Course Project

Skeleton Classes & Database Tables

By

Michael Smith

SWENG 837

Professor Nalubandhu

4/28/24

### Skeleton Classes

#### 1. User Class

##### Attributes:

userId (String): Unique identifier for the user.

username (String): User's login name.

passwordHash (String): Hashed password for security.

email (String): User's email address.

phone (String): User's phone number for MFA.

##### Methods

login(): Method to initiate the login process.

verifyPassword(password: String): Checks if the provided password matches the stored hash.

updateProfile(details: Map): Updates user profile information.

#### 2. Session Class

##### Attributes

sessionId (String): Unique identifier for the session.

userId (String): Associated user ID.

createdAt (Timestamp): Timestamp of session creation.

expiresAt (Timestamp): Timestamp when the session expires.

##### Methods

renew(): Extends the session's validity.

invalidate(): Marks the session as invalid.

#### 3. AuthToken Class

##### Attributes

tokenId (String): Unique identifier for the token.

associatedUserId (String): The user ID to which the token is linked.

expiresIn (Timestamp): Expiration time of the token.

##### Methods

validate(): Checks if the token is still valid.

renew(): Renews the token to extend its validity.

#### 4. MFAService Class

##### Attributes

userId (String): ID of the user for whom MFA is configured.

serviceType (String): Type of MFA (e.g., SMS, Email, Authenticator App).

##### Methods

generateCode(): Generates a new MFA code.

sendCode(): Sends the MFA code to the user's device.

verifyCode(inputCode: String): Verifies the input code against the generated code.

### Database Tables

#### 1. Users Table

##### Columns:

user\_id (VARCHAR): Primary key.

username (VARCHAR).

password\_hash (VARCHAR).

email (VARCHAR).

phone (VARCHAR).

#### 2. Sessions Table

##### Columns:

session\_id (VARCHAR): Primary key.

user\_id (VARCHAR): Foreign key linking to Users table.

created\_at (TIMESTAMP).

expires\_at (TIMESTAMP).

#### 3. AuthTokens Table

##### Columns:

token\_id (VARCHAR): Primary key.

associated\_user\_id (VARCHAR): Foreign key linking to Users table.

expires\_in (TIMESTAMP).

#### 4. MFARecords Table

##### Columns:

mfa\_id (VARCHAR): Primary key.

user\_id (VARCHAR): Foreign key linking to Users table.

service\_type (VARCHAR).

secret\_key (VARCHAR): Used for generating TOTP codes, if applicable.

#### Implementation

**Security:** Ensure password hashes and tokens are handled securely using modern cryptographic standards.

**Scalability:** Design classes and database schemas to support scalability, especially considering high user volumes and frequent access patterns.

**Maintainability:** Keep class methods focused and cohesive to facilitate easier maintenance and potential future expansions.

### Microservices and Associated Classes

#### 1. Token Service Microservice

##### Classes:

TokenManager: Handles the creation, validation, and revocation of tokens.

Attributes: None specifically stored; operates statelessly.

Methods: createToken(userId: String), validateToken(token: String), revokeToken(token: String).

##### Database Tables:

Tokens Table:

token\_id (VARCHAR): Primary key.

user\_id (VARCHAR): Associated user.

issued\_at (TIMESTAMP): Time when the token was issued.

expires\_at (TIMESTAMP): Time when the token expires.

status (VARCHAR): Current status of the token (active, expired, revoked).

#### 2. User Management Microservice

##### Classes:

UserManager: Manages user information and authentication credentials.

Attributes: None specifically stored; operates statelessly.

Methods: createUser(details: Map), updateUser(userId: String, details: Map), deleteUser(userId: String).

##### Database Tables:

Users Table:

user\_id (VARCHAR): Primary key.

username (VARCHAR).

email (VARCHAR).

password\_hash (VARCHAR).

phone\_number (VARCHAR).

status (VARCHAR): Indicates if the user account is active, locked, or suspended.

#### 3. MFA Service Microservice

##### Classes:

MFAProcessor: Manages MFA operations.

Attributes: None specifically stored; operates statelessly.

Methods: initiateMFA(userId: String), verifyMFA(userId: String, code: String).

##### Database Tables:

MFARecords Table:

mfa\_id (VARCHAR): Primary key.

user\_id (VARCHAR): Foreign key linking to the Users table.

mfa\_type (VARCHAR): Type of MFA (e.g., SMS, Email, TOTP).

mfa\_key (VARCHAR): The secret key for TOTP or reference for other types.

status (VARCHAR): Indicates the status of the MFA setup (configured, pending, disabled).

#### 4. Notification Service Microservice

##### Classes:

NotificationSender: Sends various notifications to users.

Attributes: None specifically stored; operates statelessly.

Methods: sendEmail(emailAddress: String, message: String), sendSMS(phoneNumber: String, message: String).

### Microservices Implementation

Each microservice operates independently, communicating over a network, usually through RESTful APIs or messaging systems. The data for each microservice is isolated, which increases security by limiting data exposure and reduces the risk of cascading failures.

**Access Control:** Implement strict access control measures for each microservice, ensuring they can only access the resources necessary for their operation.

**Audit Logging:** All operations, especially those that read or modify user information or authentication data, should be logged for audit purposes.

#### Summary

Based on the design and implementation specified, the system will not only be robust and secure but also ready to scale efficiently with increasing demand or evolving requirements.