CIS 476

Term Project:

MyPass

Heba Sayed, Afrah Mohamed, Celine Chahine

Table of contents

01

Overview and Demo

What is MyPass?

MyPass is a password management software designed to securely store and manage sensitive data such as login credentials, credit card information, personal identity details, and secure notes. Users can register, create a master strong password, and manage sensitive information in a secure vault. Key features include password suggestions, secure data masking, data unmasking options, and automatic locking for inactivity. MyPass aims to enhance user security with password management while ensuring ease of use and data protection.

Demo time!

02

Technology & Contributions

What did we use?

React

For building the user interface and managing component states.

Zustand

State management solution for handling global state across the app.

Firebase

Used for user authentication and storing user data securely in the cloud.

Chakra UI

A component library for building a responsive, accessible, and customizable UI.

Contribution Breakdown

Who did what?

Heba	Authentication with Singleton Pattern, Proxy Pattern for masking/unmasking, Observer Pattern for expiration date notifier. Firebase and Repo setup. Overall app functionalities like adding, editing, and deleting data.
Celine	Chain of Responsibility Pattern for recovering password. Mediator Pattern for UI communication on login page. Database schema diagram.
Afrah	Builder pattern for creating complex passwords. Observer pattern for warning of weak password.

Contribution Breakdown

React

For building the user interface and managing component states.

Zustand

State management solution for handling global state across the app.

Firebase

Used for user authentication and storing user data securely in the cloud.

Chakra UI

A component library for building a responsive, accessible, and customizable UI.

03

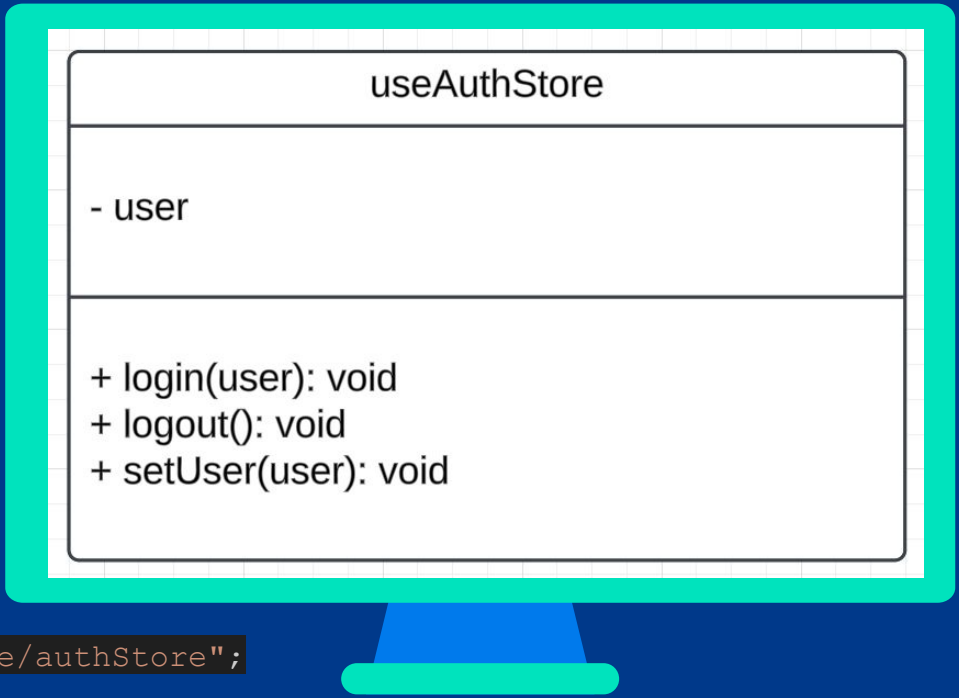
UML Diagrams

Singleton

Authentication

Zustand, by design, creates a single global store that is shared across your entire application. When we use the create function, it creates a **single instance** of the store that can be accessed via the useAuthStore hook, ensuring the state is globally accessible and consistent across the app.

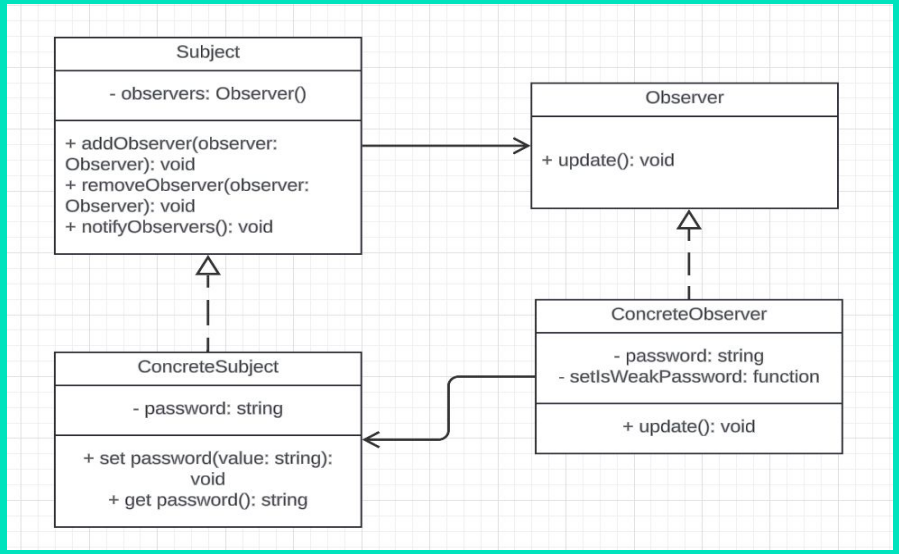
```
import useAuthStore from "../store/authStore";
```



Observer

Weak password

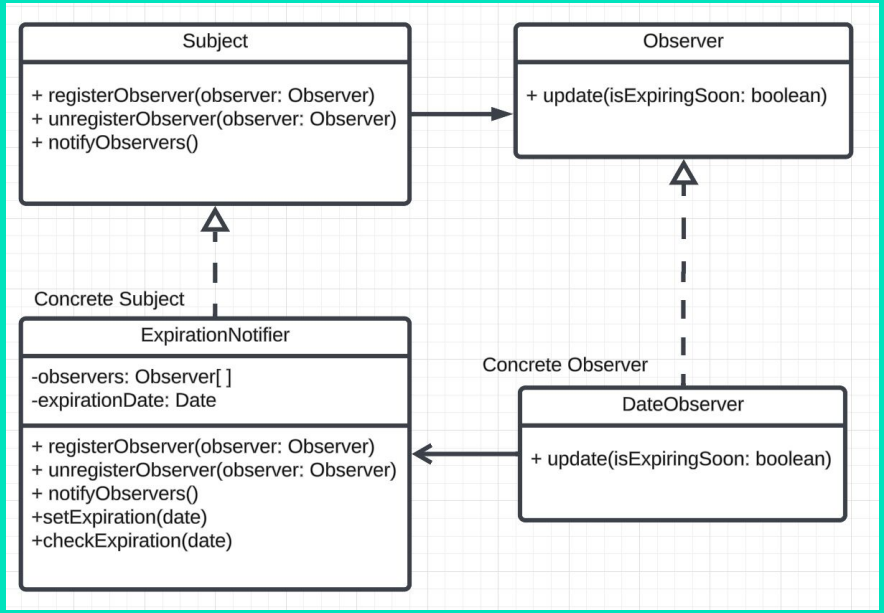
The **Observer** interface defines the *update()* methods, implemented by **ConcreteObserver**. The **Subject** class manages a list of observers, and **ConcreteSubject**, which extends **Subject**, notifies observers of state changes. The **ConcreteObserver** depends on **ConcreteSubject** for updates, and both are directly associated, with **ConcreteSubject** implementing **Subject** and **ConcreteObserver** implementing **Observer**.



Observer

Expiration Notifications

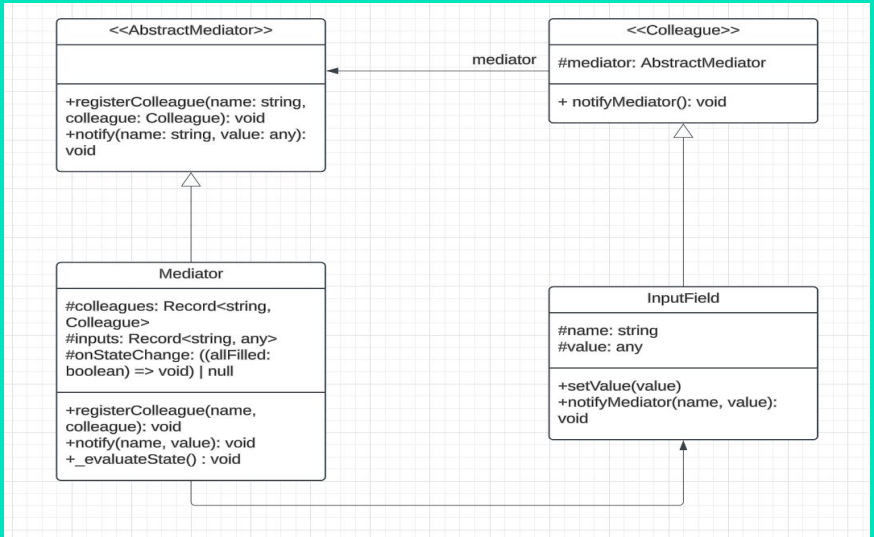
This diagram illustrates the Observer Pattern with a Subject class defining methods to manage and notify observers. ExpirationNotifier implements the subject, tracking an expirationDate and notifying observers when it changes. The Observer class defines an update() method, implemented by DateObserver, which reacts to updates using a callback.



Mediator

UI Communication

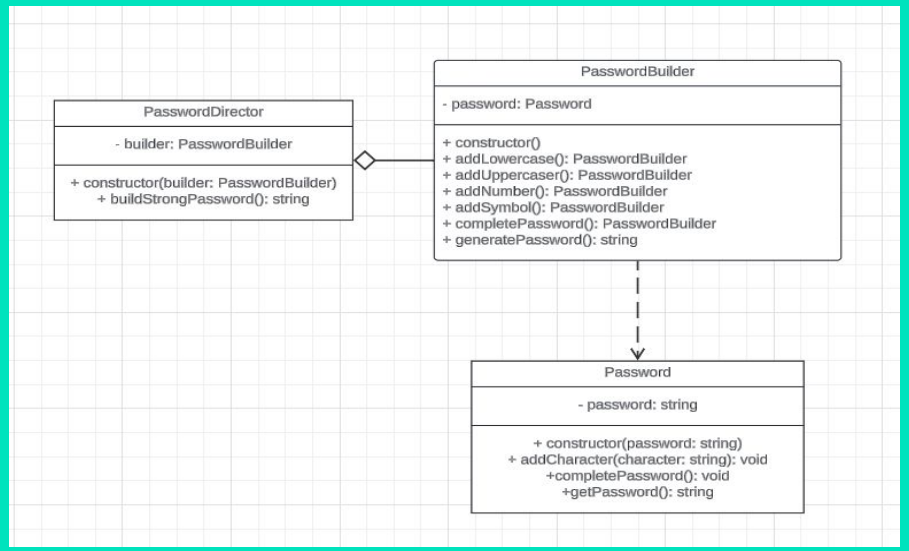
The Mediator pattern is used to coordinate interactions between input fields in a login form, ensuring the "Log in" button is enabled only when both fields are filled. The **AbstractMediator** class defines the interface, while **Mediator** handles the state of input fields and triggers updates. **InputField**, extending **Colleague**, represents individual inputs and notifies the **Mediator** when their values change.



Builder

Generate Password

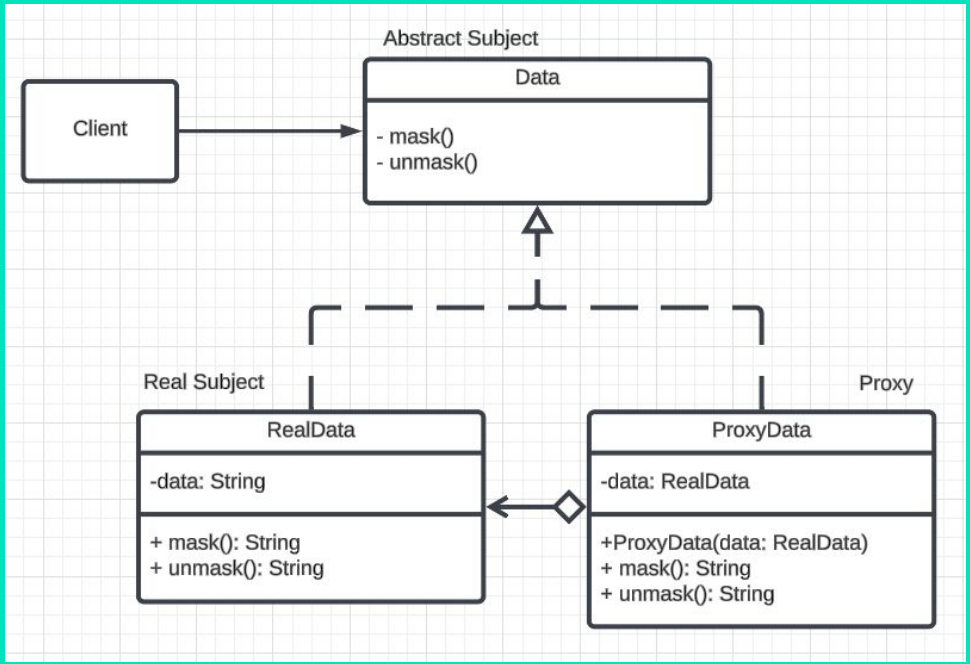
The **Password** class manages the password string and its modifications. The **PasswordBuilder** class provides methods to construct the password by adding characters such as lowercase letters, uppercase letters, numbers, and symbols. The **PasswordDirector** class oversees the **creation** of a strong password by directing the builder. The Password class has a dependency on the **PasswordBuilder**, and the **PasswordDirector** has an aggregation relationship with the **PasswordBuilder**.



Proxy

Data Masking

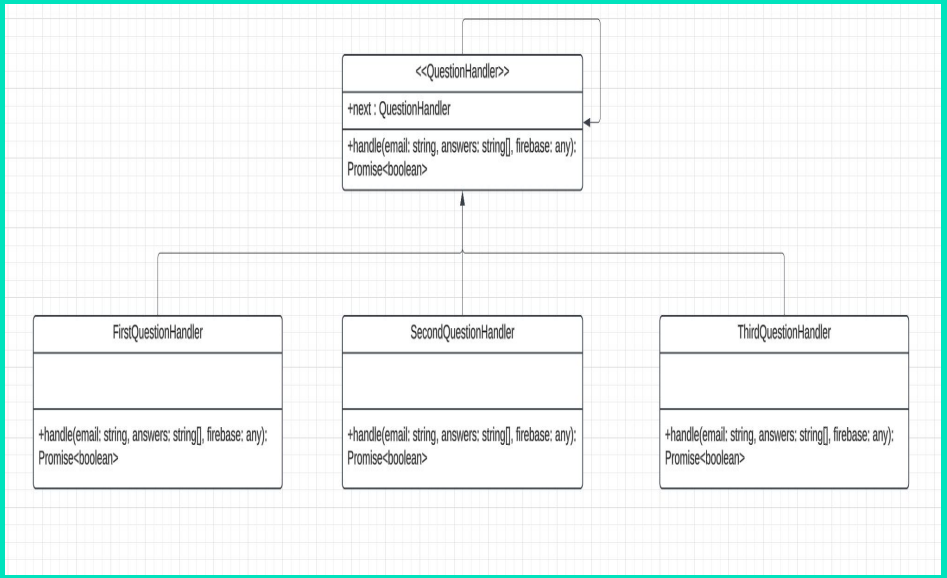
This diagram shows an abstract Data class with mask() and unmask() methods. RealData extends Data, implementing these methods by masking with asterisks and unmasking to reveal the data. ProxyData, also extending Data, acts as a proxy for RealData, delegating the mask() and unmask() calls to it, controlling access to the real data.



Chain of Responsibility

Recovery

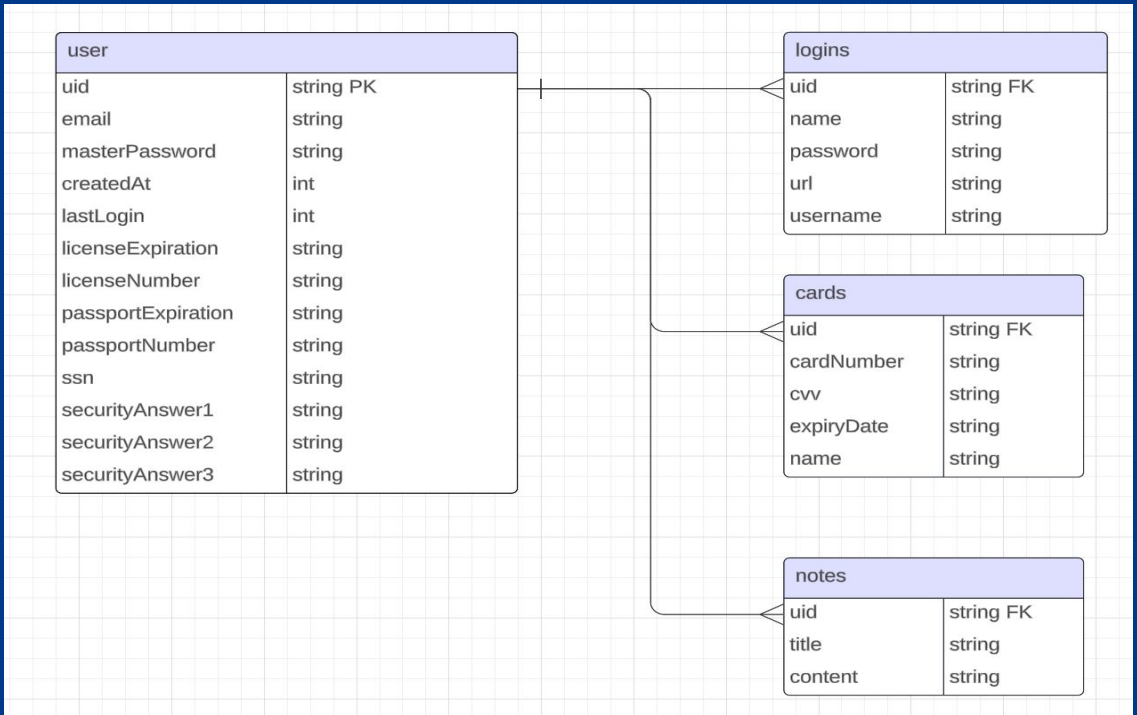
The **QuestionHandler** class is an abstract base class that sets up a structure for handling security questions during password recovery. It includes a constructor to link handlers in a chain and an abstract handle() method that must be implemented by subclasses. **FirstQuestionHandler**, **SecondQuestionHandler**, and **ThirdQuestionHandler** extend QuestionHandler and implement handle() to validate answers against specific questions in Firebase. Each handler checks an answer and, if correct, passes processing to the next handler. If an answer is incorrect, the handler logs an error and returns false, stopping the chain.



04

Database Schema

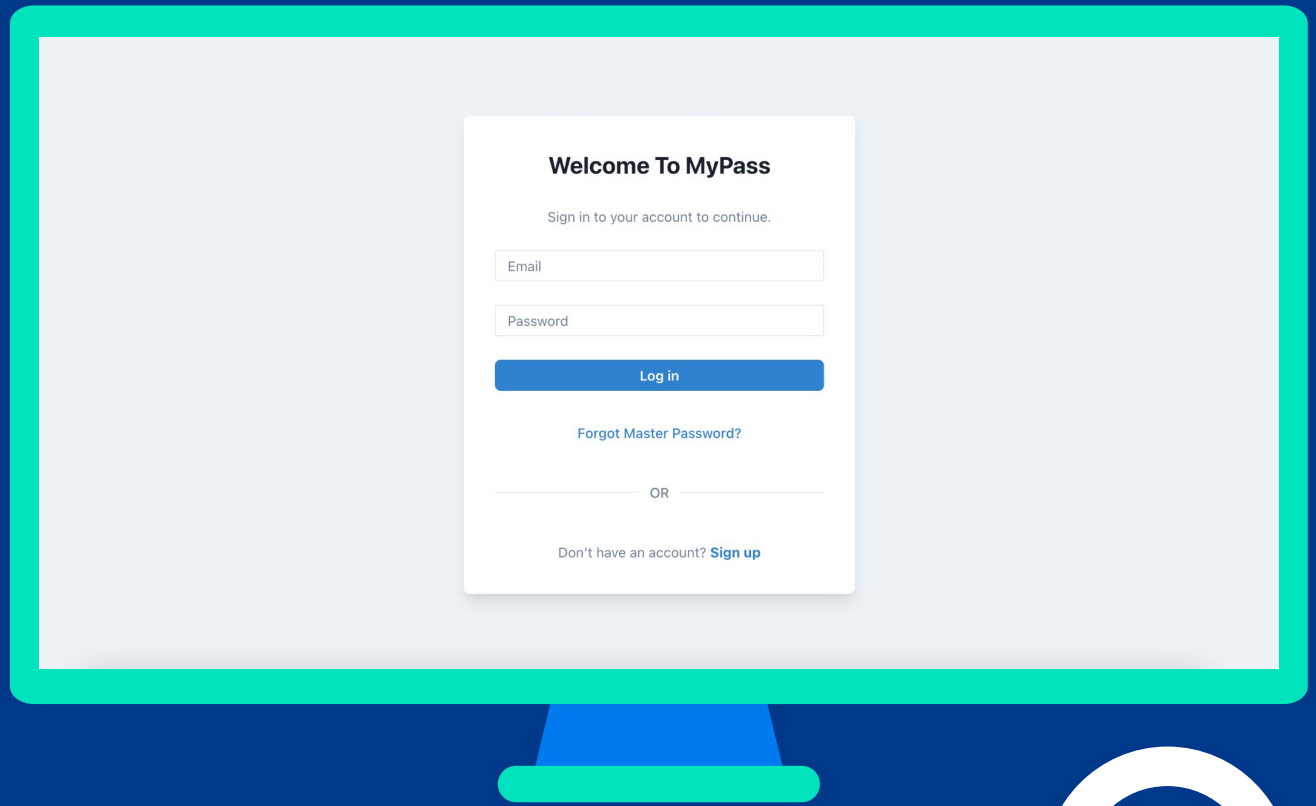
Schema


user
uid string PK
email string
masterPassword string
createdAt int
lastLogin int
licenseExpiration string
licenseNumber string
passportExpiration string
passportNumber string
ssn string
securityAnswer1 string
securityAnswer2 string
securityAnswer3 string
logins
uid string FK
name string
password string
url string
username string
cards
uid string FK
cardNumber string
cvv string
expiryDate string
name string
notes
uid string FK
title string
content string

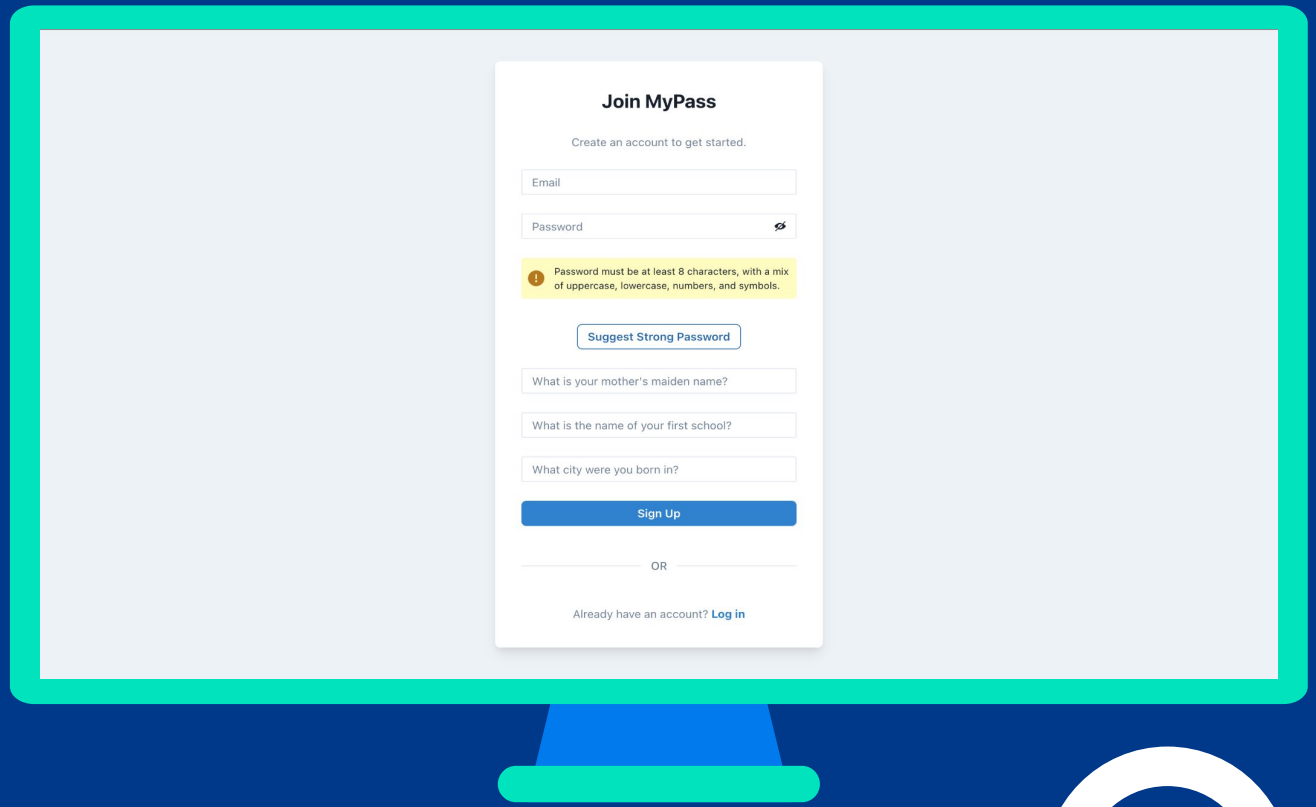
05

Interface Screenshots

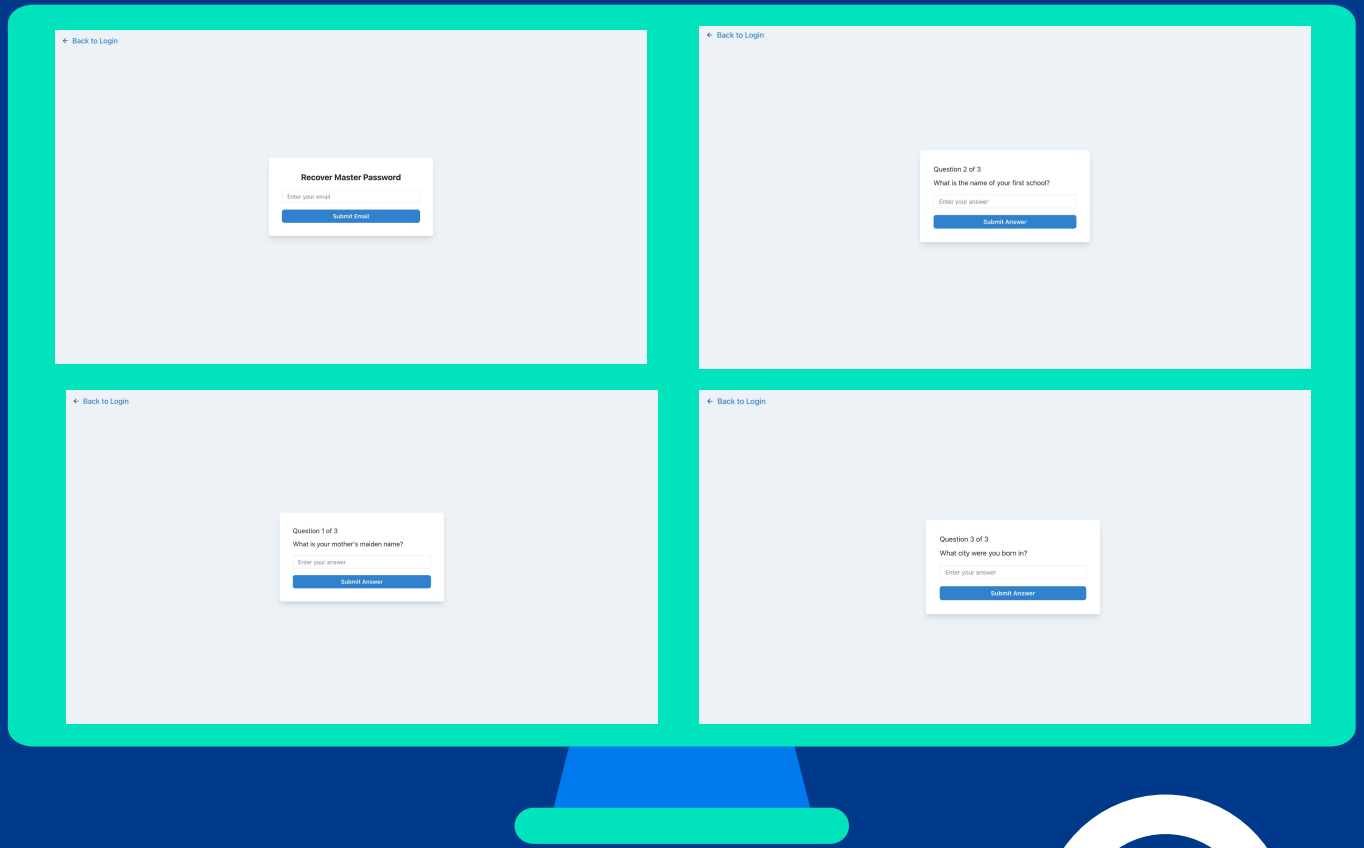
Login Screen



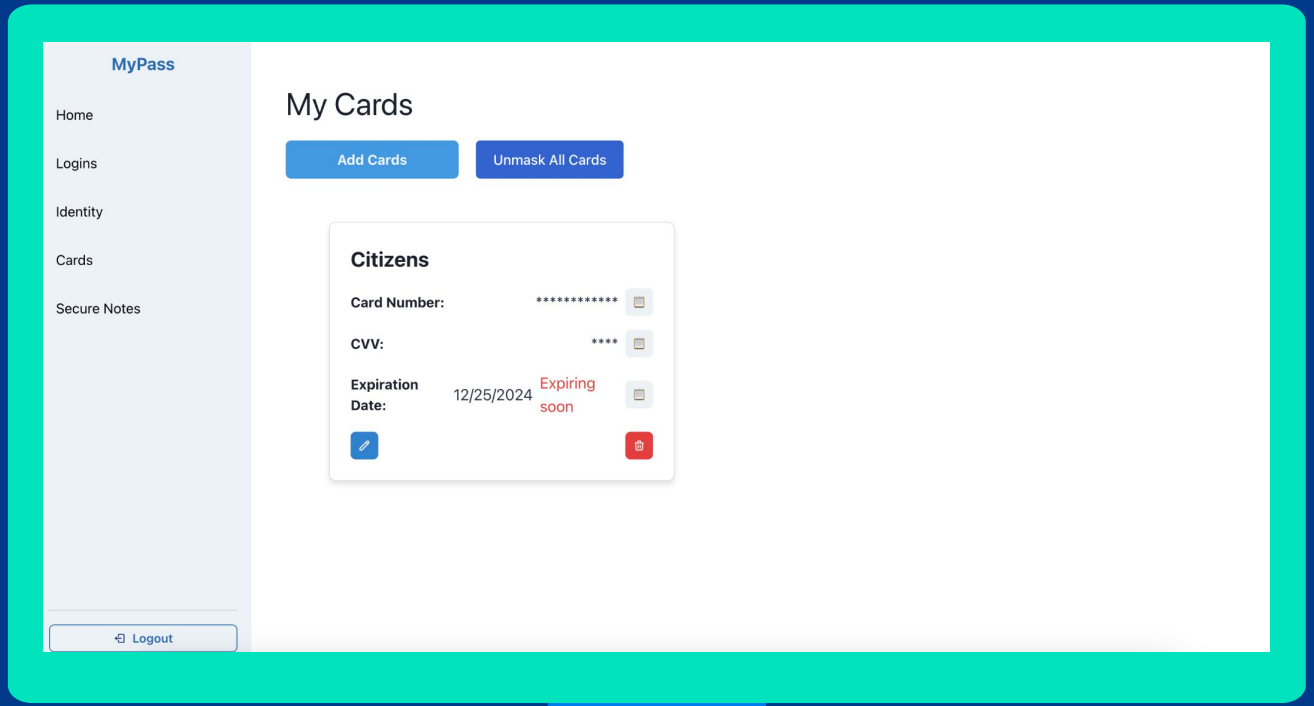
Register Screen



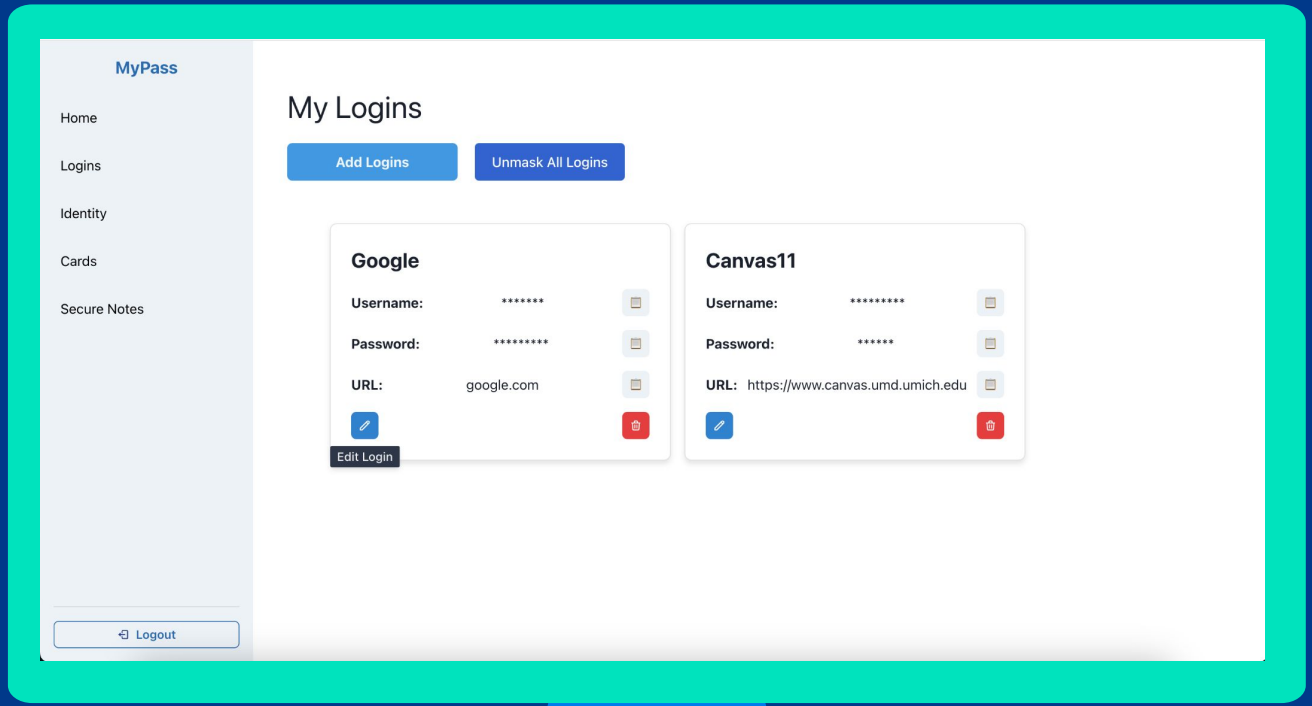
Recovery Screen



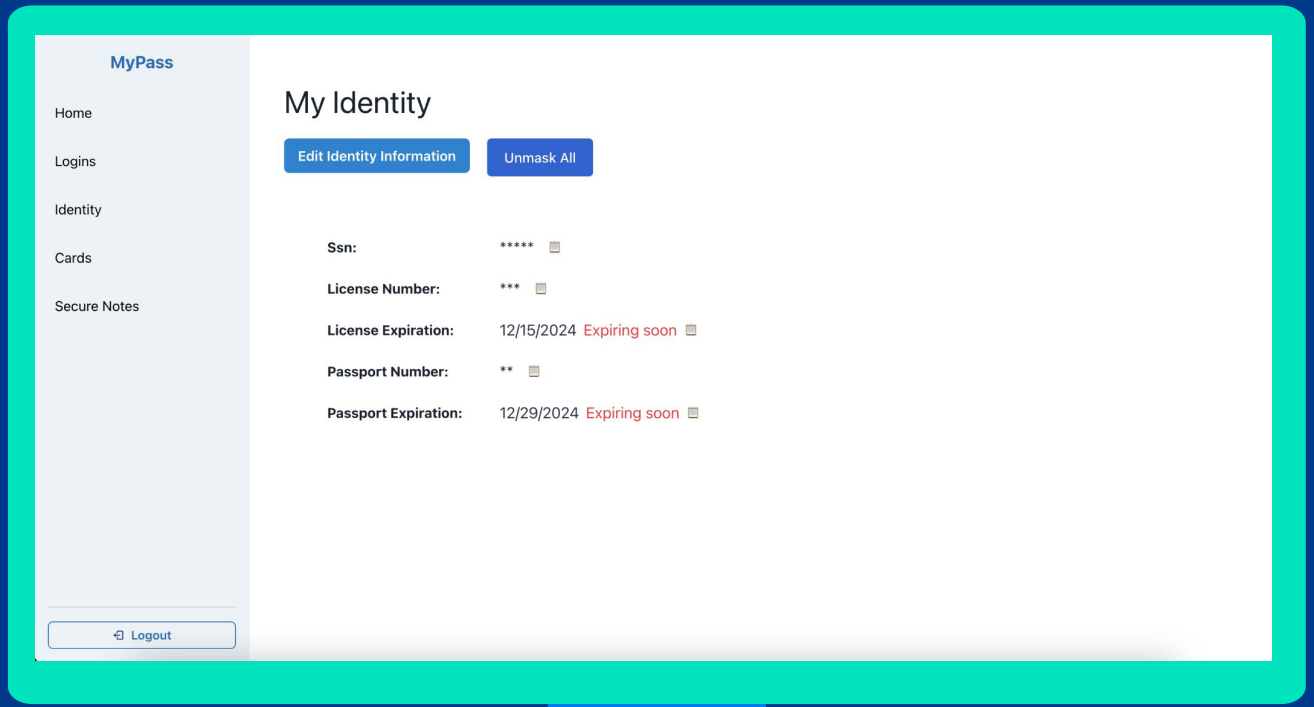
Cards Page


MyPass
Home
Logins
Identity
Cards
Secure Notes
Logout
My Cards
Add Cards
Unmask All Cards
Citizens
Card Number:
****
CVV:
****
Expiration Date:
12/25/2024
Expiring soon

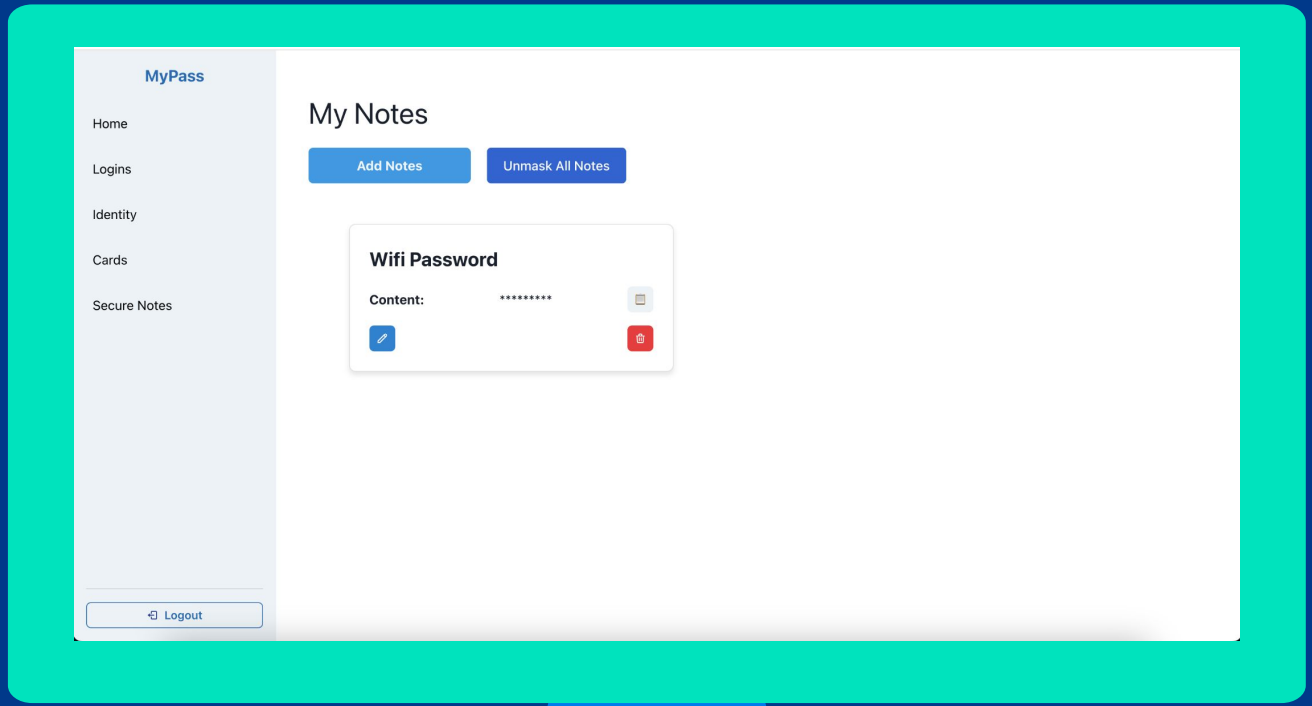
Logins Page


MyPass
Home
Logins
Identity
Cards
Secure Notes
Logout
My Logins
Add Logins
Unmask All Logins
Google
Username: *******
Password: *********
URL: google.com
Edit Login
Canvas11
Username: *********
Password: ******
URL: https://www.canvas.umd.umich.edu

Identity Page



Notes Page


MyPass
Home
Logins
Identity
Cards
Secure Notes
Logout
My Notes
Add Notes
Unmask All Notes
Wifi Password
Content:
****

Thank You!