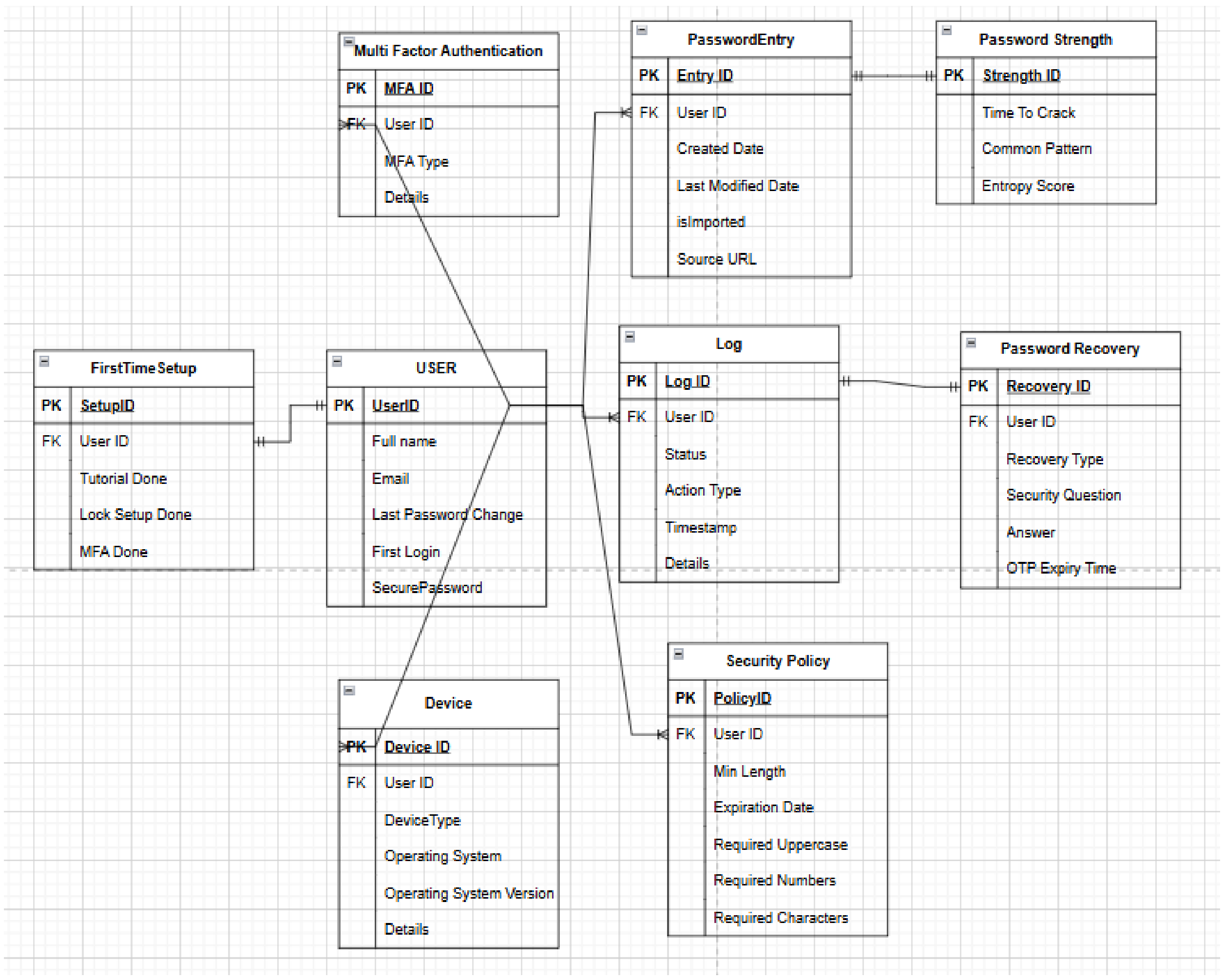
ERD



This ERD represents a password authentication tool that manages user authentication securely. At its core/center is the USER entity which stores the user's details like email and secure password. It is linked to other features such as

1 **FirstTimeSetup** records whether the user has completed the tutorial, lock setup, and multi-factor authentication (MFA).

2 **Multi Factor Authentication** stores the types and details of MFA methods associated with each user.

3 **PasswordEntry** tracks passwords used by users, including their creation date, source, and whether they were imported.

4 Password Strength is used to evaluate and store the security quality of a user's password. It includes key metrics such as time to crack.

5 **Log** records user actions, statuses, and timestamps for auditing purposes.

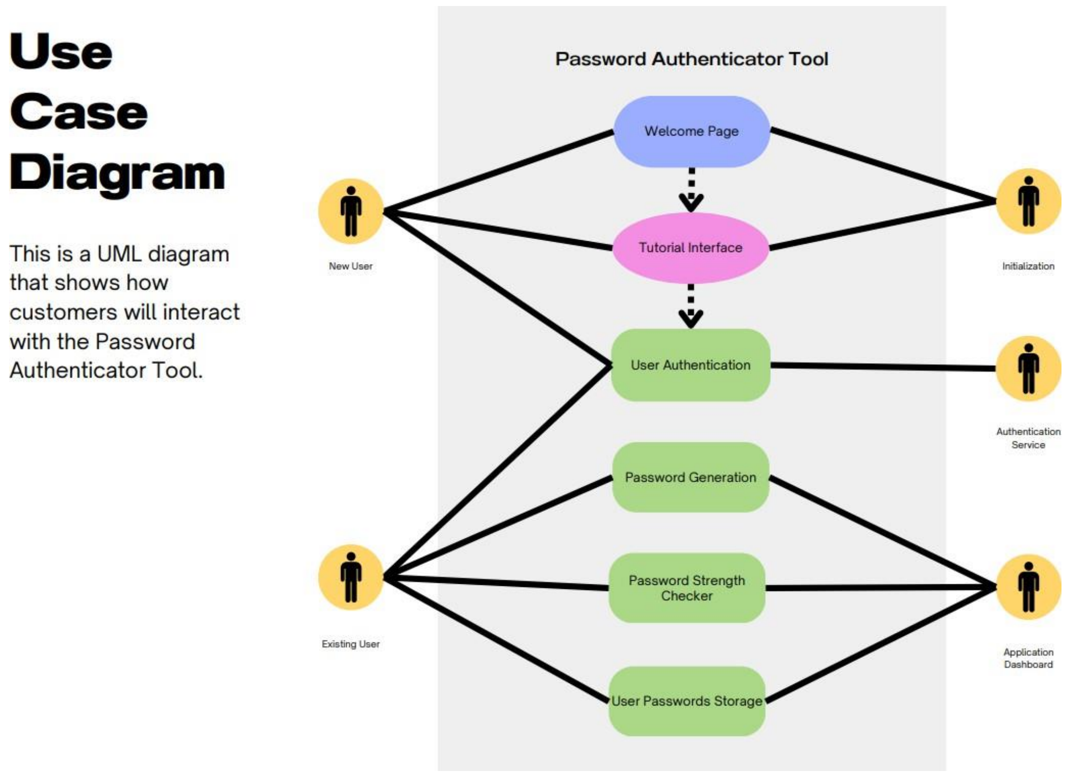
6 **Password Recovery** supports account recovery through questions, OTPs, and recovery types.

ERD

7 **Device** logs devices associated with users, including OS details.

8 **Security Policy** defines user-specific password requirements such as minimum length and character types.

Use case Diagram



Use Case Diagram

This is a UML diagram that shows how customers will interact with the Password Authenticator Tool.

Password Authenticator Tool – Use Case Description

Actors:

1. User – The sole user who sets up, accesses, and manages passwords within the app.
2. System – The Password Authenticator Tool that handles authentication, password generation, storage, and security analysis.

Use Cases and Descriptions:

1. First-Time Setup & Authentication

1. A first-time user (new user) is guided through a welcome screen along with a tutorial explaining the app's features.
2. After the tutorial, the user sets up authentication using:
1. Password
2. Passcode
3. Biometric data (fingerprint, face recognition, etc.)
3. Once setup is complete, the app locks behind the chosen authentication method.
4. An existing user only needs to enter their password, passcode, or biometric data to gain access.

2. Generate Secure Password

1. The User can request the app to generate a strong password.
2. The System creates a secure, random password based on predefined security rules (length, character types, etc.).
3. The generated password can be copied or saved within the app.

Use case Diagram

3. Store User Passwords

1. The User can save passwords for different accounts inside the app.
2. The System securely stores these passwords locally on the device using encryption.
3. Stored passwords can only be accessed after successful authentication.

4. Password Strength Checker

1. The User can enter a password to analyze its strength.
2. The System evaluates the password based on factors like:
1. Length
2. Use of uppercase/lowercase letters, numbers, and special characters
3. Predictability and common patterns
3. The app provides feedback on how strong the password is and how likely it is to be cracked.
4. If the password is weak, the System suggests improvements.

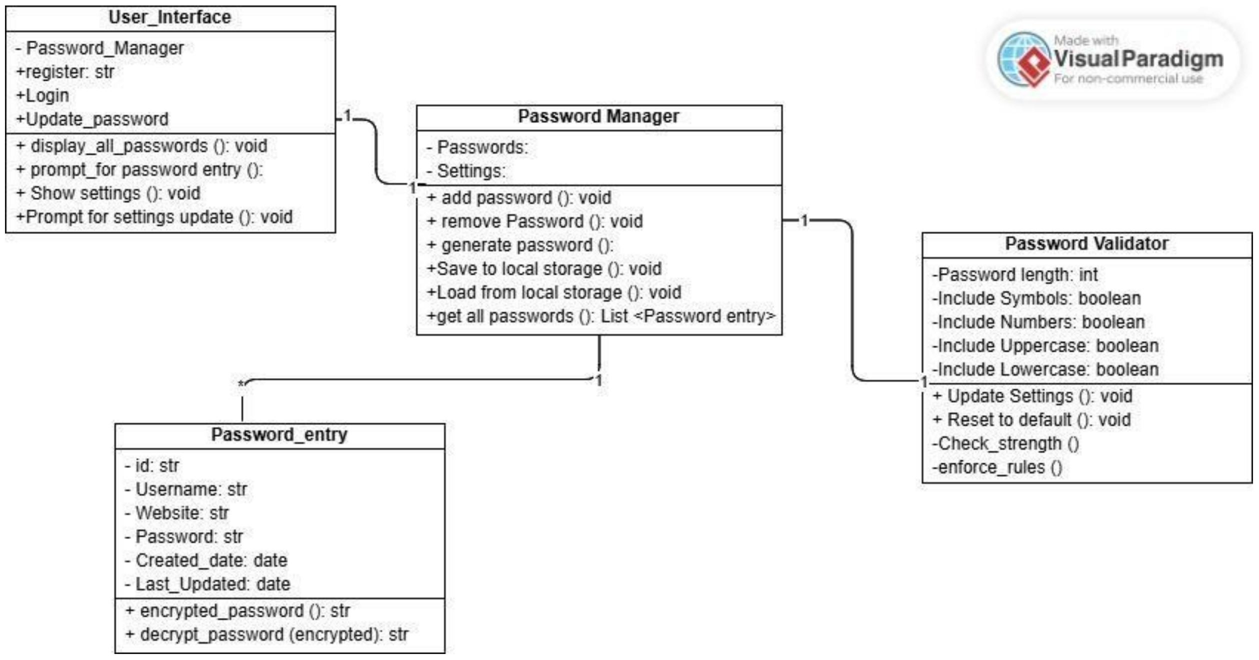
5. Reset Forgotten Password (if applicable)

1. If the User forgets their password/passcode and hasn't set up biometrics, recovery might not be possible.
2. If the app includes a recovery method, it could involve security questions or a backup code set during the first-time setup.

Relationships and Interactions:

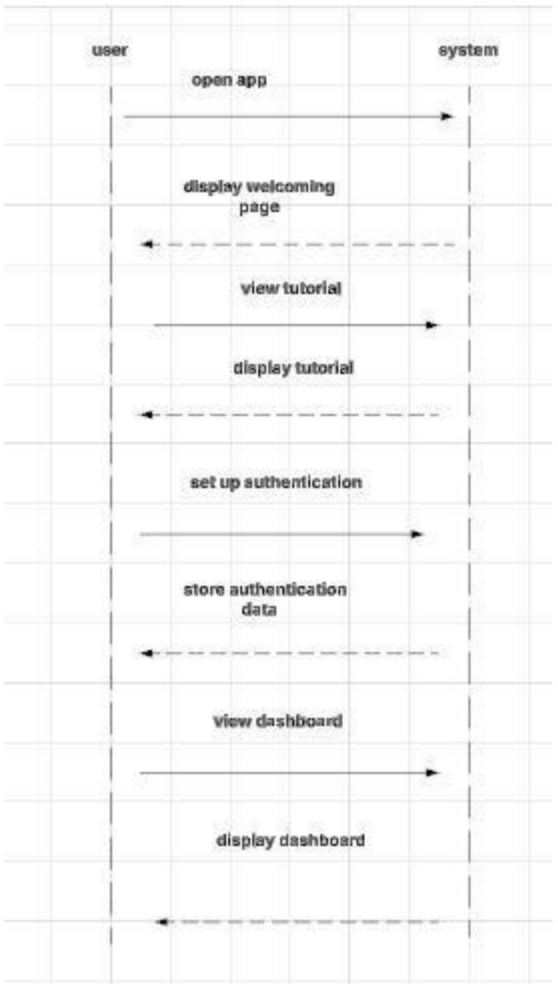
• The User interacts with all functions of the app.
• The System manages authentication, password generation, secure storage, and password strength analysis.
• The app operates offline, meaning no internet connection is required for authentication or password storage.

Class Diagram


User_Interface
- Password_Manager
+register: str
+Login
+Update_password
+ display_all_passwords (): void
+ prompt_for password entry ():
+ Show settings (): void
+Prompt for settings update (): void
Password Manager
- Passwords:
- Settings:
+ add password (): void
+ remove Password (): void
+ generate password ():
+Save to local storage (): void
+Load from local storage (): void
+get all passwords (): List <Password entry>
Password Validator
-Password length: int
-Include Symbols: boolean
-Include Numbers: boolean
-Include Uppercase: boolean
-Include Lowercase: boolean
+ Update Settings (): void
+ Reset to default (): void
-Check_strength ()
-enforce_rules ()
Password_entry
- id: str
- Username: str
- Website: str
- Password: str
- Created_date: date
- Last_Updated: date
+ encrypted_password (): str
+ decrypt_password (encrypted): str
1
1
1
1
1
*
Made with Visual Paradigm For non-commercial use

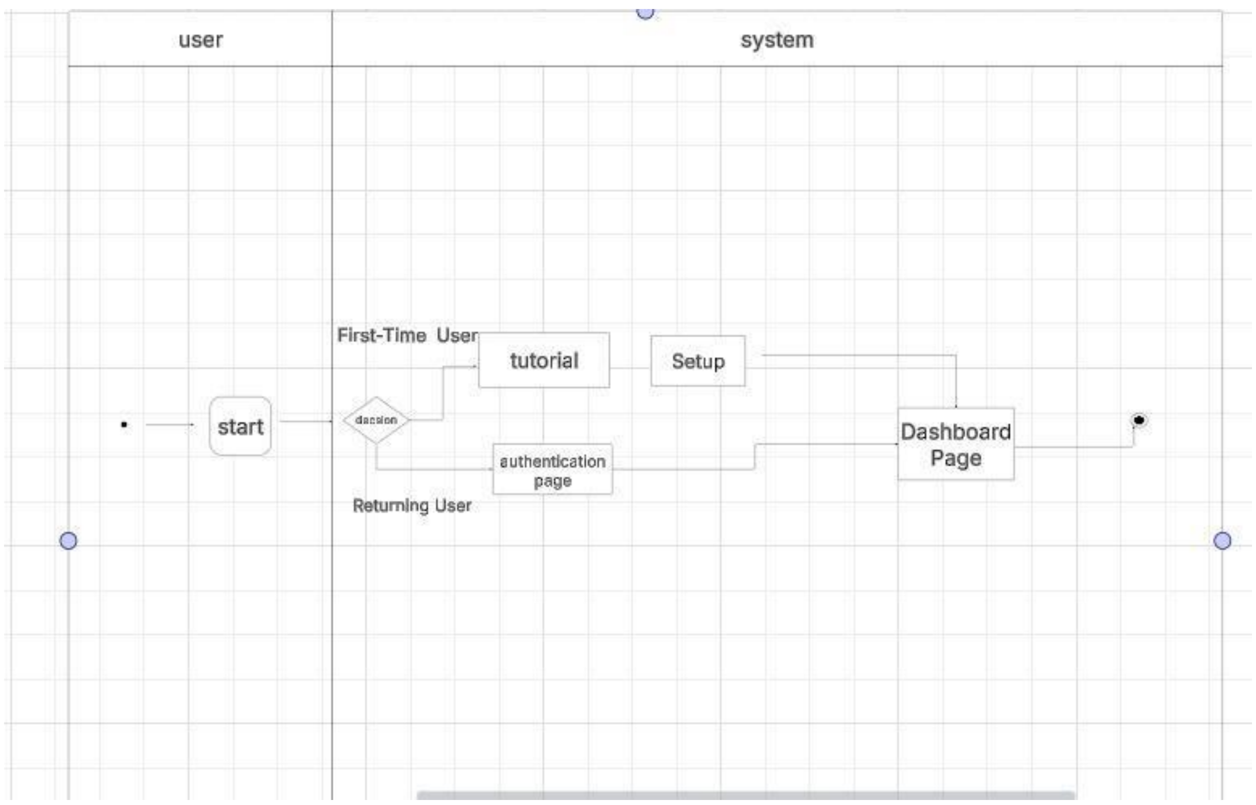
The class diagram illustrates the structural design of the Password Authenticator application, highlighting the key classes and their relationships. At the core of the system is the Password validator, which ensures password security by enforcing rules such as length requirements and the inclusion of symbols, numbers, and different letter cases. User interface: this one handle user interactions, allowing them to register, log in, update passwords, and configure settings. Password manager: manages password entries and application settings. Users can add, remove, generate, and securely store passwords locally. Password entry: Represents an individual stored password, including attributes like ID, username, website, encrypted password, creation date, and last updated date.

Sequence Diagram



This sequence diagram illustrates the initial interaction between the user and the system during the first-time setup of an authenticator app. The user begins by opening the app, prompting the system to display a welcome page followed by a tutorial. The user views the tutorial, then proceeds to set up authentication (e.g., password or biometrics). The system securely stores this authentication data. After successful setup, the user accesses the dashboard, which is rendered by the system. The flow demonstrates a clear exchange of requests and responses to guide the user through setup to main app access.

Activity Diagram



This activity diagram shows the steps a user takes when interacting with a system. First, the system checks if the user is new or returning. If it's their first time, they go through a tutorial, setup, dataflow, and authentication. Returning users skip straight to authentication. After logging in, both types of users reach either the Dashboard or a specific Page.
The diagram highlights different paths for new and existing users, showing how the system guides them before they can access the main features.