payment** – You pay using your card, UPI, or net banking.
10. **Track Order** – After placing the order, you check the delivery status.
11. **Leave Review** – After receiving the product, you rate it and write a review.

### 👨‍💼 ****Admin Side (Back-end Manager)****

Imagine someone who manages the website:

1. **Admin Login** – Admin logs in securely.
2. **Manage Products** – Add new products, change prices, or remove out-of-stock items.
3. **Manage Orders** – See all orders placed by customers and update delivery status.
4. **Manage Users** – Admin can view users, block suspicious accounts.
5. **Reports** – Admin can check how many sales happened this month.
6. **Add Offers** – Admin can add festival discounts or coupon codes.

(44) What is the role of software analysis in the development process?

Software analysis means **thinking and planning** before making the software.

It is the **first step** where we ask:

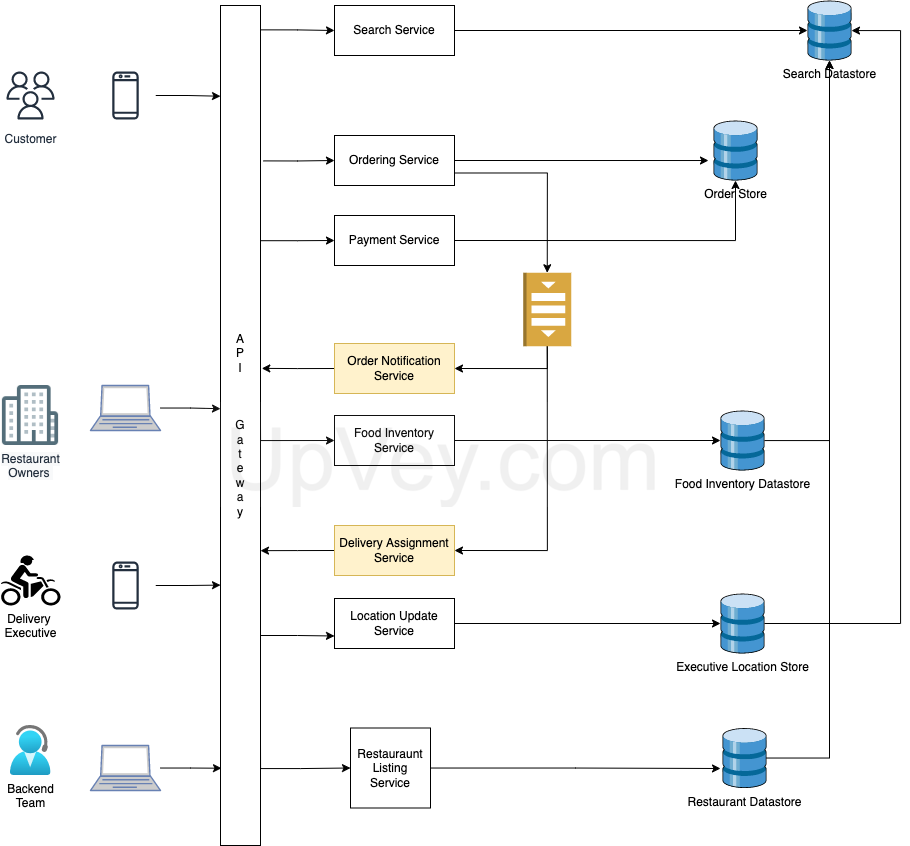
* What does the user want?
* What problem needs to be solved?
* What features should the software have?

(45) System Design

**System Design** is the process of planning and creating the overall structure of a software system.

It decides **how the software will work**, how different parts will connect, and how data will flow.

(46) Design a basic system architecture for a food delivery app.



(47) What are the key elements of system design?

 **Requirements** – What the system should do.

 **Scalability** – Can it grow with more users/data?

 **Reliability** – Does it work correctly, every time?

 **Availability** – Is it always up and running?

 **Performance** – Is it fast and responsive?

 **Security** – Is the data safe and protected?

 **Maintainability** – Is it easy to fix or update?

 **Cost** – Is it affordable to build and run?

(48)Why is software testing important?

### 1. ****Finds Bugs Early****

* Testing helps **catch errors** before the software is released.
* Fixing bugs early is **cheaper and faster**.

🧪 Example: If a calculator app gives the wrong answer, testing will catch it before users complain.

### 2. ****Improves Quality****

* Testing ensures the software works **as expected**.
* Makes sure all **features work properly**.

✅ Good testing = better performance, fewer crashes.

### 3. ****Saves Time and Money****

* Finding problems early avoids **costly fixes** later.
* Helps avoid **releasing a broken product**.

### 4. ****Ensures Security****

* Testing checks for **security issues**, like data leaks or hacking risks.
* Protects **user data** and privacy.

### 5. ****Increases Customer Trust****

* When software is tested and reliable, **users are happy**.
* Builds **confidence** in your product or company.

### 6. ****Ensures Compatibility****

* Checks that the software works on different **devices, browsers, and systems**.

📱🖥️ Testing makes sure it runs on both mobile and desktop, for example.

### 7. ****Supports Maintenance****

* Makes it easier to **update or add new features** later.
* Testing helps make sure old features still work (this is called **regression testing**).

(49) Document a real-world case where a software application required critical maintenance.

## ****Real Case: Facebook Crash – October 4, 2021****

### 😲 What happened?

* Facebook, Instagram, WhatsApp all **stopped working for 6 hours**.
* It affected **people all over the world**.

### ❌ Why it happened?

* Facebook made a **mistake in network settings** (like changing wrong wires in the internet).
* This made Facebook's servers **disappear from the internet**.
* Even **Facebook’s own tools** stopped working, so engineers had to go to the building to fix it.

### 🛠️ What they did to fix it:

* Engineers **reversed the wrong setting** manually.
* They worked on bringing back tools and services step by step.
* Facebook then updated its **safety rules** to avoid this problem in the future.

(50)What types of software maintenance are there?

## ****Types of Software Maintenance (Short)****

1. **Corrective** – Fixing bugs or errors.
2. **Adaptive** – Updating for new systems (OS, browser, etc.).
3. **Perfective** – Improving performance or adding features.
4. **Preventive** – Preventing future problems (clean-up, security).

(51)What are the key differences between web and desktop applications?

| **Feature** | **Web Application** | **Desktop Application** |
| --- | --- | --- |
| **Access** | Runs in a web browser (like Chrome, Firefox) | Installed and run on a computer |
| **Installation** | No installation needed; just open a URL | Must be downloaded and installed |
| **Platform** | Works on any device with a browser (Windows, Mac, Linux, phones) | Usually designed for specific OS (Windows, Mac, Linux) |
| **Updates** | Updated automatically by the developer | User must install updates manually |
| **Internet** | Usually needs internet to work | Can work offline without internet |
| **Performance** | May be slower due to internet and browser limits | Usually faster, can use full system resources |
| **Storage** | Data stored on remote servers (cloud) | Data stored locally on the device |
| **Security** | Depends on internet security and server protection | Depends on local device security |

(52)What are the advantages of using web applications over desktop applications?

## Advantages of Web Applications

1. **No Installation Needed**

* You just open a browser and start using it. No downloading or setup.

1. **Access Anywhere**

* Use it from any device (computer, phone, tablet) with internet.

1. **Automatic Updates**

* Updates happen behind the scenes; you always get the latest version.

1. **Easy to Maintain**

* Developers fix bugs or add features on the server; users don’t have to do anything.

1. **Cross-Platform Compatibility**

* Works on Windows, Mac, Linux, or mobile without needing separate versions.

1. **Lower Cost**

* No need to create multiple versions for different operating systems.

1. **Collaboration Made Easy**

* Multiple users can work on the same app or data online simultaneously.

(53) What role does UI/UX design play in application development?

## hat is UI/UX Design?

* **UI (User Interface):** How the app looks — buttons, colors, layout, fonts.
* **UX (User Experience):** How the app feels — how easy and enjoyable it is to use.

## 🔑 Role of UI/UX Design in Application Development

1. **Makes the App Easy to Use**

* Good UI/UX helps users find what they want quickly and easily.

1. **Improves User Satisfaction**

* A nice design keeps users happy and coming back.

1. **Reduces Errors**

* Clear design helps prevent mistakes and confusion.

1. **Increases Accessibility**

* Good UX makes sure the app works for people with different needs.

1. **Supports Brand Identity**

* UI design reflects the company’s style and values, building trust.

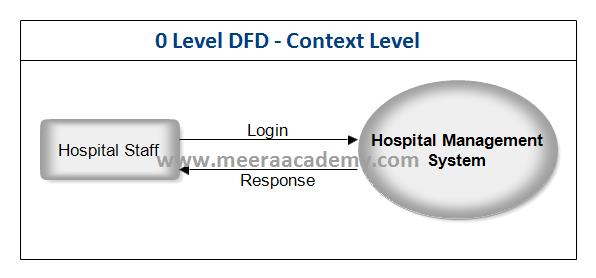
1. **Boosts Business Success**

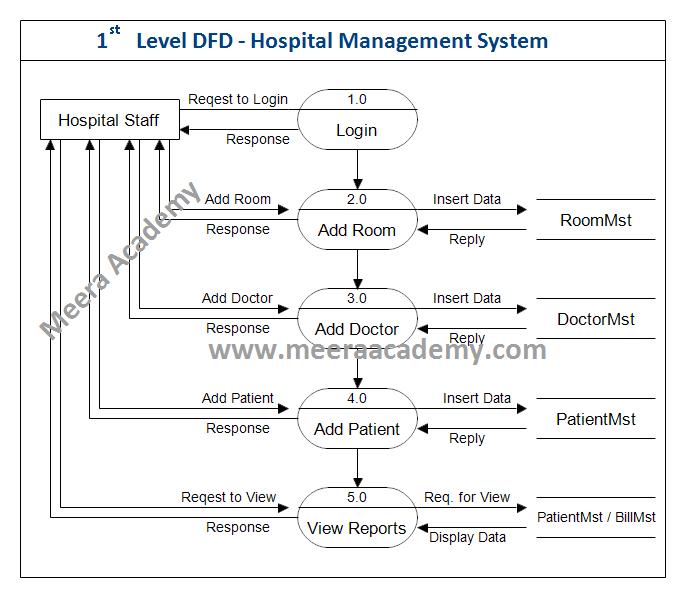
* Apps with better UX/UI get more users, better reviews, and more sales.

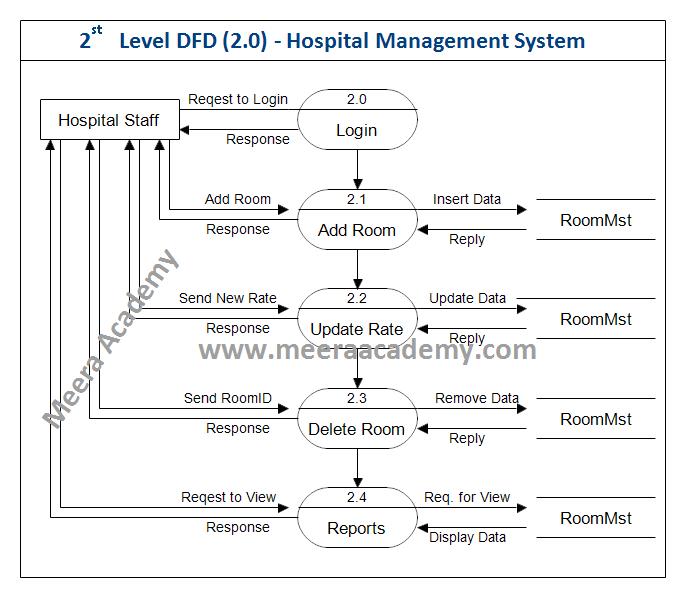
(54): What are the differences between native and hybrid mobile apps?

| **Feature** | **Native Apps** | **Hybrid Apps** |
| --- | --- | --- |
| **Development** | Built for a specific platform (iOS or Android) using platform languages (Swift, Kotlin) | Built using web technologies (HTML, CSS, JavaScript) and wrapped in a native container |
| **Performance** | Faster and smoother because they use device’s full power | Slightly slower, depends on web view inside the app |
| **Access to Device Features** | Full access to camera, GPS, notifications, etc. | Limited access, but improving with plugins |
| **User Experience** | Better, more consistent with platform design | Can feel less native or slower |
| **Development Cost** | More expensive because separate apps for iOS and Android | Cheaper; one codebase works for both platforms |
| **Updates** | Updates go through app stores | Updates can be pushed faster like web apps, but some require app store approval |
| **Examples** | Instagram, Snapchat | Twitter Lite, Gmail (some parts) |

(55)Create a DFD for a hospital management system.







(56): What is the significance of DFDs in system analysis?

### ****Visualize the System Clearly****

* DFDs show how data moves through a system in an easy-to-understand picture.

### 2. ****Identify Processes and Data Stores****

* Helps find all the important parts: where data comes from, where it goes, and where it’s stored.

### 3. ****Simplify Complex Systems****

* Breaks down complicated systems into smaller, manageable parts.

### 4. ****Communicate with Stakeholders****

* Easy for both technical teams and non-technical people (like clients) to understand the system.

### 5. ****Find Missing Requirements or Errors Early****

* By mapping data flows, analysts can spot gaps or mistakes before development starts.

### 6. ****Guide System Design and Development****

* Acts as a blueprint for developers when building or improving the system.

(57)Build a simple desktop calculator application using a GUI library

import tkinter as tk

# Create main window

root = tk.Tk()

root.title("Simple Calculator")

# Entry widget to show calculations and result

entry = tk.Entry(root, width=16, font=('Arial', 24), borderwidth=2, relief='ridge', justify='right')

entry.grid(row=0, column=0, columnspan=4)

# Function to add number or operator to entry

def click\_button(value):

current = entry.get()

entry.delete(0, tk.END)

entry.insert(0, current + value)

# Function to clear entry

def clear():

entry.delete(0, tk.END)

# Function to calculate the expression

def calculate():

try:

result = eval(entry.get())

entry.delete(0, tk.END)

entry.insert(0, str(result))

except:

entry.delete(0, tk.END)

entry.insert(0, "Error")

# Buttons layout

buttons = [

('7', 1, 0), ('8', 1, 1), ('9', 1, 2), ('/', 1, 3),

('4', 2, 0), ('5', 2, 1), ('6', 2, 2), ('\*', 2, 3),

('1', 3, 0), ('2', 3, 1), ('3', 3, 2), ('-', 3, 3),

('0', 4, 0), ('.', 4, 1), ('C', 4, 2), ('+', 4, 3),

('=', 5, 0, 4)

]

# Create buttons and place them on grid

for (text, row, col, colspan) in [(\*btn, 1) if len(btn) == 3 else btn for btn in buttons]:

if text == 'C':

action = clear

elif text == '=':

action = calculate

else:

action = lambda x=text: click\_button(x)

button = tk.Button(root, text=text, width=5, height=2, font=('Arial', 18), command=action)

button.grid(row=row, column=col, columnspan=colspan, sticky='nsew')

# Run the application

root.mainloop()

(58)What are the pros and cons of desktop applications compared to web applications?

## Desktop Applications

### Pros:

* **Faster performance:** Use full power of the computer.
* **Works offline:** Don’t need internet connection.
* **Better access to hardware:** Can use device features like printers, cameras, USB devices more easily.
* **More secure:** Data is stored locally, less risk from internet attacks.

### Cons:

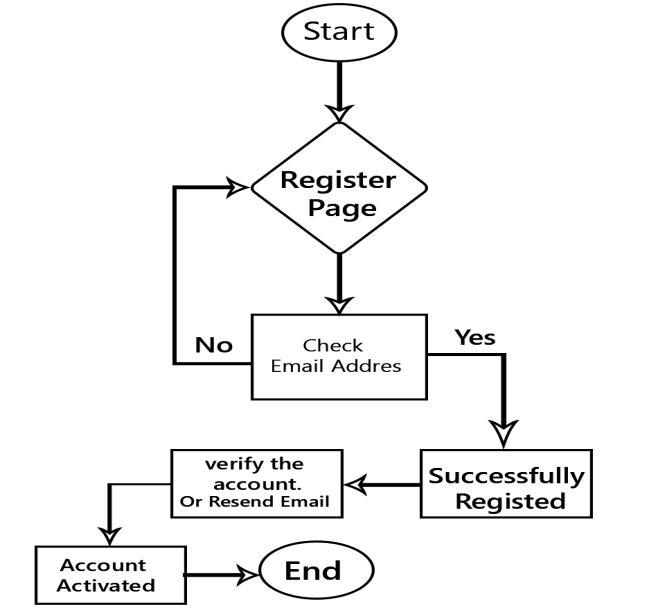
* **Installation required:** Must download and install before use.
* **Updates manual:** Users often need to update apps themselves.
* **Platform-specific:** Usually built for Windows, Mac, or Linux separately.
* **Less accessible:** Only available on the device where installed.

## 🌐 Web Applications

### Pros:

* **No installation needed:** Access via browser from any device.
* **Always up to date:** Developers update it on the server.
* **Works cross-platform:** Runs on Windows, Mac, Linux, mobile, etc.
* **Easier collaboration:** Multiple users can work online simultaneously.

(59)Draw a flowchart representing the logic of a basic online registration system



(60): How do flowcharts help in programming and system design?

### ****Visualize the Process Clearly****

* Flowcharts show each step of a program or system in pictures, making it easier to understand.

### 2. ****Plan Before Coding****

* Helps programmers organize their thoughts and plan the logic before writing any code.

### 3. ****Identify Mistakes Early****

* By mapping steps out, you can find logic errors or missing steps before building the system.

### 4. ****Simplify Complex Systems****

* Breaks down complicated tasks into smaller, manageable parts.

### 5. ****Easy Communication****

* Makes it simple to explain how a program or system works to others (team members, clients).

### 6. ****Better Documentation****

* Acts as a clear reference to understand or update the system later.