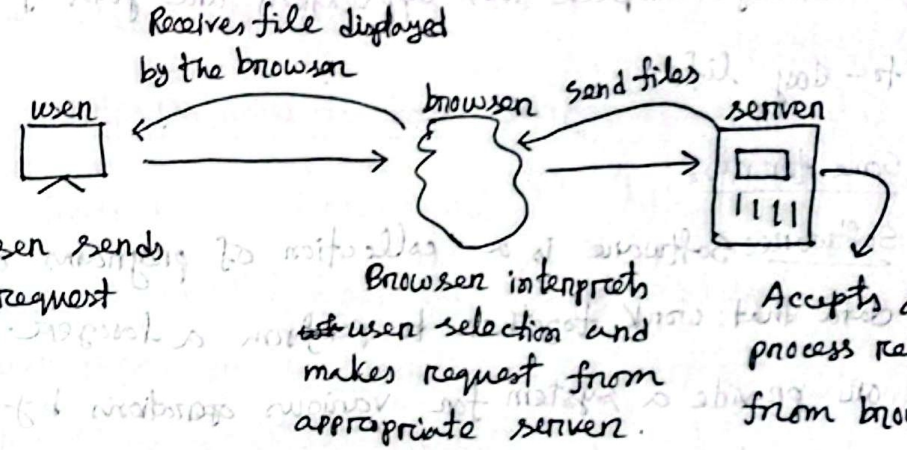
Aims →

→ To introduce the methods and techniques used in →web-based application.

→ To develop practical web-applications.



Lecture - 01

An introduction to web engineering:

History:

- The first web-site created by Tim-Berners Lee and Robert Cailliau at CERN (European Nuclear Research Center)
- Consisted of a collection of static HTML pages.
- Now has evolved into an infrastructure for the execution of complex applications.

- The web has progressively become a multi domain platform.
- Offers information delivery and application execution.
- Currently, complex web applications are part of day-to-day life.

#Some terms:-

1. Software: Software is a collection of programs and data that work together to perform a larger task or provide a system for various operations by a computer.
2. Program: A set of instructions telling a computer what to do (performs a specific task when executed by the comp.)
3. Application: A program or set of programs that is designed for an end user to perform some tasks.
4. Website: A collection of interlinked web pages that are accessible through the URL.
5. Web Pages: A collection of text, pictures, audio, video, hyperlinks, animations etc.

Internet invented — 1990
Website — 1991
AI — 1950

#URL:

URL has three parts:

https://www.facebook.com/notifications

- https:// → Protocol
- www.facebook.com → domain name
- /notifications → file name

WWW (World Wide Web): The collection of websites and web pages you can visit on the internet. Its like the "library" of the internet.

HTTP (Hyper Text Transfer Protocol): The set of rules your browser and websites follow to send and receive information. Like a "language" for the web.

Protocol: A set of rules computer follows to communicate. Think of it like traffic rules for data.

Domain Name: The easy-to-remember name of a website, like google.com, instead of using a long number (IP address).

Software Engineering:

Def: SE is an engineering discipline that is a concerned with all aspects of software production.

On the contrary, building significant software system —

→ On time
→ On budget
→ With acceptable performance.
→ Correct operations.

Program → Application → Web pages → Website.

- End User ⇒ The final person who uses the software after it has been developed, tested and delivered.
- FTP (File Transfer Protocol): Rules for sending and receiving files between computers over the internet.
- SMTP (Simple Mail Transfer Protocol): Rules for sending emails from one server to another.
- TCP/IP (Transmission Control Protocol/Internet Protocol): The main set of rules for how data is packaged, sent, and delivered across the internet.
- SSL (Secure Sockets Layer): An old method for encrypting data so its secure during transfer.
- TLS (Transport Layer Security): A newer, more secure replacement for SSL, also encrypts data.
- HTTPS (Hyper Text Transfer Protocol Secure): A secure version of HTTP that uses SSL/TLS to protect data between your browser and a website.

Q. Software Engineering Vs Web Engineering.

Aspect	SE	WE
Definition	Broad field covering the creation of all types of software systems.	Specialized branch focused on web-based applications and websites.
Scope	Desktop apps, mobile apps, operating systems, games, embedded systems, etc.	Websites, web apps, online portals, e-commerce system, etc.
Technologies used	Programming languages like, C, C++, Java, Python, etc.	Web technologies like HTML, CSS, JavaScript, PHP, databases, APIs.

Execution environment	Can be offline or online, runs on PCs, mobiles or embedded devices.	Runs online in web browsers, accessed over the internet.
Focus Areas	Software architecture, algorithm, data structures, testing, deployment.	Web design, User experience (UX), server client interaction, web security, scalability.
Example projects	MS word, Android OS, flight booking software	Facebook, Amazon, online banking portals.

#Web Application:

- A web application is a software system based on technologies and standards of the world wide web consortium (W3C) that provides web-specific resource such as content and services through a user interface, the web browser.
- A web application is an application program that is stored on a remote server and delivered over the internet through a browser interface

#Web Engineering:

- Web Engineering is the application of systematic, disciplined and quantifiable approaches (concepts, methods, techniques, tools) to cost-effective requirements analysis, design, implementation, testing, operation and maintanance of high quality web applications.
- Web engineering is also the scientific discipline concerned with the study of these approaches.

#Basic principle of web engineering:

1. Clearly defined goals and requirements [requirement analysis]
2. Systematic development of a web application in phases [design]
3. Careful planning of these phases [implementation]
4. Continuous audit of the entire development process. [testing].

Why web engineering needed?

- A 2000 survey by the Cutter Consortium identified major issues in large-scale web application development projects. The results revealed critical problems:

84% - Failure to meet business needs.
79% - Project schedule delays.
63% - Budget overruns.
53% - Lack of required functionality.
52% - Poor quality of deliverables.

Q. Web crisis: refers to the widespread failure of web projects in terms of cost, time, quality and business value.

Categories of web applications:-

→ Document centric websites (Static homepage. HTML etc.)
→ Interactive web applications (news sites).
→ Transactional " " (online banking).
→ Work-flow based " " (e-commerce application).
→ Collaborative " " (wikipedia)
→ Social web (Facebook)
→ Portal-Oriented (Google)
→ Ubiquitous use web (food panda)
→ Semantic web (recommender system)

** Product-related:
  → Content
  → Hypertext structure
  → Presentation
** Usage related
  → Natural context
  → Social context
  → Technical context.

** Development-related:

→ Team
→ Technical infrastructure.
→ Development process.
→ Integration.

** Evolution-related

→ Continous change fast pace
→ competitive pressure.

V.V.I # Web application scenario:

Divided into several layers;

- Data layer:- developer (who cares only about the backend software of distributed system) needs to understand how to best-structure the data underlying the application under development.
  → which data formats/database management system to use.
  → whether external data sources might be used.
- Application layer:-
  - Cares the programing and HTML languages, models, protocols, and application to be used.

* Presentation layer:-

Developers focused on the 'external' concern, e.g. front-end layout, HTML, templates.

Developer must be able to properly

→ Analyze requirement
→ Implement right app.
→ Test and validate result.
→ Evolve the app as needed.

* Why RE?

→ To deliver the exact product that the customer wants.
→ 340 companies Austria 1995, more than two third of these companies regarded the development of a requirement document as a major prob.
→ A survey 8000 projects - Standish Group
30% project failed
70% remaining projects did not meet expectations of customer.
→ 16% systems fully meet requirements.
→ 53% : n didnot satisfy the requirement capabilities.
(Cutter Consortium 2000)

Requirement Engineering:

*Requirements:

* A condition or capability needed by a user to solve a problem or acheive an objective.
* A condition or capabilities that must be meet of passesed by a sysstem or system component to satisfy a contract, standard, specification, or other formally imposed documents.
* A documented representation of a condition or capability.

*Source of requirement

- The individual objective and expectation of stakeholders
- Stakeholders are people or organizations that have direct or indirect influence on the requirement in system development.
- Important stakeholders are customers, users, and developers.
- Typical stakeholders for web applications include content authors, domain experts, usability experts, or marketing professionals.

Q. What is stakeholders?

Q. Project তিনটা থাকা হলে কয়টা কয়টা ?

*Requirement document:

A requirement document summerizes all requirements and constraints agreed between the contractor and the customer.

(i) Requirement Elicitation: Gathering requirements from stakeholders through interviews, surveys, and observations.

(ii) Document: Understanding, prioritizing, and resolving conflicts in gathered requirements.

(iii) Verification: Ensuring the requirements meet stakeholders need and are feasible.

(iv) Management: Handling changes, traceability and updates throughout the project life cycle.

* RE Specifics:

→ Multidisciplinarity.
→ Unavailability of stakeholders.
→ Volatility of requirements and constraints.
→ Unpredictable operational environment.
→ Impact of legacy system.
→ Significance of quality aspects.
→ Quality of the user interface.
→ Quality of content.
→ Developer inexperience.
→ Firm delivery dates.

* Principle of RE:

(i) Understanding the system context.
(ii) Involving the stakeholders.
(iii) Iterative definition of requirements.
(iv) Focusing on the system architecture.
(v) Risk orientation.

* Types of Requirement:

→ Functional.
→ Content
→ Quality
→ System environment
→ User interface
→ Evolution
→ Project constraints.

- **Functional:**
  → What the system should do.
  → e.g: user login, product search form, submission.
- **Non-Functional:**
  → How the system should behave.
  → e.g: performance, security, scalability, accessibility.
- **Content:**
  → Specific to web apps.
  → Includes multimedia, text and dynamic data
  → e.g: homepage should display latest news in real-time

- **User Requirement:**
  - → Expectations from different user types (role)
  - → e.g: admin can manage users, guest only browse product.
- **Interface Req:**
  - → Details about how users interact with the system
  - → e.g: responsive design, browser compability.
- **Quality Req:**
  - → Functionality
  - → Reliability
  - → Usability
  - → Efficiency
  - → Maintainability.
  - → Portability.

Modeling Web Application:

① System modeling:

def → The process of developing abstract models of a system.

→ Representing a system using graphical notations – UML, Use case.

② Aims of Modeling:

→ Providing a specification of a system to be built in a degree of detail sufficient for that systems implementation.

③ Results of Modeling:

→ Models representing the relevant aspects of the system in a simplified and ideally comprehensive manner.

④ System Modeling:

→ Each model represent a different view or perspective of the system.

① External Perspective:

System context and environment means identifying the system boundaries and objectives, and involving stakeholders in identifying the application services and function that the system will provide.

② Interaction Perspective:

How the systems interact with the environment.

Tool – UML, Use-case diagram.

③ Structured Perspective:

How the system is organized. Identifying the components and their relationship. e.g.: database - ERD, Tool: UML, USE-case diagram, deployment diagram and component diagram.

④ Behavioral Perspective:

Dynamic behavion of the system.

Solved 10.08.25

10.08.25

Used of Models:

- Requirement engineering:
  - i. Understand and analyze requirements.
  - ii. Document requirement clearly.
  - iii. Communicate between stakeholders.
  - iv. Validate and verify requirements

Example: Use-case diagram, DFDs, ER diagrams.

17.08.25

~~Use case diagrams:~~

~~*Goal: The goal of the~~

- Design phase:
  - Provide a blueprint for system implementation.
  - Specify architecture, components and interfaces.
  - Show data structures and algorithms.
  - Guide developers in coding.

Example: Class diagrams, Sequence diagrams, State diagrams, ER diagrams, Architectural model.

- Post-Implementation:
  - Support maintenance and updates.
  - Document as-built system for future reference.
  - Help in training and onboarding new developers.
  - Assist in system evaluation and improvement.

Example: Updated architecture diagram, workflow model, maintenance model.

Q. Why system modeling?

because it helps us!

i) Understand the system clearly — instead of vague words, we use diagrams/models to see how parts fit.

ii) Communicate with others — models are like a "common language" between developers, clients and managers.

iii) Detect problems early — easier to fix a design mistake in a diagram than in a running program.

iv) Guide implementation — the model acts as a "blueprint" for building the actual system.

Conventional Software:

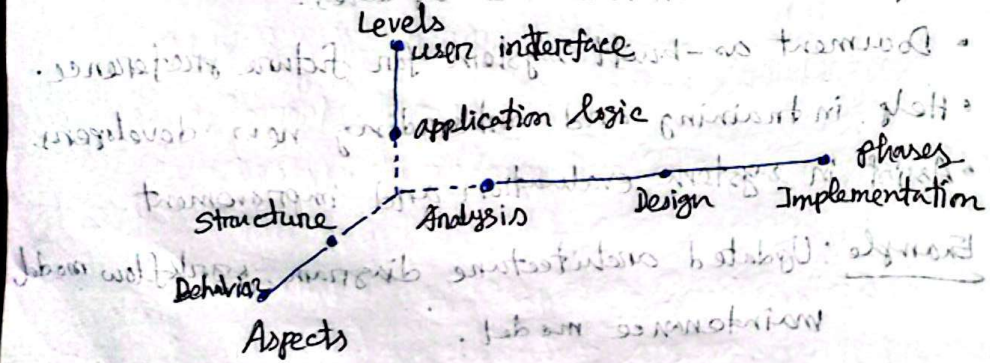


Fig: This figure shows the scope of modeling spans three orthogonal dimensions.

Diemension in system modeling:

* The first diemension traditionally comprises the application logic level and the user interface level in the sense of an encapsulation of that "what" and "how" of an application.
* Aspects known as structure (i.e., objects, their attributes, and their relationships to other objects) and behavior (i.e., functions and processes). both of the application logic and the user interface form another dimension.
* Development phases — Requirement → Design → Implementation.

UML in system modeling:

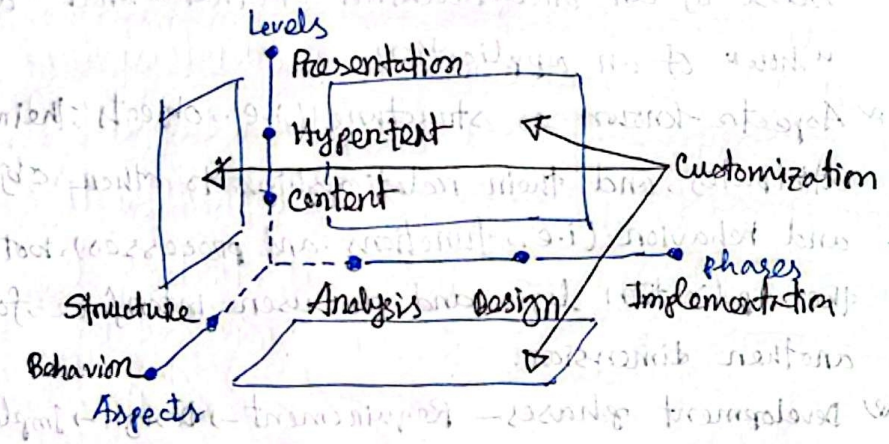
The Unified Modeling Language (UML) is a visual language for -

- Specifying and documenting the artifacts of a system.
- ~~Structural — class diagrams~~.
- Structural (what the system is)

  Example :- Class diagram → shows classes and their relationships.
- Behavioral (what the system does).

→ Use case diagram - shows how users interact with the system.

→ State machine diagram - shows how objects change state.



Web Application Model Levels:

Levels: The three levels are:

i) Content level:

- Deals with information (data, text, images, videos) and the application logic that processes it.
- Example: bus schedule, routes, booking logic in a bus scheduling system.

ii) Hypertext level:

- Organizes content into nodes (pages or sections) and links (navigation between them).
- Example: Home → About → Booking page.

iii) Presentation level:

- focuses on how information is shown to the user (UI design, layout, colors, fonts).
- Example: how the bus timetable is displayed on the website.

□ Clear Separation = Reuse + Reduced

□ Aspects:

- Structure and behavior are modeled at each of the three levels; i.e. at content, hypertext, and presentation.
- Web application that make mainly static information available require less behavior modeling compared with highly interactive web applications, such as for example, e-commerce applications which provide search engines, purchase order function etc.

□ Phases:

- Depending on the type of web application, it should be possible to pursue an information-driven approach, i.e. starting with content modeling, or a presentation driven approach i.e. starting with modeling of the applications presentation aspects

- Models also ensure the sustainability of solution ideas in contrast to shorter-lived software solutions.

☐ Customization:

- Consider the context, e.g. users preferences, device characteristics or bandwith restrictions, and allows for adaption of the web application accordingly.
- It influences all three web modeling diemensions of content, hypertext, and presentation structure and behavior, and should be taken into account on all phases of the development process.

☐ System modeling of web applications

1. Requirement modeling:
   - Shows what the system should do.
   - Diagram: Use-case, Activity.
2. Content Modeling:
   - Shows data and system behavior.
   - Diagram: Class, State machine.
3. Navigational Modeling:
   - Shows how users moves through the system.
   - Model nodes (pages) and links (navigation)
4. Presentation Modeling: • Shows how the UI looks.
   - Model page layout, interface element.

17.08.25.

* Use-Case diagram

Goal - The goal of the diagram is to provide a high-level explanation of the relationship between the system and the outside world.

☐ In this diagram, the main aspect is to view which services the system provides and how the different human users use the services.

Focus: It focuses on what the system should do (functional requirements) - not how it does it.

Purpose:
- Identify functional requirement.
- Define system scope.
- Show what interacts with the system and how.
- Provide a starting point for detailed design.

* Activity diagram:

Shows step-by-step workflows.

☐ Supports:
- Choices (branching)

- Iteration (loops)
- Concurrency (parallel actions)

This diagram is considered the detailed version of the use case.

Symbols →

System → Rectangle

Use-Case (task) → Oval

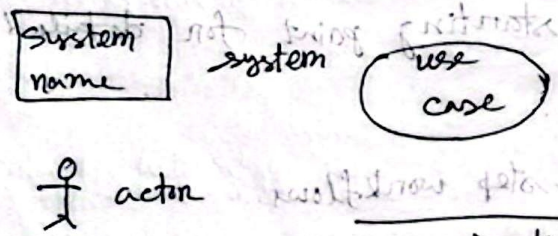
Acton → Stick figure (external user or system)

Communication line → Straight line connecting acton to use case.

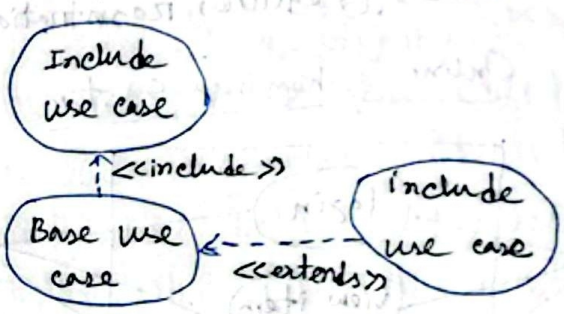
Include → One-use-case includes another.

Extend → One use-case can optionally extend another.

Notation:



Notation:



- The include relationship represents the inclusion of one use case within another.
- The extend relationship represents the extension (optional) of the use case to include optional functionality.

Online shopping system:

Actors: Customer, Credit-service, admin.

Actions:

Admin - Validates login

Customer - Views items, adds to cart, checks out, pays.

Online video sharing system:

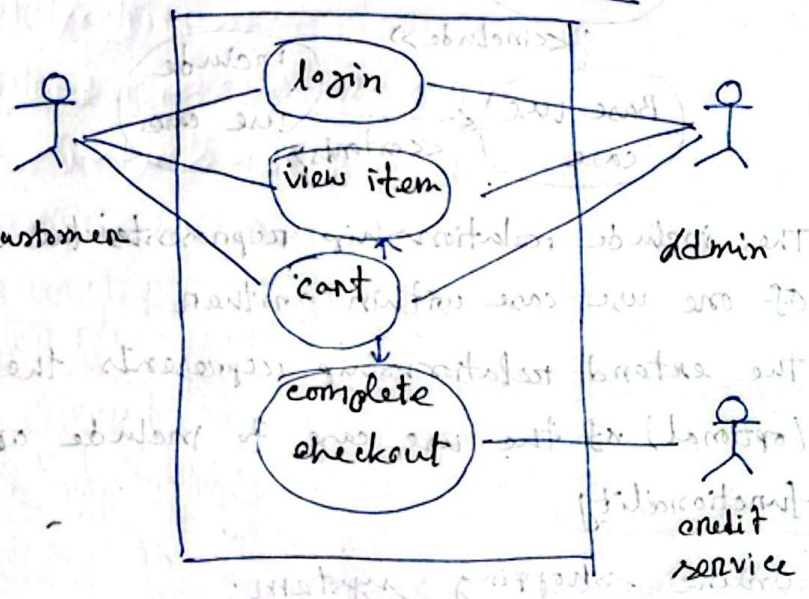
Actn - user

Actions - Search videos

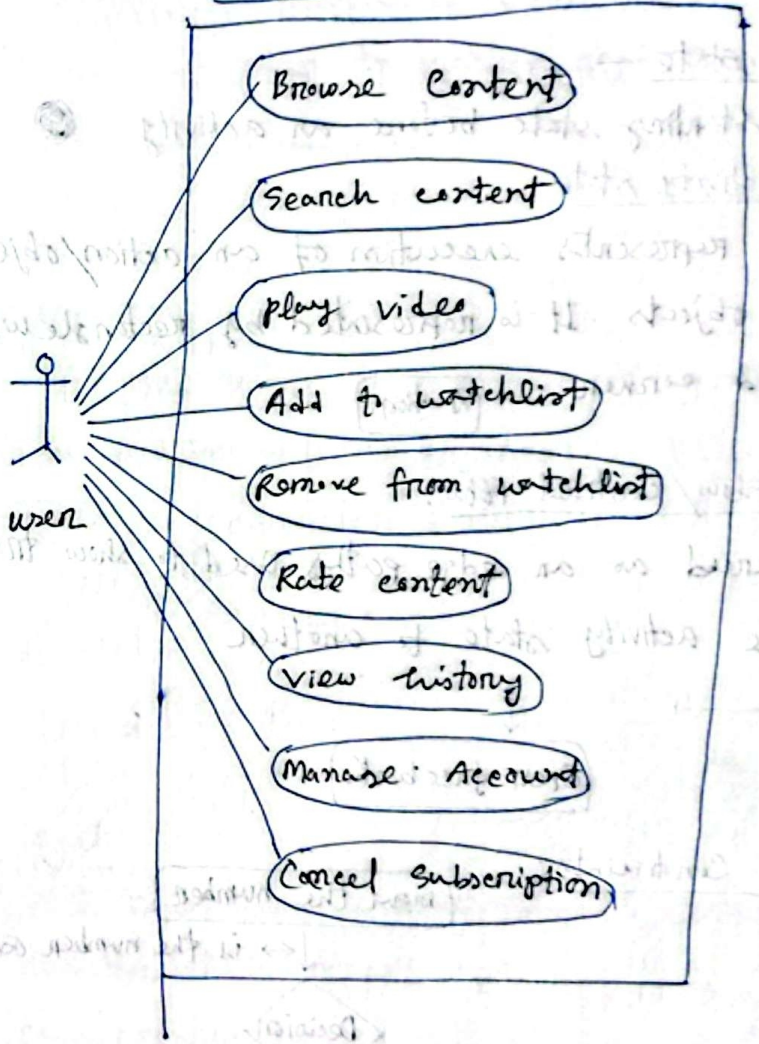
view videos.

Share videos. (requires registration).

Online Shopping System



Online Video Sharing System



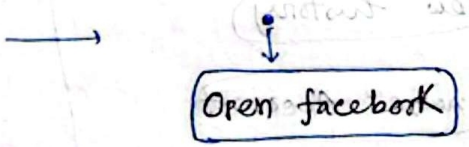
21.08.25

Activity diagrams:

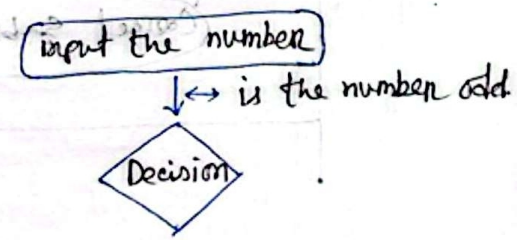
① **Initial State** →
→ The starting state before an activity

② **Action/activity state** –
Activity represents execution of an action/objects on by objects : It is represented by rectangle with rounded corner.

③ **Action Flow/control Flow**:
It referred as an edge paths. Used to show transition from one activity state to another

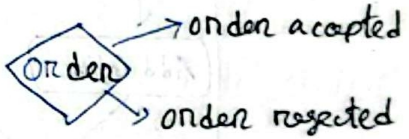


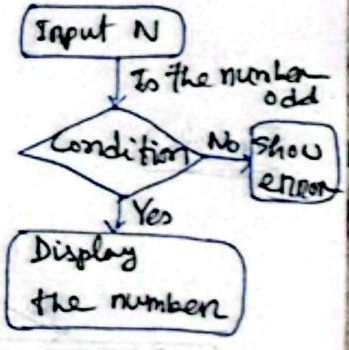
④ **Activity Constraints**:



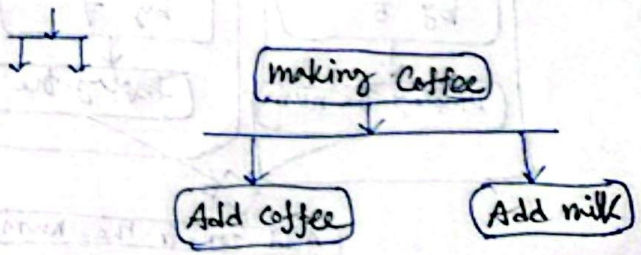
⑤ **Decision node and branching**:
When we need to make a decision before deciding the flow of control.

⑥ **Guards**
A guard refers to a statement written next to a decision node on an arrow.



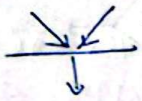


⑦ **Fork**
Fork nodes are used to support concurrent activities which means both activities will be executed concurrently.

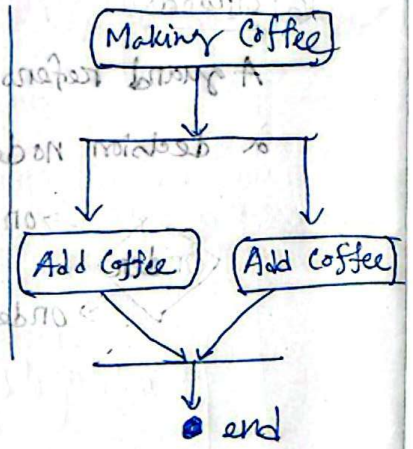


⑧ Join

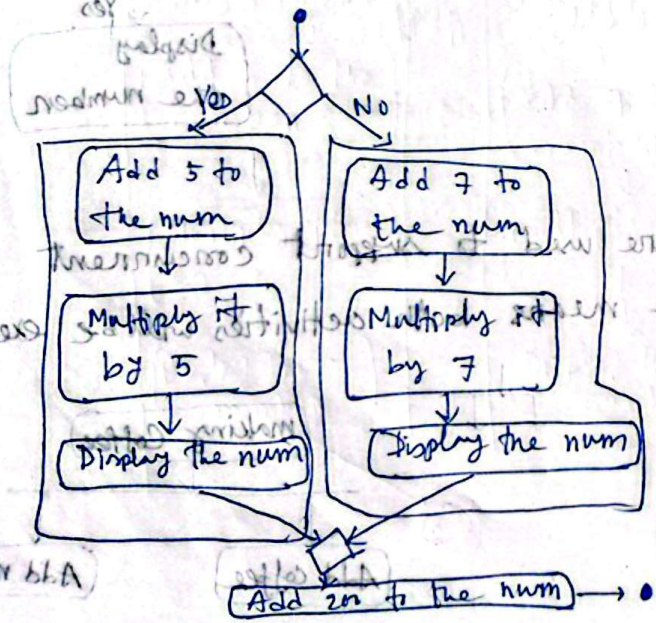
Join nodes are used to support concurrent activities converging in to one. For join notations, we have two or more incoming and one outgoing edge. If your activity states go into a fork, then you may need to join

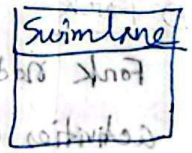


⑨ Merge Scenarios arise when activities that are not being executed concurrently must be merged. We A merge combines multiple activities into one when control moves forward regardless of the path



Examples:





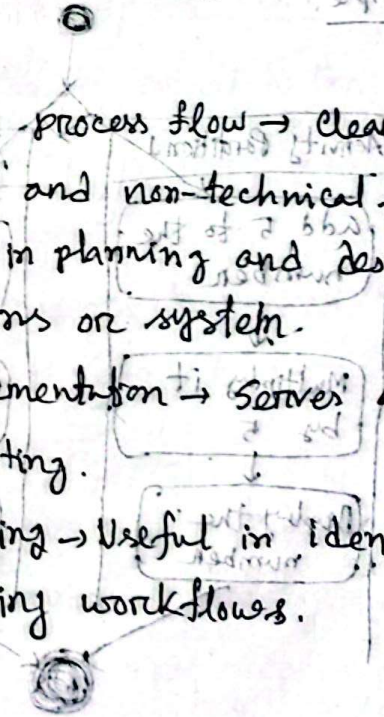
Activity diagram: is considered a detailed version of a Use Case Diagram since it describes the flow of actions within a system more precisely.

Supports:

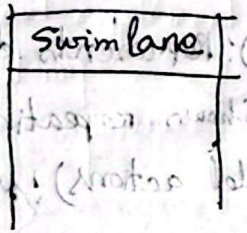
- Choices (branching): Represents decision-making.
- Iteration (loops): Shows repeating actions.
- Concurrency (parallel actions): Models activities happening at the same time.

Purpose:

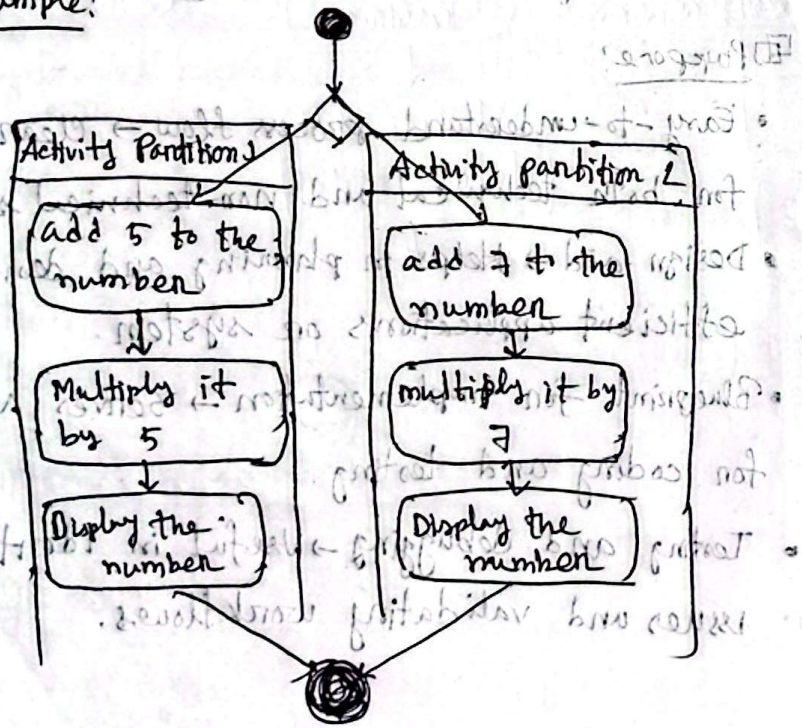
- Easy-to-understand process flow → Clear visualization for both technical and non-technical stakeholders.
- Design aid → Helps in planning and designing efficient applications or system.
- Blueprint for implementation → Serves as a guide for coding and testing.
- Testing and debugging → Useful in identifying issues and validating workflows.



- **Swimlanes** – we use swimlanes for grouping related activities in one column. Swimlanes group related activities into one column or one row. Swimlanes can be vertical and horizontal.



example:



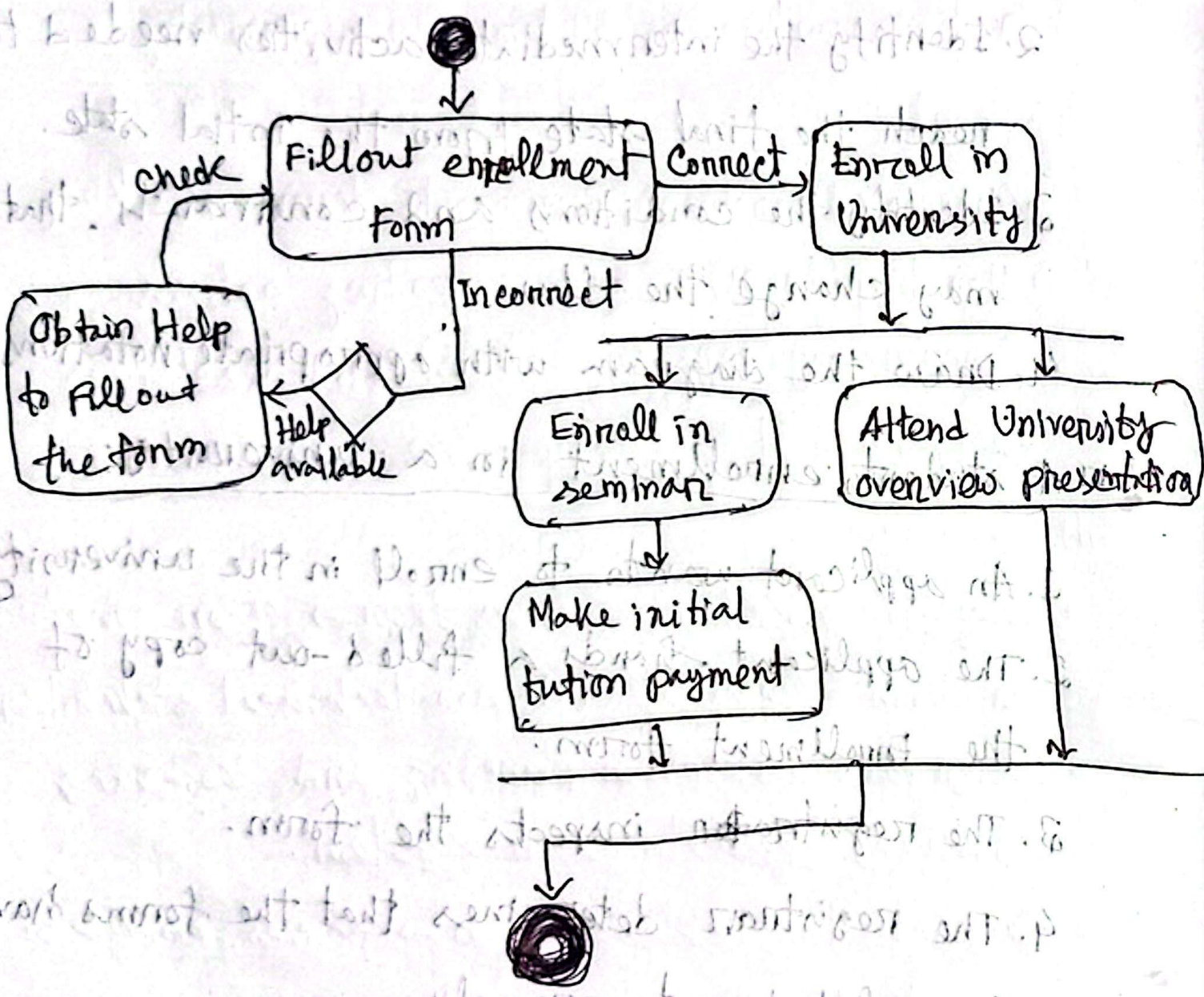
- **How to draw an activity diagram step-by-step:**

1. Identify the initial and final state.
2. Identify the intermediate activities needed to reach the final state from the initial state.
3. Identify the conditions and constraints that may change the flow.
4. Draw the diagram with appropriate notations.

- **A students enrollment in a university:**

1. An applicant wants to enroll in the university.
2. The applicant hands a filled-out copy of the Enrollment form.
3. The registration inspects the form.
4. The registrar determines that the forms have been filled out properly.
5. The registrar inform the student to attend in university overview presentation.
6. The registrar helps the student to enroll in seminars.

7. The registrar asks the student to pay for the initial tuition.



Web Architecture

02.09.25
04.09.25

Definition:

- Web architecture refers to the overall structure of a web site or web application, including the way it is designed, implemented and deployed.

It involves the use of technologies and protocols such as HTML, CSS, JS, and HTTP to build and deliver web pages and applications to users.

It defines how the components of a web application (client, server, database network, APIs etc.) interact with each other to deliver web services and contents to users.

Importance of Web Architecture

Fundation of development:

Web architecture is the bedrock upon which all other components of a web application are built. A solid architectural design ensures smoth development, saves times and cost in the long run, and makes it easier to extend the system. Without it, task such as simplyfying, scaling or adding new features across different technologies become significantly harder.

Difficult to modify:

High level web architecture decisions are hard to change once development has started. Unlike small code changes, altering the core architecture often requires starting over from scratch. This make it crucial to make the right decisions at the beginning of a project.

Impact of maintenance cost:

Poor architectural choices lead to increased maintenance costs. A poorly structured system becomes complex, harder to debug, less secure, and more expensive to maintain and update. On the other hand, a well-designed architecture reduces long-term risks and ensures sustainability.

Components of Web Architecture:

The Client: This is the web browser or application that the user interacts with. The client send requests to the server and receives response in return.

The server: This is the computer or group of computers that host the website or web application. The server process requests from the client and sends back the appropriate response.

The database: This a collection of data that is used to store and retrive information for the website or web application. The database can be located on the same server as the website or web application or it can be hosted on a separate server.

7. The design and the layout: this refers to the way the website or webapplication is structured and organized, including the layout, navigation and overall appearance.

The frameworks and libraries: these are tools and resources that are used to built and maintain the website or web application. They can be includes frameworks like Ruby on Rails or Django or libraries like jQuery or React.

The development and hosting: this refers to the way the web site or web application is deployed and hosted. Including the hosting environment (such as a shared hosting plan or a cloud platform) and the process for deploying updates and changes to the web site or web application.

Types of web Architecture:

1. Monolithic Architecture:
   - The entire app is built as a single unit (front-end, back-end, database tightly coupled).
   - Easy to build but hard to scale.
   - Example: Traditional PHP applications.
2. Two-tier architecture:
   - Client ↔ server
   - Client sends requests, server responds (database may be integrated in the server)
3. Three tier Architecture:
   - Client ↔ Application Server ↔ database.
   - Most common in web engineering.
   - Improves scalability, security, and maintainability.
4. Microservices Architecture:
   - Application is split into independent small communicating via APIs.
   - Example: Netflix, Amazon.
   - Benefits: Scalability, flexibility, fault isolation

5) Serverless Architecture:

- Developers only writes functions; cloud providers manage servers.
- Examples: AWS Lambda, Google Cloud Functions.

Modern Web Architecture

1. MVC (Model - View - Controller):

- The MVC framework is an architectural / design pattern that separates an application into three main logical components: Model, View and controller.
- Each architectural components is built to handle specific development aspects of an application. It isolates the business logic and presentation layer from each other.
- It was traditionally used for desktop graphical user interfaces (GUIs). Nowadays, MVC is one of the most frequently used industry-standard web development frameworks to create scalable and extensible projects. It is also used for designing mobile apps.

Features of MVC:

- It provides a clear separation of business logic, UI logic and input logic.
- It offers full control over your HTML and URLs which makes it easy to design web application architecture.
- It is a powerfull URL-mapping component, using which we can built applications that have comprehensible and searchable URLs.
- It supports Test Driven Development (TDD):
  - Test-driven development is a way of writing code that involves writing an automated unit-level test case that fails, then writing just enough code to make the test pass, then refactoring both the test code and the

production code, then repeating with another new test case.

MVC

Model:

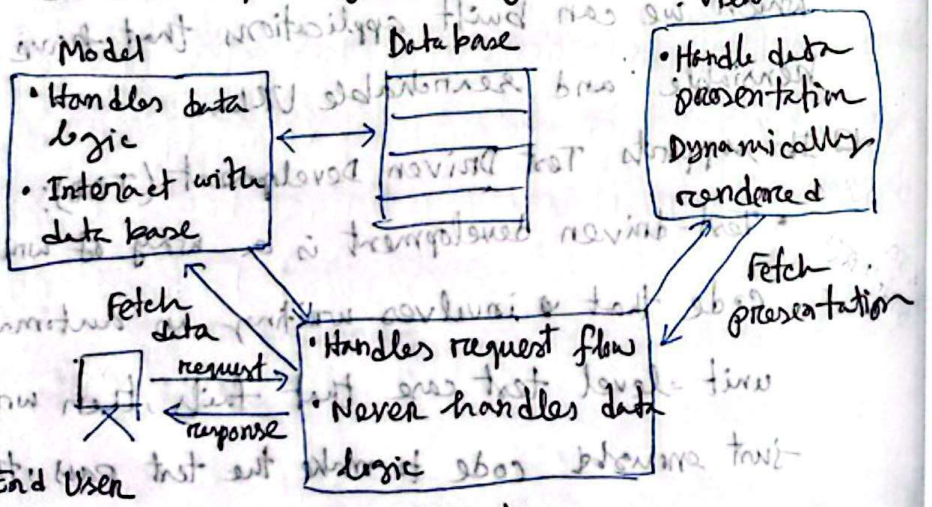
(i) Database table.
(ii) Session information
(iii) Logics.

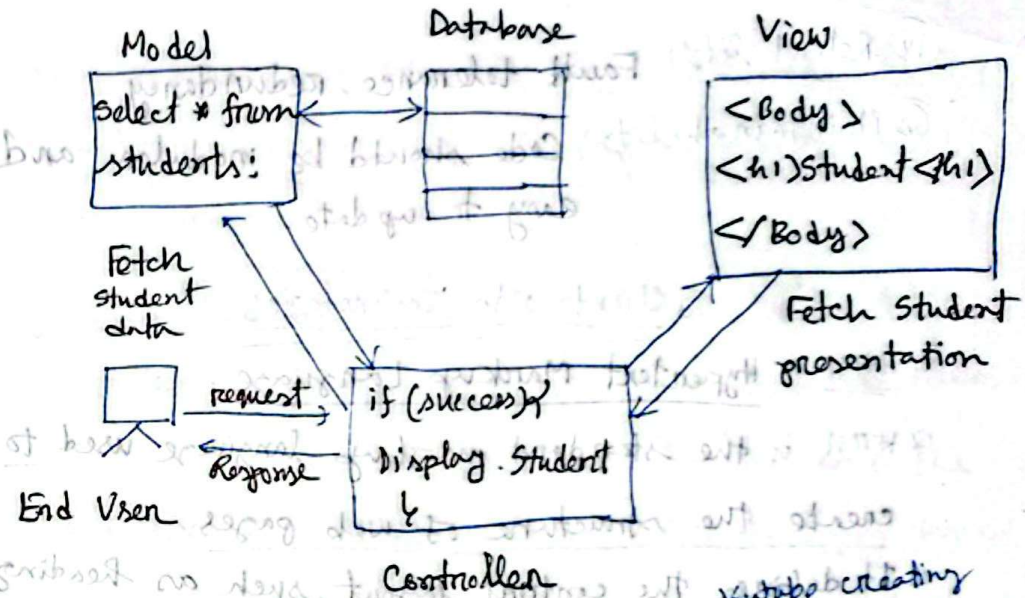
View:

(i) HTML, CSS
(ii) Server sides templates.

Controller:

(i) Client-side scripting.
(ii) HTTP Request processing.





- More Modern Architecture:
  - MVVM (Model-View-View Model): Popular in Angular, React. (Youtube creating framework)
  - Event-Driven Architecture: Users real-time event streaming (e.g. Kafka)
  - API-Centric Architecture: REST/GraphQL-based web apps.
- Key considerations in web application Architecture:
  (i) Scalability: Can handle growing traffic.
  (ii) Performance: Fast Response time.
  (iii) Security: Encryption (HTTPS, TLS/SSL), authentication, authorization.

(iv) Reliability: Fault tolerance, redundancy.

(v) Maintainability: Code should be modular and easy to update

Client-side Technologies

HTML - Hypertext Markup Language:

- HTML is the standard markup language used to create the structure of web pages.
- It defines the content layout, such as headings, paragraphs, images, links, tables, forms, and multimedia.
- Hypertext is whatever text we see in a computer display, can be linked to other text (Hyperlinks)
- The concept markup originates from the publishing industry and generally means typographic instruction for document formating.

CSS - Cascading Style Sheet.

- Describe the presentation of a document written in a mark up language like HTML. It allows developers to control the look and feel of a webpages, making it visually appealing and user-friendly.
- Cascading: This refers to the way styles are applied to HTML elements. Multiple style rules can be applied to a single element and the browser determines which rule to apply based on a hierarchy of importance.