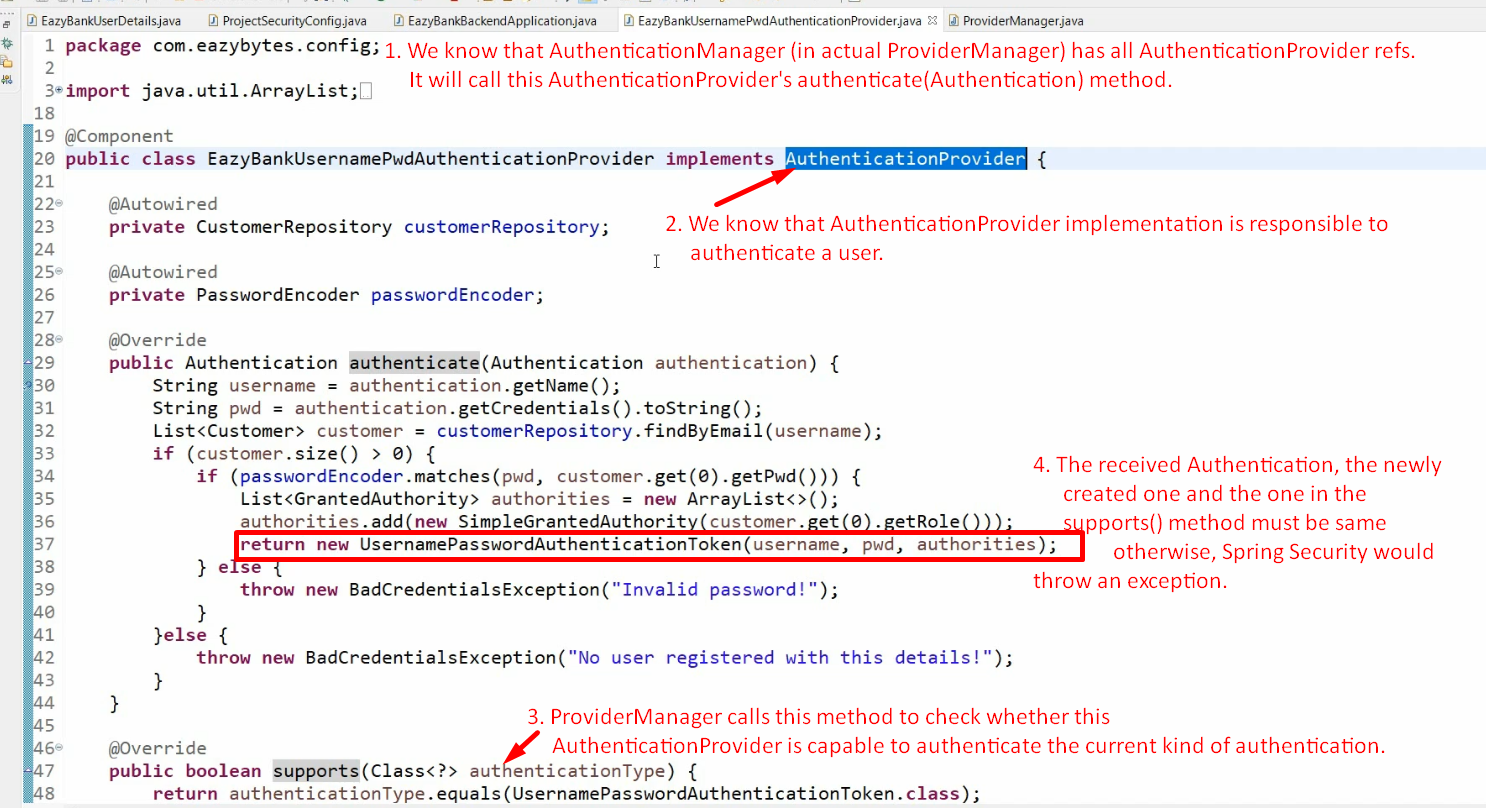
1. **Agenda**:
   1. Lets implement AuthenticationProvider and do the authentication as per our business logic.
2. 
3. **NOTE**: Don’t forget to annotate your AuthenticationProvider with @Component.   
   And Spring Boot is smart enough to identify it as AuthenticationProvider and register accordingly with AuthenticationManager.
4. 
5. What are the needs to go for your own implementations for **AuthenticationProvider**?
   1. When you have a different kind of authentication that is not supported by default Spring Security Implementations.  
      Such as Finger Print and Face Recogainzation, OTP etc.
   2. When you don’t’ want to use Spring Security implementations such as Users, UserDetails, UserDetailsManager and their implementations as you have got your own implementations such as for  
      For User, you have already existing entity class.  
      For (UserDetails, UserDetailsManager), you have JPA repository class.