TRAINING

Preparation should be done for GoF design patterns as they solve specific problems and well-documented and well-proven solutions.
https://www.javatpoint.com/state-pattern
Major schemes:

1. creational
2. structural
3. behavioural

Lab partner: 110031302 (Arshdeep Kaur)

Let us answer all the questions:

1. **In your opinion, what kind of computational problems (representations, computations, structures, algorithms, etc) do you have to solve when you read these specifications? What details are not expressed in these specifications?**

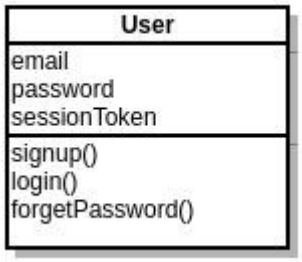
Since most of the things follow a top-down approach and methodology is clear, there is very little to solve while reading the specifications.

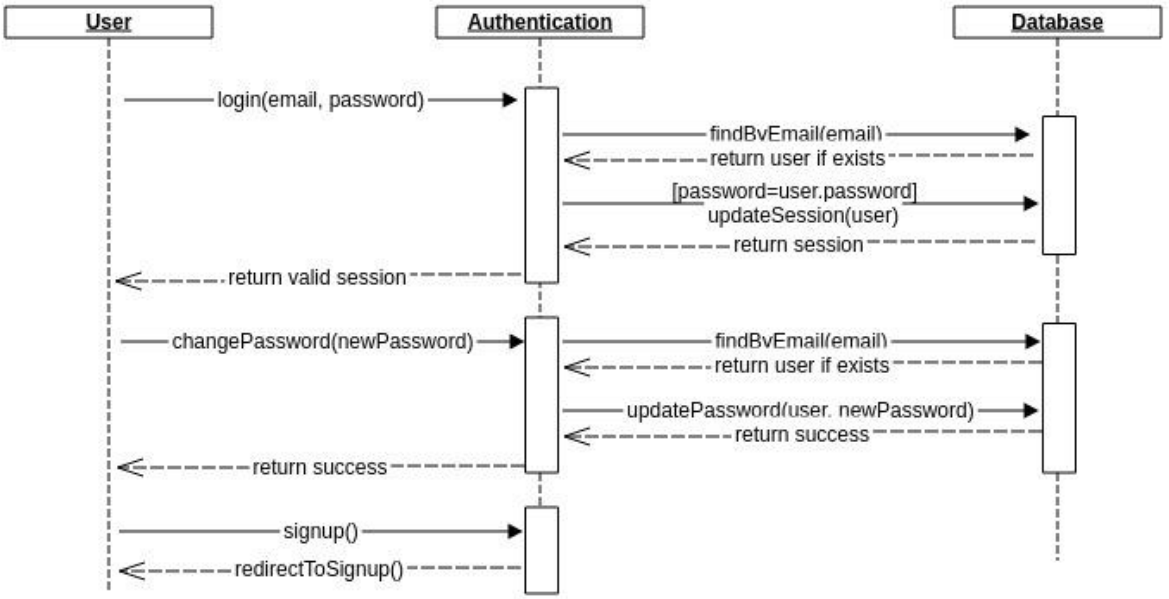
- One mismatch was found in the last requirement where user case diagram and hardware does not match with description of “Rest regime” on page#9.
- NFR contains mismatch due to mobile app
- A sequence diagram would have been easier to understand how requirements trigger the hardware/software external/internal systems.
- Little clarity on “Artificial intelligence” on page#7

2. **Rewrite the specifications from the computational point of view in an informal way.**
    a. login (has 3 alternate flows)
        i. Users can log in to the application using their email id and password
            1. input email and password
            2. output valid login session (or error in case of invalid input)
        ii. “Forget password” This feature can be used by the user in case the user forgets the password.
            1. input new password
            2. update the password for existing user
            3. output success response (or error in case of invalid input)
        iii. The "Sign Up" option allows new users to create an account.
            1. redirect to signup feature flow
    b. signup

c. home page
d. Lighting expert
e. Music of the day
f. Outfit of the day
g. Fitness regime
h. Rest regime

3. **Represent these new specifications using Class Diagrams and Sequence Diagrams. What do you notice?**





Created class diagram and sequence diagram focusing on login feature only.

a. The specifications are easier to understand and less chance of ambiguity in diagram than text
b. Output model from the last method gives rise to the input model of the new method.
c. Some implicit assumptions made on the basis of past experience.

4. **Implement with the language of your choice these diagrams without assuming more than the information represented in your diagrams. What do you notice?**

```
class User {
        String email;
        String password;
        String sessionToken;

        void signup(User userdetails);
        String login(String email, String password);
        void forgotPassword(String email, String newPassword);
}
class Authentication {
        Database db;

        User findByEmail(String email);
        String updateSession(User user);
        void updatePassword(User user, String password);
}
```

   a. Code model is based on output of methods used in previous design.
   b. Some details are found after reaching this step but cannot be changed if no assumption is made.