**Spring Security:**

[Spring Security Custom Authentication - AuthenticationProvider vs UserDetailsService](http://stackoverflow.com/questions/31630818/spring-security-custom-authentication-authenticationprovider-vs-userdetailsser)

when you are using a different authentication system, and the password is not provided in your own database/data model, you have to use the AuthenticationProvider. for example, I've worked in a project that the customer had a centralized authentication system (CAS), so my system had no idea about the password, I had to implement the AuthenticationProvider and send the given password to the CAS, and act according to its answer.

But in another system, I was storing the password in my database, so all I had to do was implementing the UserDetailsService and check if the user exists in my database or not, spring-security had to do the rest.