Experiment 6: Sketch Sequence diagram for the project

Learning Objective: Students will able to draw Sequence diagram for the project

Tools: Dia, StarUML

Theory:

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

Sequence Diagram representation

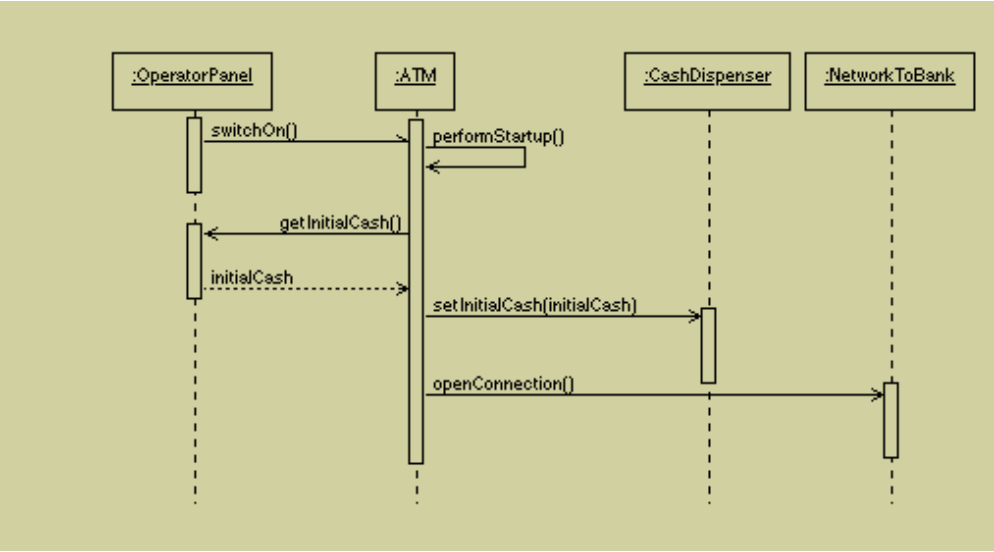
Call Message: A message defines a particular communication between Lifelines of an Interaction.

Destroy Message: Destroy message is a kind of message that represents the request of destroying the lifecycle of target lifeline.

Lifeline: A lifeline represents an individual participant in the Interaction.

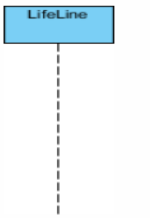
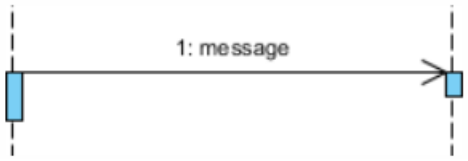
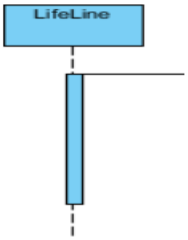
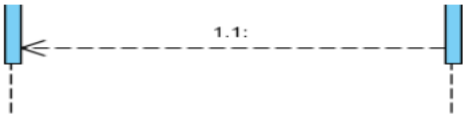
Recursive Message: Recursive message is a kind of message that represents the invocation of message of the same lifeline. Its target points to an activation on top of the activation where the message was invoked from.

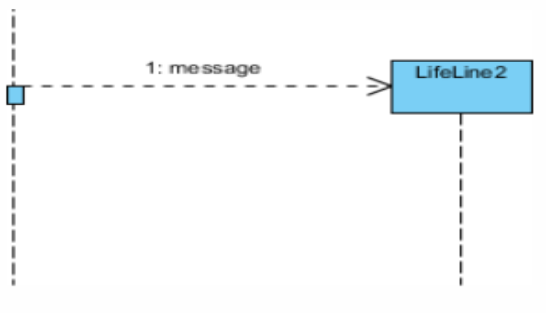
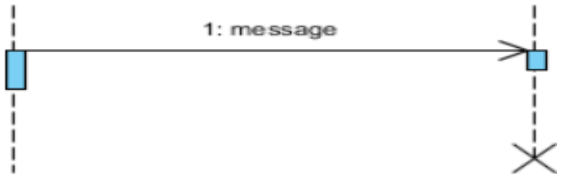
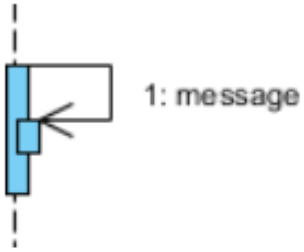
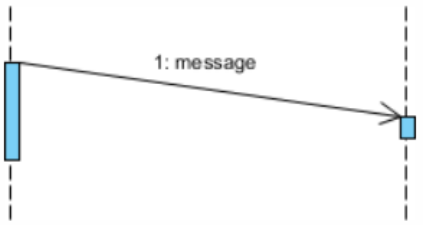
Sequence Diagram: Example for ATM System startup



It is clear that sequence charts have a number of very powerful advantages. They clearly depict the sequence of events, show when objects are created and destroyed, are excellent at depicting concurrent operations, and are invaluable for hunting down race conditions. However, with all their advantages, they are not perfect tools. They take up a lot of space, and do not present the interrelationships between the collaborating objects very well.

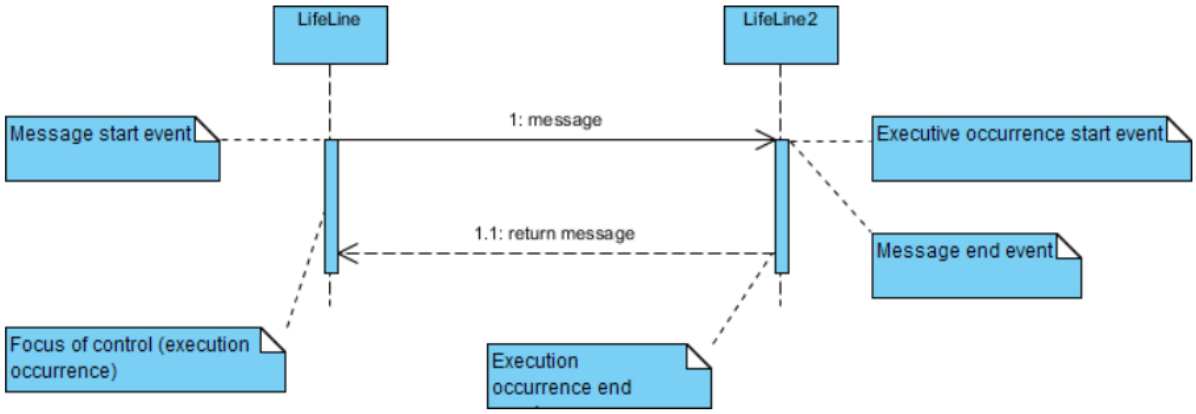
A sequence diagram is a type of interaction diagram from the Unified Modeling Language (UML) that shows how objects interact in a particular scenario of a system. It illustrates the sequence of messages exchanged between the objects involved in that scenario over time. Here are some more key theoretical concepts related to sequence diagrams:

Lifeline: A lifeline represents the existence of an object over a period of time. It extends vertically downward from the object's name or instance, indicating its duration during the sequence of interactions.	
Messages: Messages are depicted as arrows between lifelines, indicating communication between objects. Messages can be synchronous (blocking), asynchronous (non-blocking), or self-referencing (sent by an object to itself).	
Activation Bar: An activation bar, also known as an execution occurrence, represents the time during which an object is performing an operation. It is shown as a solid vertical bar extending from the lifeline, indicating the duration of the operation's execution.	
Return Messages: Return messages indicate the flow of control back to the sender after the completion of an operation. They are represented by dashed arrows.	

Create Messages: A create message is used to depict the creation of a new object. It is shown as an arrow with a solid circle at the receiving end, indicating the object being created.	
Destroy Messages: A destroy message is used to represent the destruction of an object. It is shown as an arrow with an "X" at the receiving end, indicating the termination of the object's existence.	
Recursive Message: Recursive message is a kind of message that represents the invocation of message of the same lifeline. It's target points to an activation on top of the activation where the message was invoked from	
Duration Message: Duration message shows the distance between two time instants for a message invocation.	
.	

Message and Focus of Control

- An Event is any point in an interaction where something occurs.
- Focus of control: also called execution occurrence, an execution occurrence
- It shows as tall, thin rectangle on a lifeline)
- It represents the period during which an element is performing an operation. The top and the bottom of the rectangle are aligned with the initiation and the completion time respectively.



Advantages of Sequence Diagrams:

1. **Clarity:** Sequence diagrams offer a clear visualization of interactions between objects, making it easier for stakeholders to understand system behavior.
2. **Communication:** They serve as effective communication tools among stakeholders, including developers, designers, and clients, aiding in discussing and refining system requirements.
3. **Dynamic Behavior:** Sequence diagrams depict dynamic aspects of a system, showing the order of message exchanges and the timing of interactions, which is essential for understanding system behavior.
4. **Testing:** They help in generating test cases by identifying potential scenarios and sequences of interactions, thereby aiding in the testing process.
5. **Design Validation:** Sequence diagrams can be used to validate the design of a system by identifying inconsistencies, ambiguities, or missing functionalities early in the development process.

Disadvantages of Sequence Diagrams:

1. **Complexity:** In complex systems, sequence diagrams can become intricate and difficult to interpret, especially when dealing with numerous objects and interactions.

2. **Abstraction Limitation:** Sequence diagrams might oversimplify or abstract certain aspects of a system, potentially leading to misunderstandings or overlooking critical details.

3. **Time-Consuming:** Creating and maintaining sequence diagrams can be time-consuming, especially when the system undergoes frequent changes or updates.

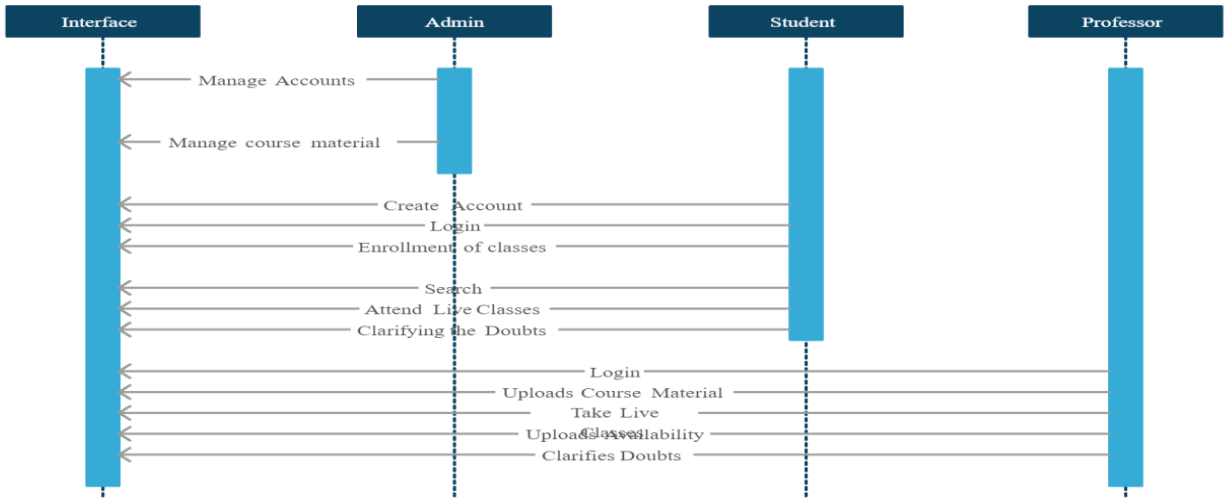
4. **Limited Scalability:** Sequence diagrams might not scale well for large systems with numerous objects and interactions, leading to cluttered and unreadable diagrams.

5. **Dependence on Implementation Details:** Sequence diagrams might inadvertently focus on implementation details rather than higher-level system behaviors, leading to a loss of abstraction and flexibility.

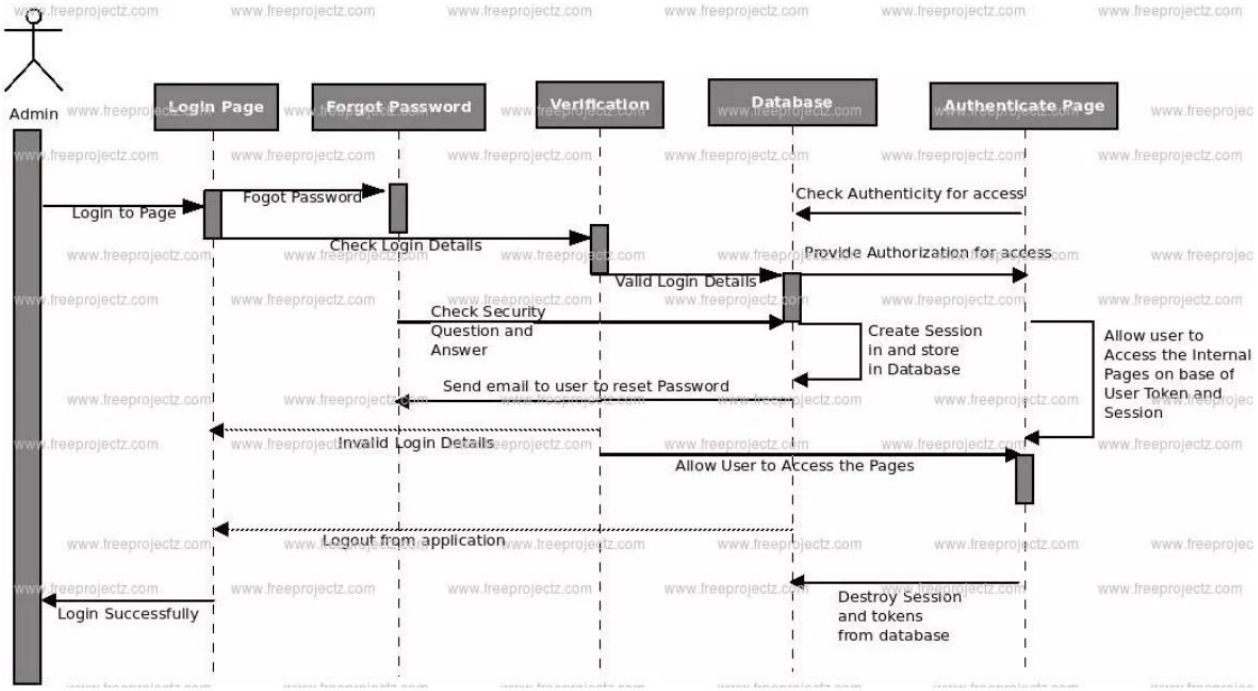
Sequence diagrams are valuable tools for understanding and visualizing the dynamic behavior of systems, particularly in scenarios where the sequence of interactions among objects is crucial to understanding system functionality and communication.

Result and Discussion:

Sequence diagram1 - Online Learning App



Sequence Diagram 2 – Online Learning App



Learning Outcomes: Students should have the ability to

LO1: Identify the classes and objects.
LO2: Identify the interactions between the objects
LO3: Develop a sequence diagram for different scenarios

Outcomes: Upon completion of the course students will be able to draw the sequence diagram for the project.

Conclusion:

Sequence diagrams in UML provide a concise visual representation of interactions between objects in a system. They illustrate the flow of messages, object lifetimes, and the sequence of operations, making them essential for understanding the dynamic behavior of software systems.

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				