**🧪 Software Testing – SDLC (Software Development Life Cycle)**

**1. Requirement Analysis & Information Gathering**

* First step of SDLC where project details are collected from the client.
* **Client Interaction:** Understand the project goals, budget, and delivery timelines.
* **Key Considerations:**
  + **Type of Project** (e.g., Website, Application)
  + **Resources Required:** Developers, Testers, Designers
  + **Deadline & Costing:** Define the timeline and cost estimation
  + **Team Allocation:** Allocate the project team including interns (e.g., 1 intern per 50 PHP tools/technologies)

**Example Project Plan:**

* **Team Roles:**
  + Business Analyst (BA)
  + Project Manager
  + Team Leader
* **Budget:** ₹10 Lakh
* **Duration:** 3 Months
* **Resources:** 5 Members
* **Resource Cost:** ₹1 Lakh per person
* **Project Cost:** ₹5 Lakh to ₹10 Lakh
* **Website Count:** 15–20
* **Additional Features:** 5 extra pages

**2. Design**

* Involves preparing the **architecture** and **flow of the project**
* **System Architecture:** Define front-end and back-end interaction
* **Flow Diagrams:** Use Case Diagrams, Data Flow Diagrams (DFD), etc.
* **Actor Identification:** Identify users, admins, and other actors interacting with the system

**3. Development**

* Actual coding phase begins here
* **Frontend Development:** UI/UX design using HTML, CSS, JavaScript, etc.
* **Backend Development:** Server-side logic using PHP, Python, etc.
* **Web Design:** Layout, templates, responsiveness

**4. Testing**

* Ensure the software works as expected and is bug-free
* **Manual Testing:** Performed by testers without tools (e.g., functional, regression testing)
* **Automation Testing:** Using tools (e.g., Selenium, QTP) for repetitive test cases

**5. Deployment**

* The software is launched for end-users
* **Live Hosting:** Project is made available on the internet
* **Launch:** Includes versioning, hosting setup, domain linking

**6. Maintenance**

* Ongoing support after deployment
* **Free Maintenance Period:** Usually 1 year
* **Bug Fixes and Updates:** Continuous improvements based on user feedback
* **Cost Range:** ₹20,000 to ₹30,000 per year after the free period

**Software Testing Assignment**

**✅ Software Tester – Job Profile**

* Understand **client requirements** and convert them into a **test plan** or test file.
* Perform tests on different modules like:
  + **Login via Gmail or Twitter (with OTP)**
  + **Add to Cart** functionality
* Identify **bugs or issues**
* Compare **Expected Result** vs **Actual Result**
* Ensure the app is:
  + **Bug-free**
  + **User-friendly**
  + **Easy to navigate**

**🧱 Testing Levels / Pillars**

**🔹 Level 1 – Unit Testing**

* Testing of individual components or **single functions**
* Done by developers
* Example: Login functionality only

**🔹 Level 2 – Integration Testing**

* Testing more than one function **together**
* Ensures modules work correctly when combined
* Example: Login + Add to Cart + Checkout flow

**🔹 Level 3 – System Testing**

* Testing the **entire website or application**
* Done by **automation testers**
* Ensures complete system works as per requirements

**🔹 Level 4 – UAT (User Acceptance Testing)**

* Done by **end-users or clients**
* Validates whether software meets business needs
* Feedback is collected before final deployment

**Practical Testing Types**

**🔸 1. Database Testing**

* Verify data is properly saved, updated, and fetched
* Use **SQL** for queries

**🔸 2. API Testing**

* Check if backend and frontend communicate properly
* Use **Postman** or tools like **JMeter, REST Assured**

**🔧 Testing Tools**

* **Postman:** For API Testing
* **SQL:** For Database Testing
* **Selenium:** For UI Automation Testing (optional)
* **JIRA/Bugzilla:** For bug tracking (optional)

**🌐 What is an API?**

**API = Application Programming Interface**

* Acts as a bridge between **Frontend** (user interface) and **Backend** (server)
* Used by testers to validate data transfer and business logic

**⚙️ Advantages of API Testing**

1. **Language Independent**
2. Faster and more reliable than UI testing
3. Can test core logic and data easily

**📦 Types of APIs**

| **Type** | **Description** | **Format** |
| --- | --- | --- |
| **REST API** | Lightweight, no strict rules, less secure | JSON, XML, HTML |
| **SOAP API** | Protocol-based, strict structure, more secure | XML only |

**Common API Data Formats**

| **Format** | **Full Form** |
| --- | --- |
| JSON | JavaScript Object Notation |
| XML | eXtensible Markup Language |
| HTML | HyperText Markup Language |
| XHTML | Extensible HTML |
| TEXT | Plain text |

**🛠️ HTTP Methods in API**

| **Method** | **Action** |
| --- | --- |
| GET | Fetch data (Read) |
| POST | Create data |
| PUT | Full update |
| PATCH | Partial update |
| DELETE | Remove data |

**Database Testing**

**🔍 What is Database Testing?**

Database Testing ensures:

* Data is **stored**, **retrieved**, **updated**, and **deleted** correctly.
* The **backend database** works accurately with the **frontend UI**.
* Data is stored in a **structured and organized format**.

**🧱 Where is Data Stored?**

Data is stored in **databases** using systems like:

**🗂️ 1. RDBMS – Relational Database Management System**

* Stores data in **tables (rows & columns)**
* Data is **related** using keys
* Uses **SQL (Structured Query Language)**

**Examples of RDBMS:**

* MySQL (used with XAMPP)
* Oracle
* SQLite
* PostgreSQL

**🔗 2. NoSQL – Non-relational Databases**

* Stores data in **JSON-like documents**
* Flexible schema (not table-based)
* Used for large, fast-changing data

**Example:**

* MongoDB

**Example: Registration Form Data**

| **Field** | **Data Type** |
| --- | --- |
| Name | Text |
| Email | Text |
| Password | Text |
| Mobile No. | Number |

**Database Structure in MySQL**

**1. Database → Tables → Records (Data)**

In XAMPP, **MySQL** is used to manage database operations.

**Basic SQL Commands with Examples**

| **#** | **SQL Command** | **Purpose** | **Example** |
| --- | --- | --- | --- |
| 1 | CREATE DATABASE | Creates new database | CREATE DATABASE instagram; |
| 2 | DROP DATABASE | Deletes entire database | DROP DATABASE instagram; |
| 3 | CREATE TABLE | Defines a table | CREATE TABLE users (id INT, name VARCHAR(100)); |
| 4 | DROP TABLE | Deletes a table | DROP TABLE users; |
| 5 | SELECT | Retrieves data | SELECT \* FROM users; |
| 6 | INSERT | Adds new data | INSERT INTO users VALUES (1, 'Shahid'); |
| 7 | UPDATE | Modifies data | UPDATE users SET name='John' WHERE id=1; |
| 8 | DELETE | Removes data | DELETE FROM users WHERE id=1; |
| 9 | VIEW | Virtual table based on SELECT | CREATE VIEW user\_view AS SELECT name FROM users; |
| 10 | JOIN | Combines rows from 2+ tables | SELECT users.name, orders.product FROM users JOIN orders ON users.id = orders.user\_id; |

**Why is Database Testing Important?**

* Ensures **data accuracy** and **consistency**
* Avoids data loss or corruption
* Confirms data is stored and fetched **as expected**
* Detects mismatches between UI and DB (e.g., order total showing wrong in invoice)

**How Testers Do Database Testing**

1. Use **phpMyAdmin** in XAMPP or a **MySQL Client**
2. Run SQL queries to check:
   * If data inserted from forms is stored correctly
   * If updates and deletes reflect properly
3. Compare data in UI vs Database

DATABASE – SQL Commands (MySQL/XAMPP)

📘 1. Create Table

Creates a new table named student with 3 columns:

❌ 2. Drop Table

Deletes the entire student table permanently:

➕ 3. Insert Data

Inserts a new student into the table:

🗑️ 4. Delete Record

Deletes the student whose sid is 3:

🔍 5. Select Data (All Records)

Displays all data from the student table:

🔎 6. Select Specific Record

Displays only the record where sid = 2:

✏️ 7. Update Record

Updates the student record with sid = 4 to new values: