**Signup and Login with DB Persistence**

**Requirements**

Your need to implement complete flow for user signup and login, storing User, Session and Credentials in Database.

1. First you need to add 2 APIs in AuthController
   * /auth/signup which will accept Body in form of SignupRequest and return SignupResponse to client.
   * /auth/login which will accept Body in form of LoginRequest and return LoginResponse to client. This API will throw BadCredentialsException or UserNotSignedUpException if it will receive from Service Layer. Both are present in exceptions package. ControllerAdvisor will catch these exceptions and return desired output to client. Nothing need to be added or changed in ControllerAdvisor.
   * Both these APIs will internally call methods provided in IAuthService.
   * Please refer to dtos package for SignupRequest, SignupResponse, LoginRequest, LoginResponse.
2. You need to add implementation in AuthService's Methods as well
   * In signup method, you need to create User and AuthCredential object using parameters provided in signupRequest and persist both User and AuthCredential in DB using respective repos. After that you need to create SignupResponse object taking help from SignupRequest and return that SignupResponse Object.
   * In login method, you need to first get AuthCredential from DB corresponding to email provided in LoginRequest. If AuthCredential is not there, throw UserNotSignedUpException with Message Please sign up first !!. If password is wrong, throw BadCredentialsException with message Please provide correct password !!. After that you need to check if there is already an active session corresponding to same user, if yes, you need to delete that session from Database. Now create a new Session object, set it's TTL as current timestamp + 172800000 milliseconds. Please call given method getToken to generate token. Persist this session object into DB using SessionRepo. Now create LoginResponse and return from this method.
3. You are provided with 3 Entities - AuthCredential, User and Session. You are provided with all required fields in these classes. You just need to define relation(cardinalities) between these and make sure tables are created with following names for each Entity
   * AuthCredential -> Table should be created with name AUTH\_CREDENTIALS
   * Session -> Table should be created with name USER\_LOGIN\_SESSIONS
   * User -> Table should be created with name USERS
4. You are provided with few methods in repos which you might want to use in Service Layer.
5. You need not to add or remove dependencies in any file or change anything in ControllerAdvisor, dtos , exceptions, IAuthService, repos.

**For Reference/Testing**

Note ->

You can also check which tables with what fields are created in H2 by running Application in IntellIJ and opening http://localhost:8080/h2-console on browser and put values as below

* Saved Settings: Generic H2(Embedded)
* Setting Name: Generic H2(Embedded)
* Driver Class: org.h2.Driver
* JDBC URL: jdbc:h2:mem:class9\_ques3
* User Name: sa
* Password: password
* click Connect

**Hints**

1. If you will try run testcases without providing solution, all will fail.
2. No new File need to be created
3. No field need to be added/removed/modified in models. You just need to define Cardinalities and make sure tables are created with names given above
4. You need not to add anything in pom.xml or application.properties for DB Integration.