SecureBankPro

SecureBankPro - A security-focused full-stack app developed during a Computer Science Academic Course.

Tech stack

- JavaScript v16.16.0
- TypeScript v5.1.6
- React v18.2.0
- Java v17
- Spring Boot v3.2.0
- Spring Security v6.2.0
- Spring Data Jpa v3.2.1
- Spring Data Redis v3.2.0
- Spring Session v3.2.0
- redis v7.2.0
- mysql v8.0.33
- modelMapper v3.2.0
- nbvcxz v1.5.1
- bouncycastle v1.70
- lombok
- docker
- nginx v1.24

Explanation of less-known libraries:

- nbvcxz - responsible for veryfing passwords and for couting password Entropy
- bouncycastle - implementation of cryptographic algorithms in java

Architecture

Explanation

- nginx - proxy server
- redis - responsible for caching sessions
- mysql - main storage

Algohritms

As a algorithm for hashing passwords, I used Argon2 algoritm, beacuse its very secure, slow hashing algorithm.

Security methods

I used sessions as the authentication mechanism. To authenticate, the user must provide a valid username and an accurate part of their password (which the system will randomly generate and prompt for each time). If all is correct, the system verifies the user's access rights and then grants them access to their account.

If the credentials are not valid, the system will not indicate whether the password or username is incorrect.

If the user provides a username that does not exist, the system will not inform them; however, it will still request a portion of their password. Obviously, user will not be authenticated.

Brute-force attacks are highly unlikely in this situation.

- If you enter your credentials incorrectly five times, your IP address will be blocked.
- There is no possibility of a timing attack. If an attacker supplies a non-existent username and then password, the server's time response will be similar to that of a valid username.

SessionID cookie is set with all appropriate flags. Expiration time for Session cookie is 15 minutes.

There is also CSRF protection.