Shri Ramdeobaba College of Engineering and Management Nagpur

Name- Harsh Agrawal CSE SEM-6 Roll no-43 Section-B

Subject-Software Engineering LAB.

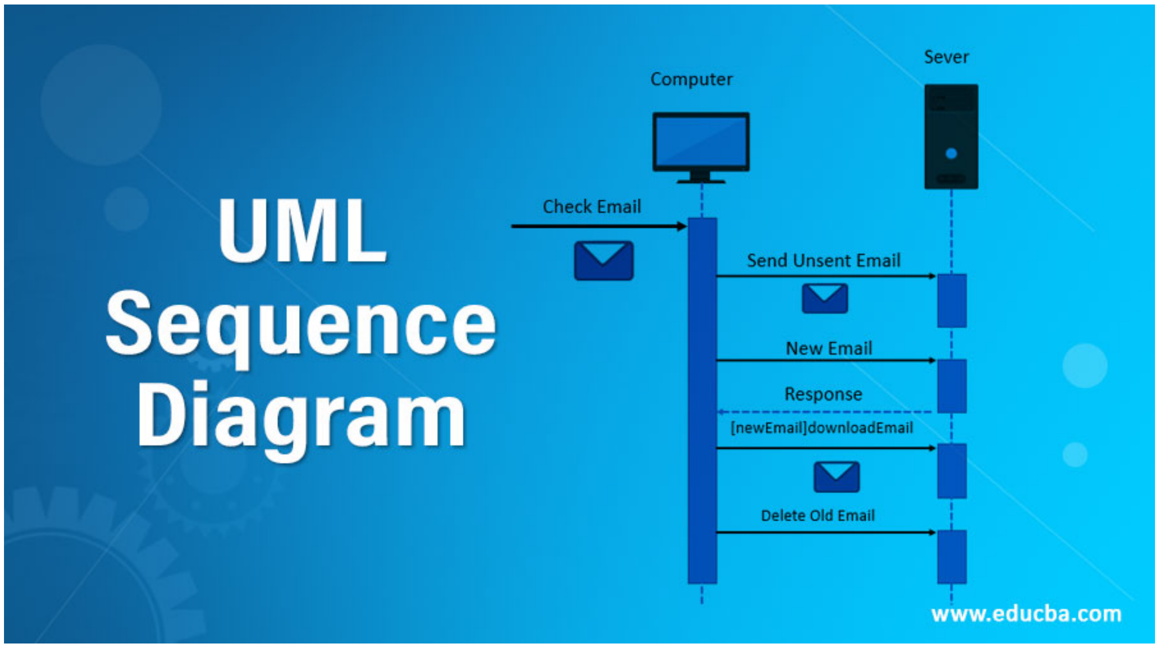
Practical no- 6

Aim→ Creating a Sequence Diagram To Implement the Employee Management System

---

Theory→ Sequence Diagram

- A sequence diagram is the most commonly used interaction diagram. Interaction diagram – An interaction diagram is used to show the interactive behavior of a system. Since visualizing the interactions in a system can be a cumbersome task, we use different types of interaction diagrams to capture various features and aspects of interaction in a system.



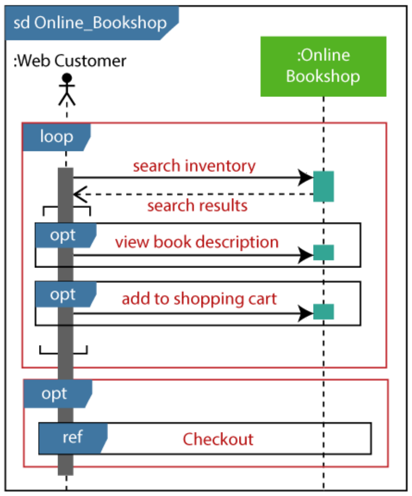
Sequence Diagrams –

- A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

- UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focused and they show the order of the interaction visually by using the vertical axis of the diagram to represent time, what messages are sent and when.
- Sequence Diagrams captures: the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams) high-level interactions between user of the system and the
- System, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams).

Uses of sequence diagrams –

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.



Sequence Diagram Notations –

  1. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram.
  2. **Lifelines –** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format – Instance Name : Class Name

3. We display a lifeline in a rectangle called head with its name and type. The head is located on top of a vertical dashed line (referred to as the stem) as shown above. If we want to model an unnamed instance, we follow the same pattern except now the portion of lifeline's name is left blank.
4. **Difference between a lifeline and an actor –** A lifeline always portrays an object internal to the system whereas actors are used to depict objects external to the system. The following is an example of a sequence diagram:

Messages – Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages from the core of a sequence diagram.

Synchronous messages – A synchronous message waits for a reply before the interaction can move forward. The sender waits until the receiver has completed the processing of the message. The caller continues only when it knows that the receiver has processed the previous message i.e. it receives a reply message. A large number of calls in object oriented programming are synchronous. We use a solid arrow head to represent a synchronous message.

Purpose of a Sequence Diagram

1. To model high-level interaction among active objects within a system.

2. To model interaction among objects inside a collaboration realizing a use case.
3. It either models generic interactions or some certain instances of interaction.

Types of fragments

Following are the types of fragments enlisted below;

Operator	**Fragment Type**
alt	Alternative multiple fragments: The only fragment for which the condition is true, will execute.
opt	Optional: If the supplied condition is true, only then the fragments will execute. It is similar to alt with only one trace.
par	Parallel: Parallel executes fragments.
loop	Loop: Fragments are run multiple times, and the basis of interaction is shown by the guard.
region	Critical region: Only one thread can execute a fragment at once.
neg	Negative: A worthless communication is shown by the fragment.

ref	Reference: An interaction portrayed in another diagram. In this, a frame is drawn so as to cover the lifelines involved in the communication. The parameter and return value can be explained.
sd	Sequence Diagram: It is used to surround the whole sequence diagram.

Benefits of a Sequence Diagram

1. It explores the real-time application.
2. It depicts the message flow between the different objects.
3. It has easy maintenance.
4. It is easy to generate.
5. Implement both forward and reverse engineering.
6. It can easily update as per the new change in the system.

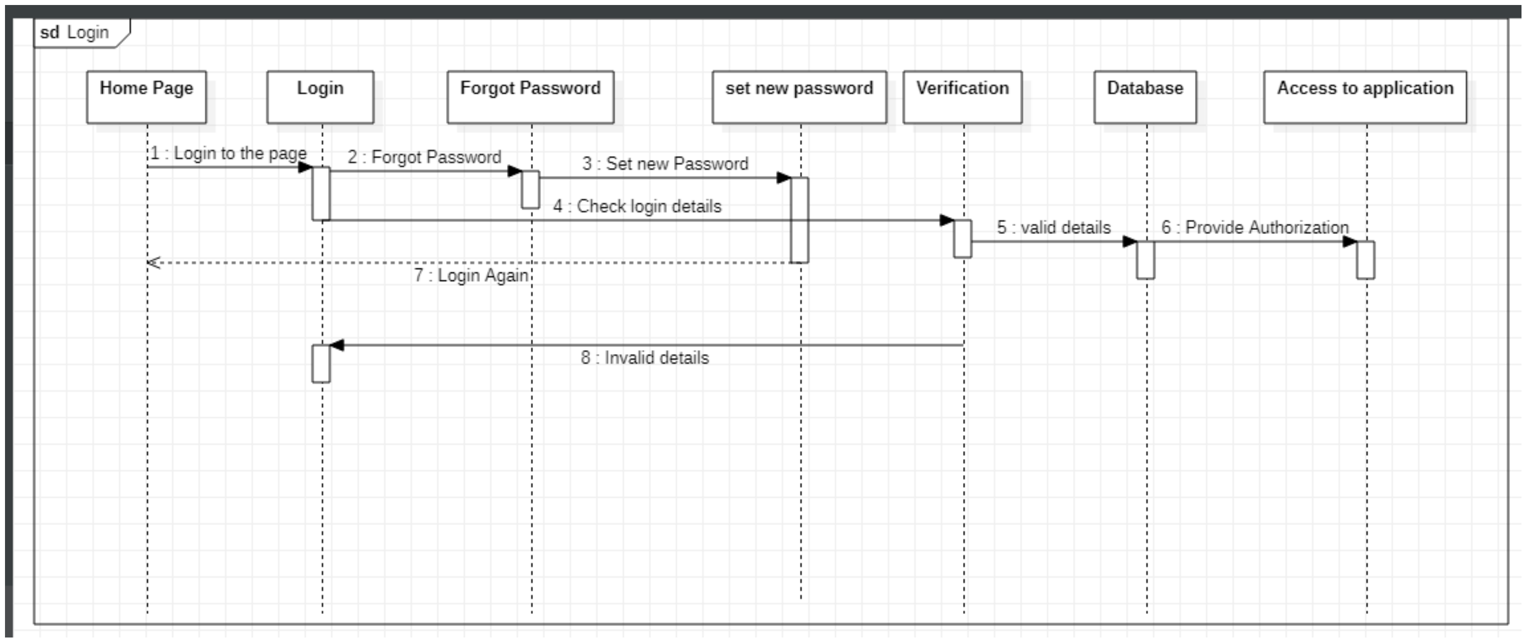
The drawback of a Sequence Diagram

1. In the case of too many lifelines, the sequence diagram can get more complex.
2. The incorrect result may be produced, if the order of the flow of messages changes.
3. Since each sequence needs distinct notations for its representation, it may make the diagram more complex.
4. The type of sequence is decided by the type of message

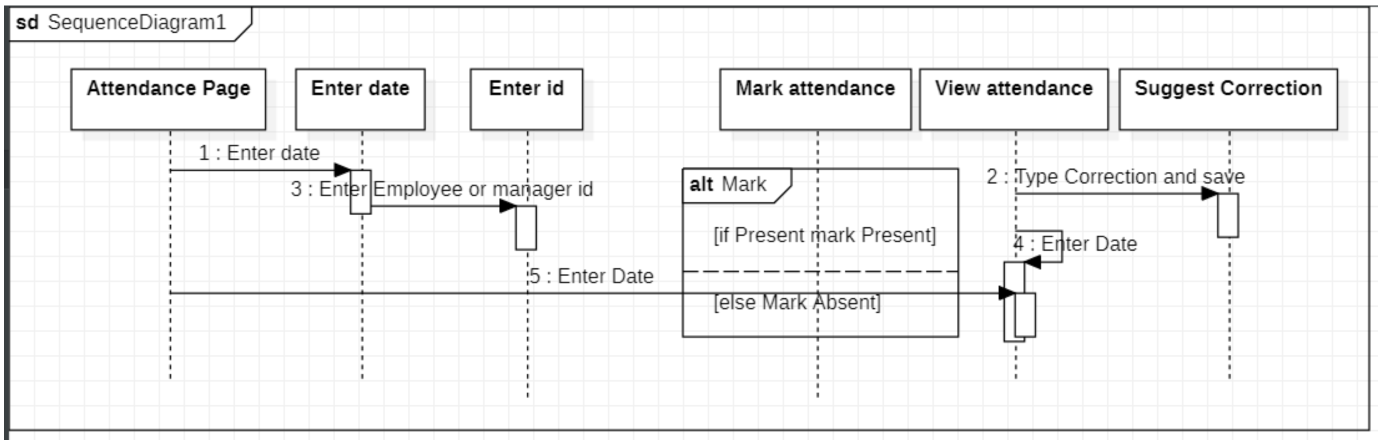
Sequence Diagram created for Employee Management system.

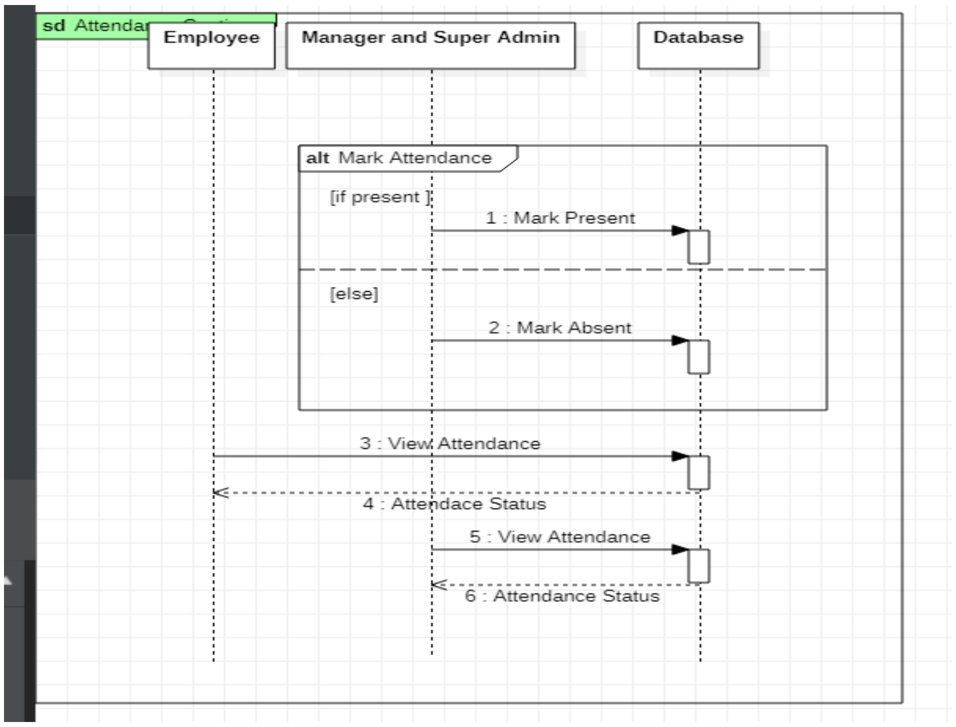
Employee Management System has several features to be used and accessed by the four actors which are Super Admin, manager, Employee and System User.

Sequence diagram for login to the application explains the process of login to the system with entering the username and password.The UI is connected with the database.After entering the login details the details would be matched with the database. Forget password option would also be available where user can set the new password.

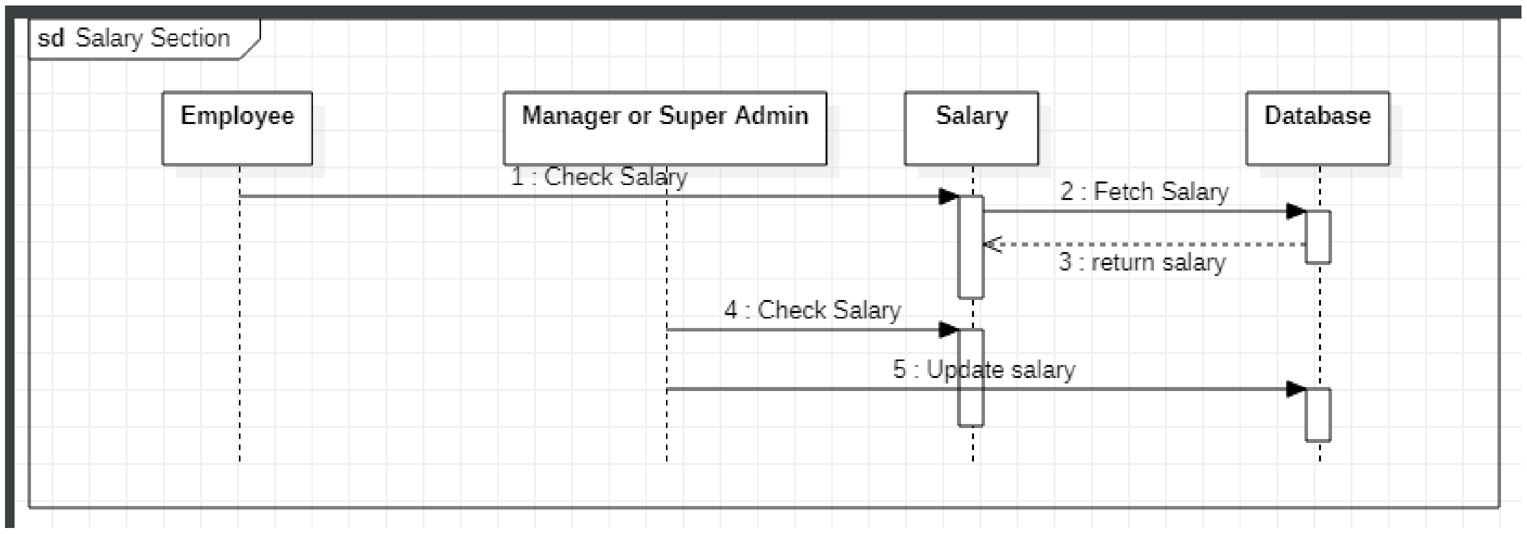


This is the sequence diagram for entering the **attendance details**. Employees can only view the attendance and suggest the corrections if any. Managers and Super admin can mark attendance of other employees.



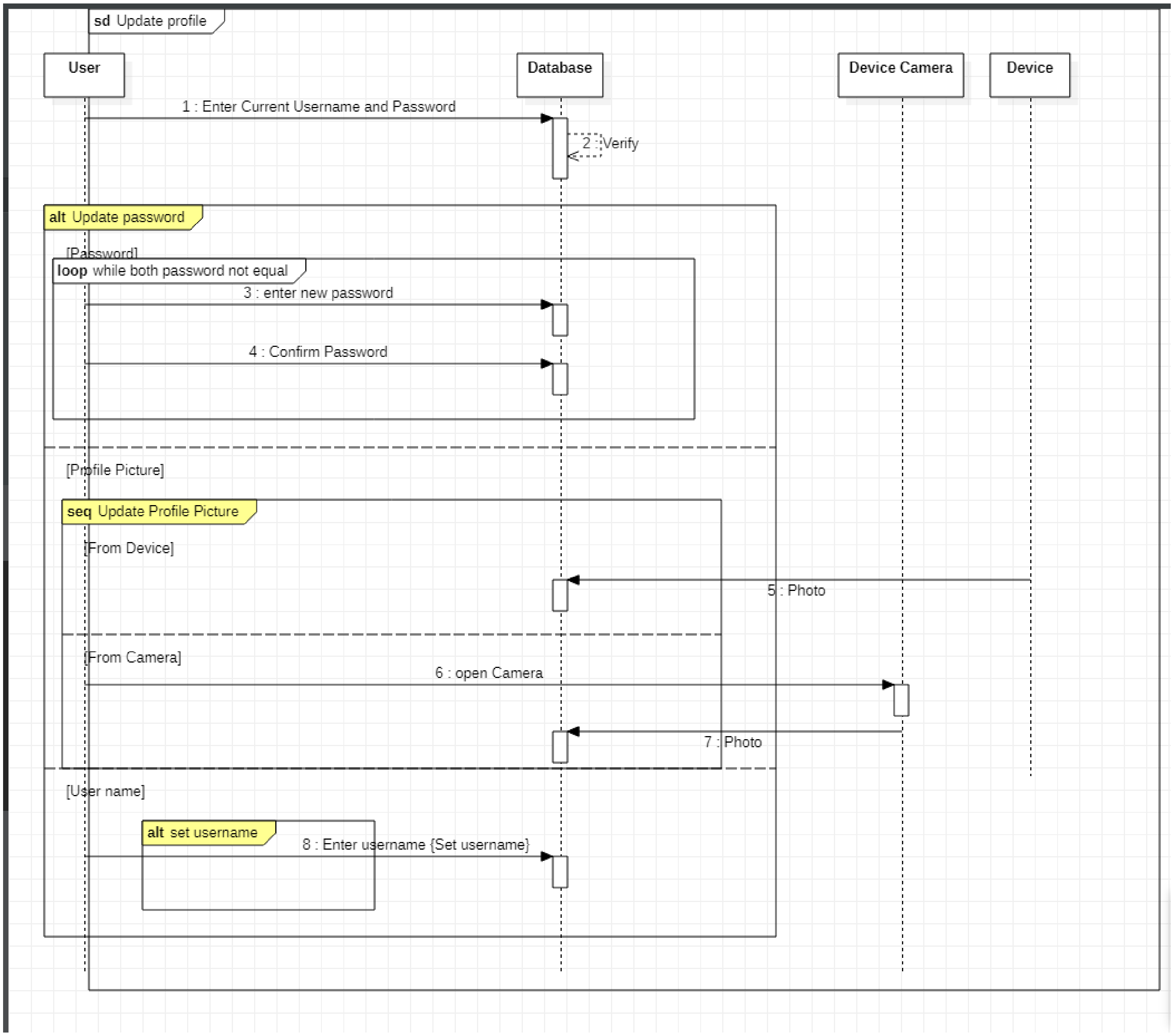


This is the Sequence diagram for the **salary section** where employees can view their salary and Managers and Supervisors can update the salary of other users.

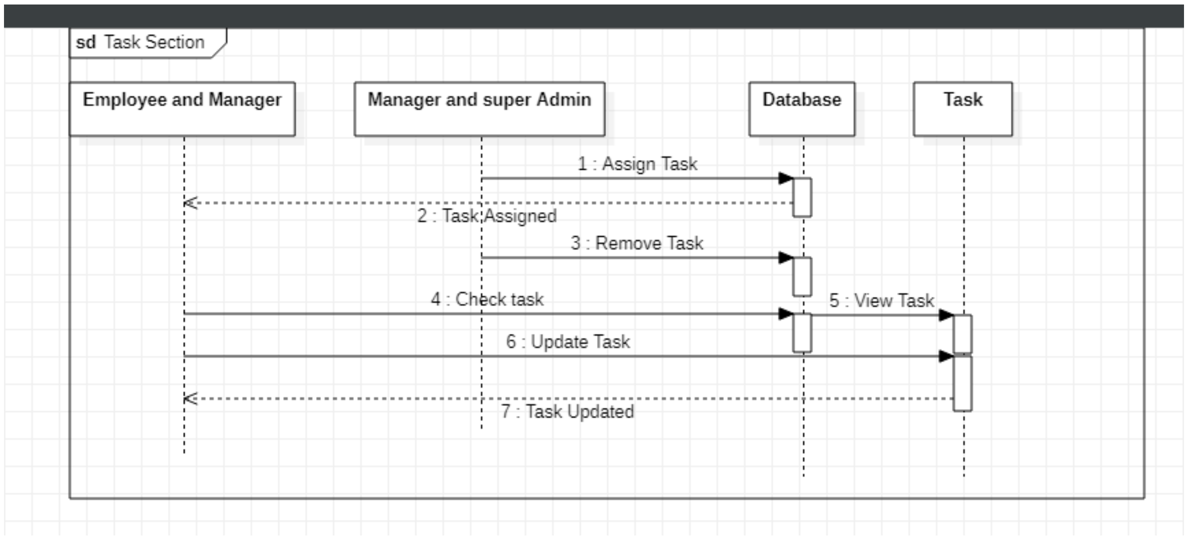


This is the **update profile** sequence diagram for the application. This feature is available for every user of the application. To update the profile the user has to first enter the current username and password if correct we can proceed further. If we select the update **password** option we need to enter the password two times and validation would be done. If both match the password would be updated. Similar is for updating **username.**

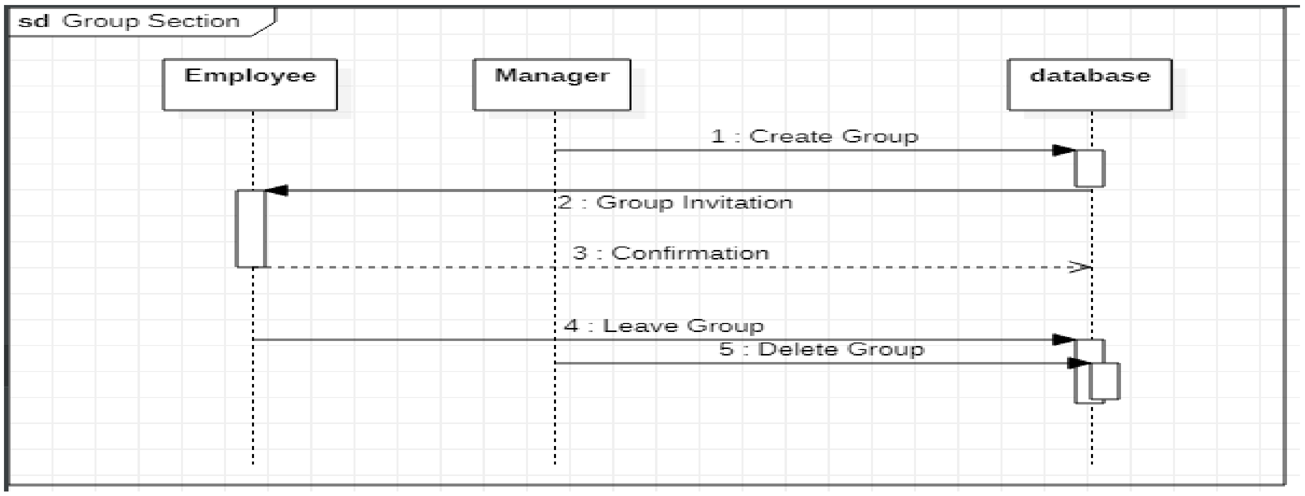
Updating **profile picture** has two options to select the picture. Firstly we can select the image from the device else we use the device camera to take a picture at runtime.



This is the **task section** for the application. Employees can directly check the task assigned to them. Managers and Super Admin can assign the task as well as delete the task.They can also make updates to the task. The same task changes would be reflected to the Employees



This is the **group section** for the Application. Managers can create groups and assign members to the group. Then the group invitation would be sent to the respective people. In return the members of the group have to send the confirmation. Members can leave the group anytime they want.



Result→ Sequence Diagram has been Studied. Sequence Diagram has been created for the Employee Management System Successfully.
