# SYSC 4810

# Assignment 1 Report

# Arjun Pathak

# 101212581

1. RBAC is the most appropriate choice. RBAC assigns permissions to users based on their roles within the organization, aligning with justInvest's structure, which includes roles such as Clients, Premium Clients, Financial Advisors, Financial Planners, and Tellers. This model simplifies permission management by grouping permissions according to job functions, ensuring that users have access only to the information and operations necessary for their roles. Additionally, RBAC enhances security by enforcing the principle of least privilege and facilitates compliance with regulatory requirements by providing clear access controls.
2. Hg
3. Hfg

Valid User Authorization:

Test: Client views account balance.

Result: Authorized, operation successful.

Unauthorized Operation:

Test: Client tries to modify investment portfolio.

Result: Denied, operation unauthorized.

Teller Business Hours Restriction:

Test: Teller accesses system during and outside business hours.

Result: Authorized during business hours, denied outside.

Premium Client Privileges:

Test: Premium Client modifies portfolio and views Financial Planner contact.

Result: Both operations authorized.

Invalid Credentials:

Test: Invalid username or password.

Result: Authentication failed.

Role-Specific Operations:

Test: Financial Advisor accesses private consumer instruments but is denied money market instruments.

Result: Authorized for private consumer instruments, denied for money market instruments.

2.

(i). To securely store user passwords, we employ the **Argon2id** hashing algorithm, recognized for its resistance to both GPU-based and side-channel attacks. This approach aligns with best practices for password storage

**Key Parameters:**

* **Salt Generation and Length:** A unique 16-byte (128-bit) salt is generated for each password using a cryptographically secure random number generator.This ensures that identical passwords yield distinct hashes, mitigating risks associated with rainbow table attacks.
* **Hash Length:** The output hash is set to 32 bytes (256 bits), balancing security and storage efficiency.
* **Memory Cost:** Allocating 64 MiB of memory increases the difficulty of attacks by requiring significant memory resources for each hashing operation.
* **Time Cost (Iterations):** Configuring the algorithm to perform 3 iterations adds computational complexity, further deterring brute-force attacks.
* **Parallelism:** Setting the parallelism factor to 4 allows the hashing process to utilize multiple CPU cores, enhancing performance without compromising security.

By implementing Argon2id with these parameters, we ensure robust protection of user passwords against various attack vectors, adhering to industry standards for password security.

ii) To securely store user credentials, each record in the password file comprises the following components:

1. **Username:** A unique identifier for the user.
2. **Salt:** A unique, randomly generated value for each user, used to ensure that identical passwords result in distinct hashes.
3. **Hashed Password:** The result of hashing the user's password combined with the salt using the Argon2id algorithm.
4. **Argon2id Parameters:** The specific parameters used during hashing, including memory cost, time cost, and parallelism, which are essential for verifying the password.