Practical no:- 07

AIM : Modeling Sequence diagram

**What is a Sequence Diagram?**

A **sequence diagram** models the **interaction between classes or components** in a system **step by step**, in **chronological order**. It's especially useful for:

* Visualizing **use case scenarios**
* Understanding how **objects communicate**
* Designing **system logic**

**Basic Elements of a Sequence Diagram:**

1. **Actors**: External users or systems that interact with your system.
2. **Objects (Lifelines)**: Represented by rectangles with dotted lines (lifelines) extending down.
3. **Messages**: Arrows that show communication (calls, returns, data).
4. **Activation Bars**: Thin rectangles on lifelines showing when an object is active.
5. **Loops, Conditions, Alternatives**: Control structures for logic flow.

**Example Scenario: "User Logs into a System"**

**Use Case:**

A user logs in → System checks credentials → System grants access.

**Sequence Diagram:**

Actor: User

Objects: LoginPage, AuthService, Database

User → LoginPage: enterCredentials()

LoginPage → AuthService: validateUser()

AuthService → Database: queryUser(username, password)

Database → AuthService: return userData

AuthService → LoginPage: loginSuccess()

@startuml

actor User

participant LoginPage

participant AuthService

database Database

User -> LoginPage: enterCredentials()

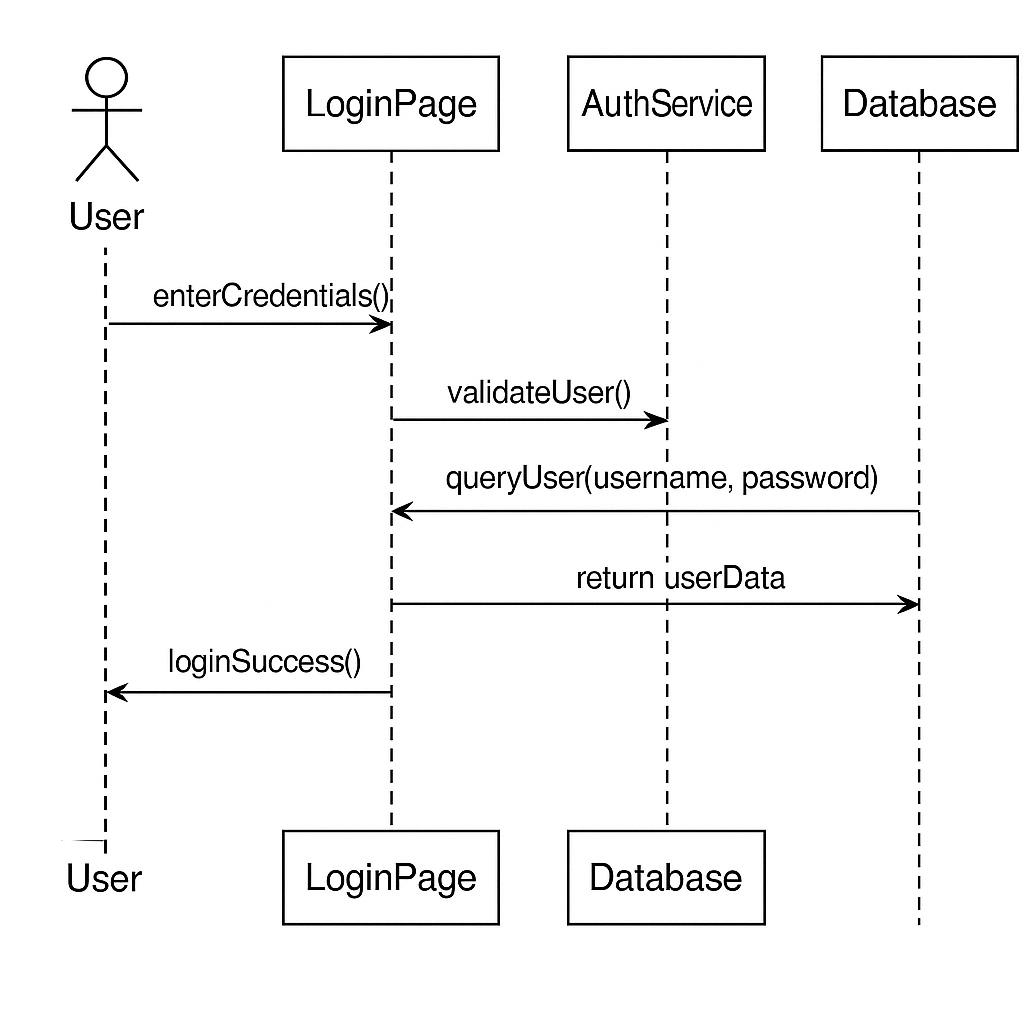
LoginPage -> AuthService: validateUser()

AuthService -> Database: queryUser(username, password)

Database --> AuthService: return userData

AuthService -> LoginPage: loginSuccess()

LoginPage -> User: showDashboard()@enduml



**Tips for Drawing:**

* Place **actors on the left**, then objects/components.
* Use **arrows** to show messages (solid for synchronous, dashed for returns).
* Use **alt/opt blocks** for conditions like login failure.
* Time flows **top to bottom**.