2 days Study Plan for Salesforce Developer

📅 Day 1 – Apex, Triggers & Data Modeling

🧠 1. Salesforce Development Basics

- Apex is Salesforce's object-oriented programming language (like Java).
- Used when point-and-click tools (like Flow) are not enough.
- Helps customize business logic, process automation, and integrations.

🧱 2. Data Modeling in Salesforce

- Work with Standard & Custom Objects.
- Use **Lookup** and **Master-Detail** relationships.
- Understand formula fields, roll-up summary fields, and schema builder.

🔁 3. Apex Triggers

- Automates logic before/after database operations (insert, update, delete).
- Syntax: trigger TriggerName on ObjectName (events) {}
- Use Trigger Handler patterns for clean, reusable code.
- Access Trigger.new, Trigger.old, isInsert, isUpdate, etc.

🔍 4. SOQL & SOSL

- **SOQL**: Query specific records like SQL.
  - Example: SELECT Id FROM Contact WHERE LastName = 'Sharma'
- **SOSL**: Search across multiple objects using keywords.

🚦 5. Governor Limits

- Limits ensure fair usage in multi-tenant architecture.
  - Ex: Max 100 SOQL queries or 150 DMLs per transaction.
- Learn how to bulkify code and avoid exceeding limits.

🛡️ 6. Exception Handling

- Use try-catch blocks to handle errors.
- Catch DML exceptions or custom exceptions gracefully.
- Ensure data integrity and smooth user experience.

📅 Day 2 – LWC, Async Apex, Testing & Integration

⚡ 7. Lightning Web Components (LWC)

- Modern frontend framework using HTML, JS.
- Use @track, @api, @wire decorators.
- Communicate with Apex using @wire or imperative methods.

⏳ 8. Asynchronous Apex

- Useful for long or background tasks:
  - @future, **Queueable**, **Batch Apex**, **Schedulable**
- Choose based on complexity and data volume.

🧪 9. Apex Test Classes

- Every Apex class must be tested (min 75% coverage).
- Use @isTest, Test.startTest(), Test.stopTest()
- Validate logic using System.assertEquals().

🚀 10. Deployment Tools

- Use **Change Sets**, **VS Code + SFDX**, or **ANT Migration Tool**.
- Understand Sandbox > UAT > Production path.
- Track versions and changes with Git.

🌐 11. Integration Techniques

- Work with **REST** & **SOAP APIs**.
- Use HttpRequest, HttpResponse for REST callouts.
- Manage external calls with Named Credentials.

🔔 12. Platform Events

- Real-time event-driven communication within or outside Salesforce.
- Publishes/subscribes to events like OrderPlaced, StatusChanged.

📅 Day 1 – Apex, Triggers & Data Modelling – Detailed Explanation

---

🧠 1. Salesforce Development Basics

What it is:
Apex is an object-oriented, strongly typed language used to build backend logic in Salesforce. It runs on the Lightning Platform and supports classes, interfaces, exceptions, and database integration.

Where it's used:
When you need complex logic that can't be achieved with point-and-click tools like Flows or Process Builder.

🔍 What the interviewer expects:

- When to use Apex over declarative tools
- Real-life examples where you used Apex
- Syntax familiarity and governor limit awareness

🧱 2. Data Modeling in Salesforce

What it is:
The structure of how data is stored and related. Includes:

- **Standard/Custom Objects**
- **Fields** (text, picklist, formula)
- **Relationships** (Lookup, Master-Detail)
- **Schema Builder** for visualization

🔍 **What the interviewer expects:**

- Explain Lookup vs Master-Detail (and use cases)
- Design a mini schema on the spot (e.g., Product-Categories-Orders)
- Show understanding of roll-up summary fields (only for Master-Detail)

🔁 3. Apex Triggers

What it is:
Triggers automate logic before/after DML operations on records like insert, update, delete.

Syntax:

```
trigger AccountTrigger on Account (before insert, after update) {
   // logic
}
```

✅ Use Trigger Handlers for cleaner logic separation.

🔍 **What the interviewer expects:**

- Write or debug a trigger
- Explain Trigger.new, Trigger.old, isInsert, etc.
- Avoid DML inside loops (bulk-safe code)

🔍 4. SOQL & SOSL

SOQL (Salesforce Object Query Language):

- Structured query for single object or related objects
- Example: SELECT Name FROM Account WHERE Industry = 'Tech'

SOSL (Salesforce Object Search Language):

- Full-text search across multiple objects
- Example: FIND 'John' IN ALL FIELDS RETURNING Contact(Name), Account(Name)

🔍 **What the interviewer expects:**

- Differences and when to use each
- Nesting and relationship queries
- Avoid SELECT * pattern — query only what's needed

🚦 5. Governor Limits

What it is:
Salesforce puts execution limits on each transaction (multi-tenant platform).
Examples:

- 100 SOQL queries
- 150 DML statements
- 10MB heap size

🔍 **What the interviewer expects:**

- Why limits exist
- How to optimize Apex to stay within limits
- Proper use of collections, maps, sets to bulkify logic

🛡️ 6. Exception Handling

What it is:
Handling unexpected errors to prevent system crashes.
Use try-catch-finally blocks:

```
try {
  insert accountList;
} catch (DmlException e) {
  System.debug('Error: ' + e.getMessage());
}
```

🔍 **What the interviewer expects:**

- Show structured exception handling
- Differentiate between system vs custom exceptions
- Use meaningful error messages/logs

📅 Day 2 – LWC, Async Apex, Testing & Integration

⚡ 7. Lightning Web Components (LWC)

What it is:
A modern, fast frontend framework for Salesforce UI.
Uses JavaScript + HTML and integrates with Apex using decorators.

Key decorators:

- @api – public properties
- @track – reactive variables
- @wire – connects Apex/data to template

🔍 **What the interviewer expects:**

- Build a basic LWC
- Explain communication (parent-child, Apex)
- Compare LWC vs Aura

⏳ 8. Asynchronous Apex

Why it's needed:
For long-running tasks that exceed synchronous limits.

Types:

- @future – fire-and-forget
- **Queueable Apex** – chain jobs, more control
- **Batch Apex** – large data processing (10k+ records)
- **Schedulable** – run jobs on schedule

🔍 **What the interviewer expects:**

- When to use async vs sync
- Write a basic Queueable or Batch class
- Handle callouts in async code

🧪 9. Apex Test Classes

Why important:

- Required for deployment (min 75% coverage)
- Ensure your code works properly
- Written using @isTest annotation

Example:

```
@isTest
public class TestMyClass {
  static testMethod void testLogic() {
    // test logic
  }
}
```

🔍 **What the interviewer expects:**

- Write a simple test class
- Use of Test.startTest() and System.assertEquals()
- Good naming and separation of test data

🚀 10. Deployment Tools

Tools:

- **Change Sets** – simple UI-based deploy (within orgs)
- **VS Code + SFDX CLI** – modern, code-based
- **ANT Migration Tool** – XML & scripts
- **Git** – version control

🔍 **What the interviewer expects:**

- Deployment strategy awareness
- Know how to resolve deployment failures
- Familiar with version control practices

🌐 11. Integration Techniques

Why it's needed:
Salesforce needs to communicate with other systems.

Common ways:

- **REST API** – JSON-based, lightweight
- **SOAP API** – XML, legacy systems
- **Callouts** – External requests from Apex
- **Named Credentials** – secured endpoints

🔍 **What the interviewer expects:**

- Build a REST callout (GET/POST)
- Explain endpoint structure, auth
- Error handling in integration

🔔 12. Platform Events

What it is:
A real-time event-publishing system (Pub/Sub model).
Good for communicating between apps, microservices, or internal modules.

Example Use:
Publishing an event when an order is shipped, and multiple systems react to it.

🔍 **What the interviewer expects:**

- Explain Pub/Sub model
- Show use case of Platform Event
- When to use this over triggers or flows