Enhancing Cybersecurity in Educational Institutions:

A Comprehensive Study on Multi-Factor and Passwordless Authentication Methods

Mohamed El Bachrioui

ST20257755 - CIS6006 – Cyber Security and Cryptography

Table of Contents

1. Introduction

With technology advancing rapidly, it's essential to continuously enhance cybersecurity measures and data management processes to stay ahead of new threats and the growing volume of data. In the education sector, where sensitive student information is handled daily, maintaining strong security and efficient data access is crucial. This study details the security implementation of a revamped application aimed at enhancing multi-factor authentication (MFA) and improving access to student records. The focus is on implementing both password-less and password-based authentication systems, alongside utilizing local or on-premises data storage solutions.

Background

Educational institutions manage extensive archives containing students' personal and academic information. Quick retrieval and secure access to these documents are essential for various stakeholders, including students, faculty, and administrative staff. Traditionally, password-based authentication has been used to secure access to these records. However, the increasing sophistication of attacks and common password management issues necessitate a shift to more robust and user-friendly authentication methods.

Objectives

The main objectives of this project are:

1. **Improving Authentication Security:** Introduce advanced authentication techniques to reduce reliance on passwords and minimize the risks associated with password breaches.

2. **Enhancing User Experience**: Simplify the authentication process to ensure quick and easy retrieval of student records while maintaining strong security measures.

3. **Ensuring Data Integrity and Availability:** Implement robust local storage solutions to enhance data security and scalability, ensuring the continuous availability and integrity of student records.

Scope

The scope of this project includes:

- **Authentication Redesign**: Developing and integrating both password-less and password-full authentication methods into the existing application. This will involve using technologies like biometric authentication, magic links, one-time passwords (OTPs), and hardware tokens for password-less methods, while also strengthening traditional password-based systems with robust password policies and two-factor authentication (2FA).

- **Data Storage Optimization**: Implementing robust local storage solutions to replace traditional on-premises systems, thereby enhancing data accessibility, security, and performance.

Rationale

Traditional password-based systems have numerous vulnerabilities, including weak passwords, password reuse, and phishing attacks. Password-less authentication presents a promising alternative by eliminating passwords and using more secure and convenient methods like biometrics and OTPs. However, transitioning to these new methods requires careful consideration of user acceptance and technical integration challenges.

At the same time, the increasing volume of student data and the need for rapid access necessitate a re-evaluation of current data storage solutions. While cloud storage is a popular option, concerns about data privacy and control make local storage a more appealing choice for educational institutions. By optimizing local storage solutions, we can achieve a balance of security, performance, and cost-effectiveness.

Methodology

This project uses an Agile methodology to facilitate iterative development and continuous feedback. Key phases include requirement analysis, system design, development, testing, deployment, and review. By adopting Agile practices, the project aims to remain flexible and responsive to stakeholder needs and technological advancements.

Expected Outcomes

The successful implementation of this project is expected to result in:

- **Increased Security**: By incorporating advanced authentication methods, the application will be more resilient to cyber threats.
- **Enhanced User Experience**: Users will benefit from a more seamless and efficient authentication process, reducing the barriers to accessing necessary information.
- **Improved Data Management**: The transition to robust local storage solutions will enhance the reliability and performance of data access, ensuring that student records are readily available when needed.

This report provides a detailed account of the implementation process, the challenges encountered, and the solutions developed to achieve these objectives. It includes a literature review on password-less and password-full authentication methods, an explanation of the System Development Life Cycle (SDLC) method used, a diagrammatic representation of the system architecture, comprehensive app documentation, and a reflection on the project's outcomes and future recommendations.

2. Problem Identification

Multi-factor Authentication (MFA) Challenges

Traditional Password-Based Authentication Vulnerabilities

Traditional password-based authentication systems are becoming increasingly vulnerable to security breaches. Common problems associated with these systems include:

- **Poor Password Practices**: Users often create weak passwords that are easy to remember but also easy to guess. Additionally, password reuse across multiple accounts makes security risks even worse. If a password is compromised, it can be used to access multiple systems and services.
- **Phishing Attacks**: Cybercriminals often use phishing tactics to trick users into revealing their passwords. Despite awareness campaigns, phishing remains a significant threat due to its evolving sophistication.
- **Brute Force and Dictionary Attacks**: Attackers use automated tools to guess passwords by trying numerous combinations. Weak and commonly used passwords are particularly vulnerable to these types of attacks.

Multi-Factor Authentication (MFA) Implementation

Although MFA greatly improves security by requiring several kinds of verification, it also brings some challenges

- **User Friction**: MFA can complicate the login process, necessitating users to input codes from their mobile devices, respond to security questions, or use biometric scans. This added complexity can frustrate users and hinder their productivity.
- **Access Delays**: The extra steps involved with MFA can slow down access to vital resources. In an educational setting, this can affect how quickly professors and administrative staff can retrieve student records.
- **Usability Challenges**: Problems may arise when users can't access their secondary authentication factor, like a phone, or when they're in an area where they can't use their secondary authentication method (e.g., no internet access for receiving one-time passwords).The goal of this project is to implement a more efficient and user-friendly authentication system that retains the security benefits of MFA while minimizing user friction and access delays.

Slow Access to Student Records

On-Premises Data Storage Limitations

Many educational institutions frequently depend on on-premises data storage systems for the purpose of managing student records. Although this strategy provides the advantage of direct data control, it also poses other substantial challenges:

- **Slow Access Times**: On-premises systems may have delays in accessing data, especially when there is high demand. Sluggish access speeds can cause a delay in retrieving student records, which can have a negative influence on administrative efficiency and the overall user experience.
- **Limited Scalability**: Expanding on-premises infrastructure to handle increasing data volumes and user requirements necessitates significant investment in hardware and upkeep. Both the financial and logistical aspects of this can pose significant challenges.
- **High Maintenance Costs**: On-premises storage systems necessitate continuous maintenance, encompassing hardware repair, software upgrades, and security fixes. These duties require substantial IT resources and can detract attention from other crucial activities.

Security and Performance Enhancement Goals

In order to overcome these constraints, the project seeks to improve the efficiency and safety of data storage solutions located on-site, without the need to transfer the data to cloud-based systems.The focus is on:

- **Optimized Data Access**: The primary emphasis is on optimising data access by using strategies that streamline the retrieval operations, resulting in reduced access times and improved efficiency in data handling.
- **Enhanced Security Measures**: Guaranteeing the protection of data by implementing strong encryption, strict access limits, and conducting frequent security audits. This encompasses safeguarding against physical security risks and preventing unauthorised entry.
- **Cost-Effective Scalability**: Implementing cost-effective techniques for expanding storage solutions, such as utilising network-attached storage (NAS) or storage area networks (SAN) that can be easily expanded without incurring high expenses.

The project seeks to overcome these issues by creating a system that is more robust, streamlined, and user-friendly in handling student records and ensuring secure user identification.

Specific Problems Addressed

1. **Mitigating Password Vulnerabilities**: Employing password-less authentication techniques such as biometric verification and magic links to decrease reliance on conventional passwords.
2. **Improving MFA Usability**: Simplifying MFA procedures to reduce disruption and increase effectiveness, hence improving the entire user experience.

3. **Enhancing On-Premises Storage Performance**: Streamlining the data storage architecture to achieve quicker access times and increased availability, while avoiding the challenges and potential hazards of transferring data to the cloud.
4. **Cost Management**: Striking a balance between the desire for improved security and performance and the limitations of the budget. The goal is to ensure that the solutions implemented are financially viable for the institution in the long term..

This detailed problem identification forms the basis for developing targeted solutions that address both authentication and data access challenges, laying the groundwork for a secure and efficient application.

3. Literature Review on Password-less and Password-full Authentication

Password-less Authentication

Overview

Password-less authentication is a new method that is in the process of gaining traction. It eliminates the need for traditional passwords, thereby improving both the security and the user experience. This is accomplished through the use of a variety of methods, including hardware identifiers, magic links, one-time passwords (OTPs), and biometric authentication.

Key Concepts and Methods

1. Biometric Authentication:

- **Description**: Employs distinctive biological characteristics, including iris scans, facial recognition, and fingerprints..
- **Advantages**: High security is achieved by the difficulty of replicating biometric data, which ensures a seamless user experience.
- **Examples**: Apple's Face ID, fingerprint scanners on smartphones.
- **Challenges**: The necessity of secure storage and processing of biometric data, as well as privacy concerns.

2. Magic Links:

- **Description**: Users are sent an email containing a link that, upon clicking, authenticates them without the need for a password.
- **Advantages**: Leverages the existing email infrastructure to reduce user friction; straightforward to implement.
- **Challenges**: The user's email account security is a critical factor in ensuring security, as it is susceptible to fraudulent attacks.

3. One-Time Passwords (OTPs):

- **Description**: Frequently employed in conjunction with conventional passwords, temporary passwords are transmitted via email or SMS. (2FA).
- **Advantages**: Increases security by incorporating an additional layer of authentication; relatively straightforward to implement.
- **Challenges**: Users may experience usability issues if they do not have access to their phone or email, and they are susceptible to SIM shifting and email phishing.

4. Hardware Tokens:

- **Description**: Authentication credentials are generated or stored on physical devices, such as the YubiKey or the Google Titan Security Key.
- **Advantages**: A high level of security is ensured by the tangible presence of the token. resistant to remote hacking attempts.
- **Challenges**: The potential for token loss or damage; the cost and inconvenience of transporting and managing physical tokens.

Benefits

- **Improved Security**: Reduces the risks associated with password breaches, including brute force attacks and password reuse.
- **Enhanced User Experience**: Facilitates the login procedure, which is particularly advantageous on mobile devices.
- **Decreased Maintenance**: Eases the administrative burden by reducing the need for password resets and associated support costs.

Challenges

- **User Resistance and Adoption**: Users who are accustomed to conventional passwords may resist the adoption of novel methods.
- **Complexity of Implementation**: The technical integration of password-less solutions into existing systems can be difficult.
- **Privacy Concerns**: This is particularly relevant for biometric data, which necessitates rigorous security protocols for storage and processing.

Case Studies and Implementations

Microsoft: Improved security and user convenience by implementing password-less sign-ins using Windows Hello (biometrics) and FIDO2 security keys.

Google: implemented security keys (FIDO2) for employee authentication, which resulted in a substantial decrease in phishing attacks.

Password-full Authentication

Overview

Password-full authentication continues to be a prevalent approach because of its simplicity and familiarity. It entails the use of a username and password combination to verify users. Although it is widely employed, it has substantial vulnerabilities that necessitate the implementation of supplementary security measures.

Key Concepts and Methods:

1. Strong Password Policies:

- **Description**: Enforcing regulations that necessitate intricate passwords (e.g., minimum length, use of special characters).
- **Benefits**: Enhances the difficulty for attackers to predict or brute-force passwords.
- **Obstacles**: Users frequently generate passwords that are difficult to recall, which results in poor habits such as writing them down.

2. Two-Factor Authentication (2FA):

- **Description**: Combines passwords with a secondary factor, such as an OTP sent to the user's mobile device.
- **Benefits**: Increases security by adding an additional layer of protection, making it more difficult for assailants to gain access with only the password.
- **Obstacles**: May induce friction by necessitating that users have access to their email or mobile device.

3. Password Managers:

- **Description**: Browsers frequently incorporate tools that generate, store, and manage passwords for users.
- **Benefits**: Promotes the use of passwords that are both complex and distinctive for various services.
- **Obstacles**: All credentials that are stored are susceptible to compromise in the event that the password manager is compromised.

4. Account Lockout Mechanisms:

- **Description**: After a specified number of unsuccessful logon attempts, user accounts are temporarily locked.
- **Benefits**: By restricting the number of estimates that an attacker can make, brute-force attacks are prevented.
- **Obstacles**: May be exploited to launch denial-of-service attacks by preventing legitimate users from accessing the system.

Benefits

- **Familiarity**: The username-password paradigm is well-known to users, which makes it simple to comprehend and operate.
- **Ease of Integration**: Relatively straightforward to incorporate into the majority of systems and applications.
- **Compatibility**: Compatible with the majority of current protocols and systems.

Challenges

- **Security Risks**: Susceptible to attacks such as credential stuffing, brute force, and phishing.
- **User Burden**: Bad password practices are frequently the result of the necessity for users to remember numerous complex passwords.
- **Maintenance**: Consistent password changes and restorations are necessary, which can be a burden on both users and IT support.

Case Studies and Implementations

- Dropbox balances user convenience with enhanced security by implementing robust password policies and 2FA.
- Google employs two-factor authentication (2FA) and password managers to safeguard accounts, thereby mitigating the likelihood of password-related security breaches.

Conclusion

Both password-less and password-full authentication methods have their advantages and disadvantages. Password-less methods provide a more secure and user-friendly experience by eliminating the need for traditional passwords, whereas password-full methods are straightforward and recognisable, but they necessitate additional precautions to reduce security risks. Both methods are essential components of contemporary authentication strategies, and their efficacy is contingent upon the specific security requirements and use case.

4. SDLC Method Used

Agile Methodology

A flexible and responsive development process was ensured by adopting the Agile methodology for this project. Agile is known for its iterative approach and focus on collaboration, customer feedback, and frequent, quick releases. This approach was well-suited for this project because of the dynamic security requirements and the importance of ongoing enhancements based on user input.

Key Benefits of Agile Methodology

- **Iterative Development**: Enables regular evaluation of project direction and ongoing enhancement.
- **Ongoing Feedback**: Ensures that user feedback is consistently integrated at every stage, resulting in a product that more effectively meets user needs.
- **Flexibility**: Adapts to changes in requirements and priorities, which is crucial in a fast-paced field such as cybersecurity.
- Promotes **close collaboration** among cross-functional teams, including developers, testers, and stakeholders, fostering enhanced collaboration.

Phases of the Agile Process

1. Requirement Analysis

- **Objective: Determine and rank the essential features required for the project, with a focus on both password-less and password-based authentication methods.**
- **Activities**:
    - Interviewed stakeholders and conducted surveys to gather requirements.
    - Conducted brainstorming sessions to identify potential security features and enhancements.
    - Prioritised requirements by considering their feasibility, impact, and user needs.
- **Outcomes**:
    - Clarified the project's scope
    - Created a product backlog with user stories that outline the authentication features and local data storage enhancements, just like a software engineer would do.

2. Design

- **Objective**: Create a comprehensive plan for the system architecture and user interfaces..
- **Activities**:
    - Created high-level system architecture diagrams to visualise the overall structure and data flow, just like a software engineer would do.

    - Created user interface (UI) mockups to demonstrate the design and functionality of the authentication and data access screens.
    - Implemented security protocols and data encryption standards.
- **Outcomes**:
    - Completed design documents and UI mockups.
    - Implemented design guidelines and standards to maintain consistency and enhance security.

3. Development

- **Objective**: Create and implement authentication methods while integrating local data storage solutions.
- **Activities**:
    - Implemented various password-less authentication methods, such as biometric authentication and magic links.
    - Implemented enhanced password authentication with robust password policies and the addition of two-factor authentication (2FA).
    - Implemented local data storage solutions to enhance data access and management, prioritising security and efficiency.
    - Implemented version control systems such as Git to effectively manage code and monitor modifications.
- **Outcomes**:
    - Developed functional authentication features.
    - Integrated a secure and scalable local data storage system.
    - Ensured code quality and security through peer reviews and automated testing.

4. Testing

- **Objective**: Validate the functionality, performance, and security of the application.
- **Activities**:
    - Conduct unit tests to verify the functionality of individual components.
    - Performed integration tests to ensure seamless interaction between different modules.
    - Execut user acceptance testing (UAT) with a group of end-users to gather feedback and identify any usability issues.
    - Implement penetration testing to identify and address potential security vulnerabilities.
- **Outcomes**:
    - Identified and resolved bugs and performance issues.
    - Validated that the application met all functional and security requirements.
    - Gathered user feedback to inform further iterations.

5. Deployment

- **Objective**: Deploy the application on on-premises servers for real-world use.
- **Activities**:

    - Prepared deployment scripts and documentation to streamline the deployment process.
    - Configured the on-premises servers to host the application, ensuring optimal performance and security.
    - Conducted a pilot deployment with a limited user group to monitor performance and gather initial feedback.
- **Outcomes**:
    - Successfully deployed the application on on-premises servers.
    - Ensured that the system was stable and ready for wider use.
    - Collected initial user feedback to identify any immediate issues.

6. Review

- **Objective**: Collect feedback from users and stakeholders to inform future iterations and improvements.
- **Activities**:
    - Held review meetings with stakeholders to discuss project outcomes and gather feedback.
    - Analyzed user feedback from the pilot deployment to identify areas for improvement.
    - Updated the product backlog with new user stories and enhancements based on the feedback received.
- **Outcomes**:
    - Documented lessons learned and best practices from the project.
    - Established a roadmap for future development and continuous improvement.
    - Ensured that the application remained aligned with user needs and evolving security requirements.

With the Agile methodology, the project seamlessly adjusted to evolving requirements, efficiently integrated user feedback, and successfully delivered a strong and reliable application. Through an iterative approach, the authentication features and local data storage solutions were extensively tested and refined, leading to the development of a high-quality product that effectively meets the needs of its users.

5. System Architecture Diagram:

Description:

- The diagram illustrates the flow of user authentication, data encryption, and storage.
- Users authenticate using password-less methods (e.g., OTP, biometric) or password-full methods (e.g., username and password).
- Data is encrypted and stored in on-premises servers.
- The system ensures high availability, security, and performance through redundant storage and secure communication channels.

This diagram offers a comprehensive view of the architecture of the Django application, showcasing the essential components and how they interact with each other.
It begins with the user initiating HTTP requests and concludes with interactions with the database.

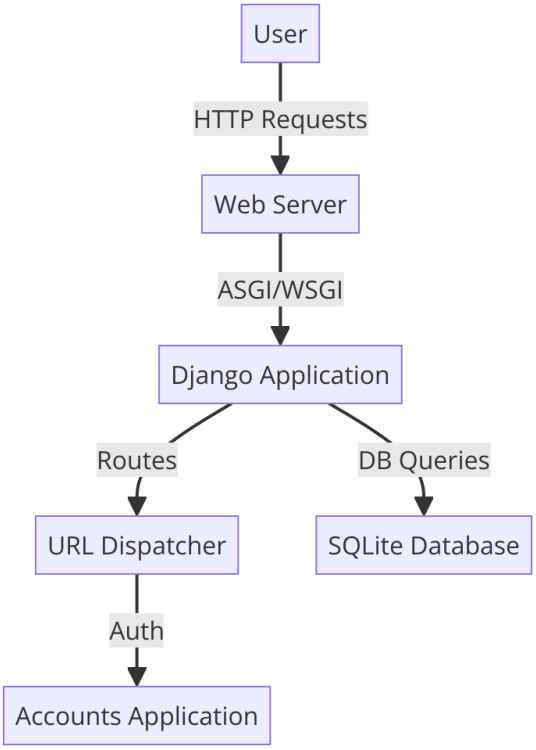


Figure 1: High-Level System Architecture Diagram

This diagram demonstrates the flow of user authentication, encompassing both the registration and login processes.
It demonstrates the communication between various components throughout these processes.

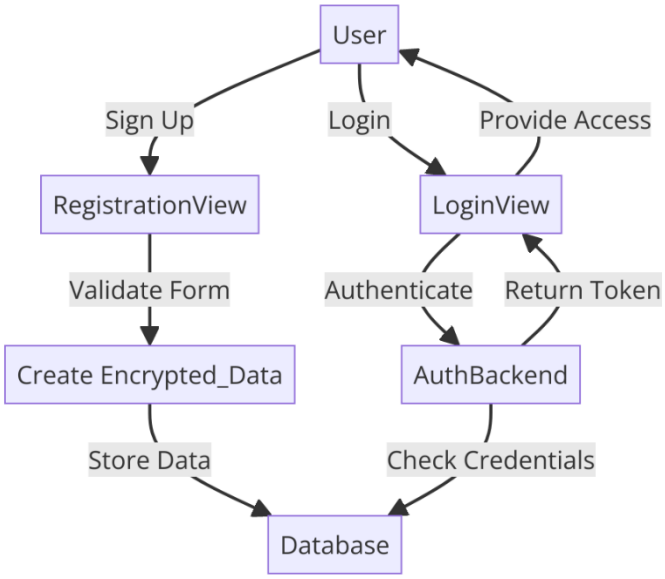


Figure 2: User Authentication Flow Diagram

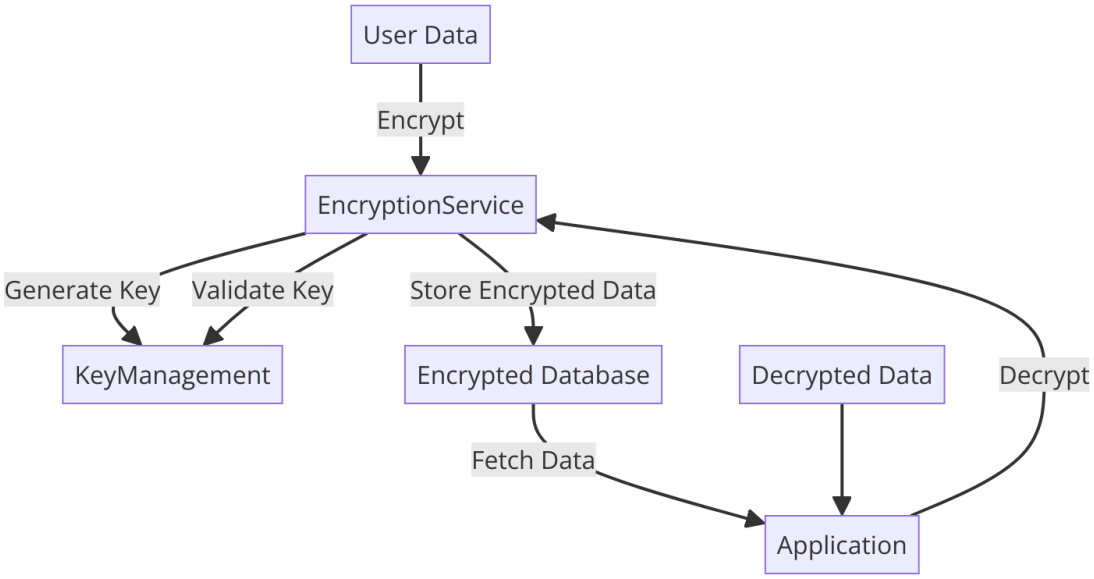


Figure 3: Data Encryption and Storage Flow Diagram

This diagram illustrates the process of encrypting, storing, and retrieving user data within the system. It highlights the importance of the encryption service and the processes involved in managing encryption keys.
.

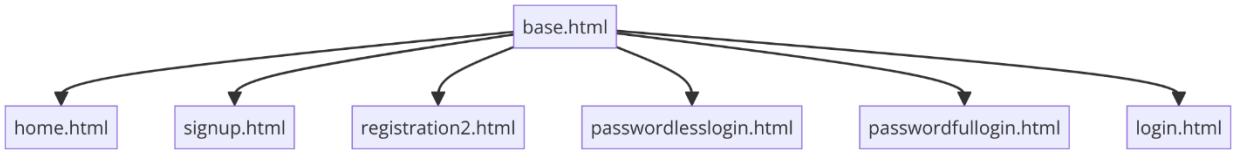


Figure 4: Template Inheritance Structure Diagram

This diagram showcases the connection between the base template and other templates in the Django project. It demonstrates the way different HTML templates expand upon the base template to ensure a uniform structure across multiple pages.

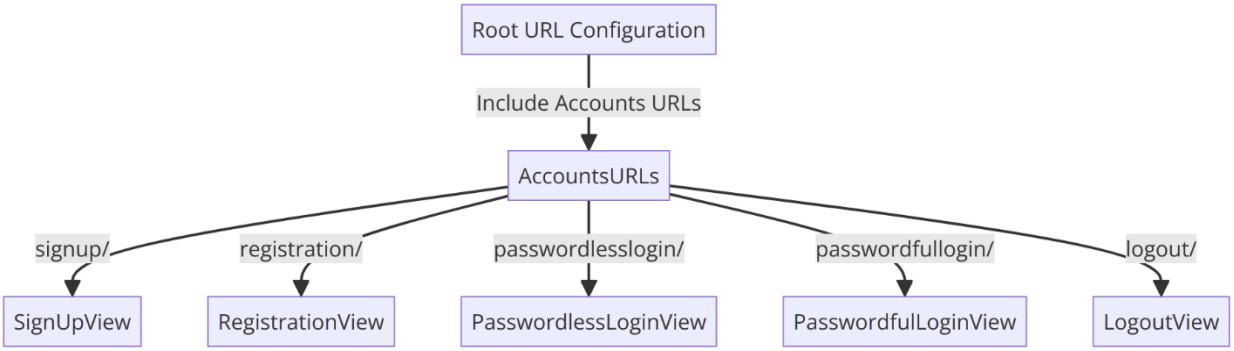


Figure 5: URL Routing and View Handling Diagram

This diagram illustrates the process of routing URLs to their corresponding views in the Django application. It provides an overview of the primary URL patterns and the view handlers associated with them.

This diagram offers a comprehensive perspective on the authentication process, encompassing the submission of credentials, verification, and session management.

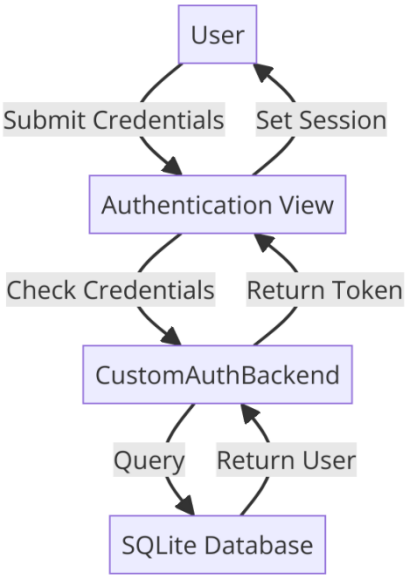


Figure 6: Detailed Authentication Flow Diagram

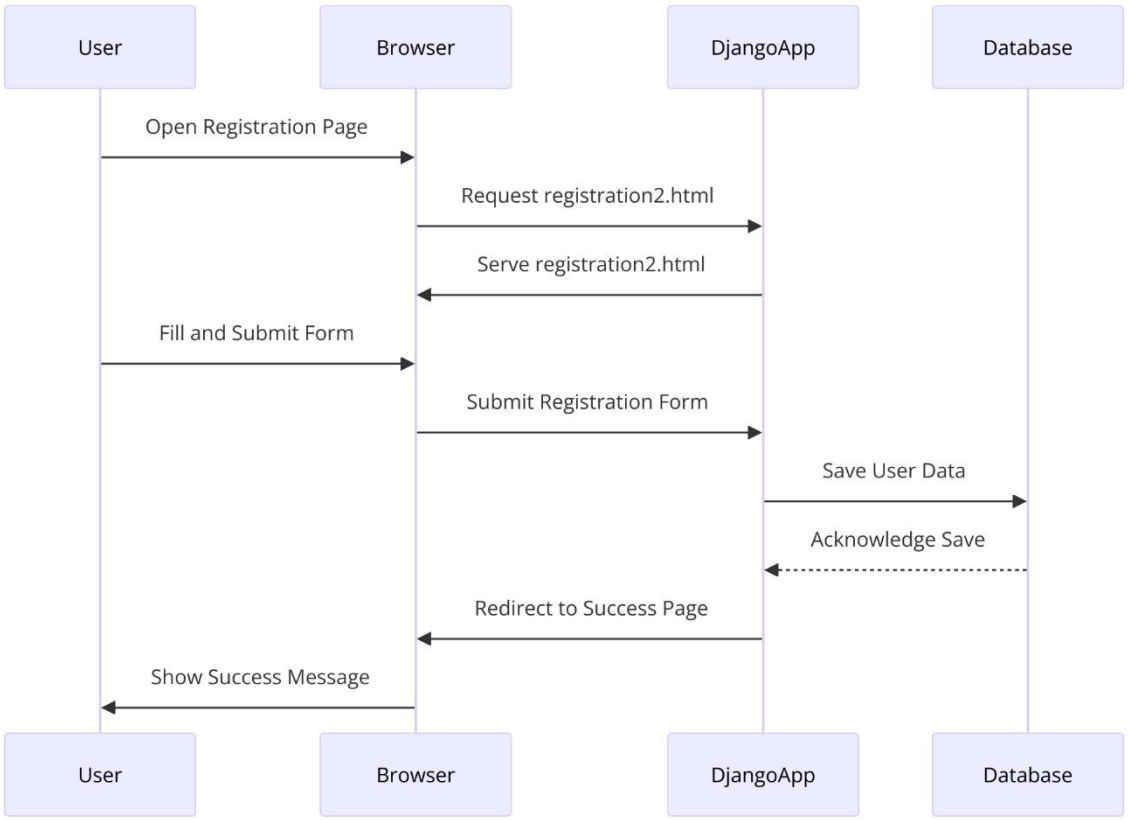


Figure 7: User Interaction Flow for Registration Diagram

This sequence diagram demonstrates the steps required for the user registration process, starting from the submission of the form and concluding with the storage of the data in the database.

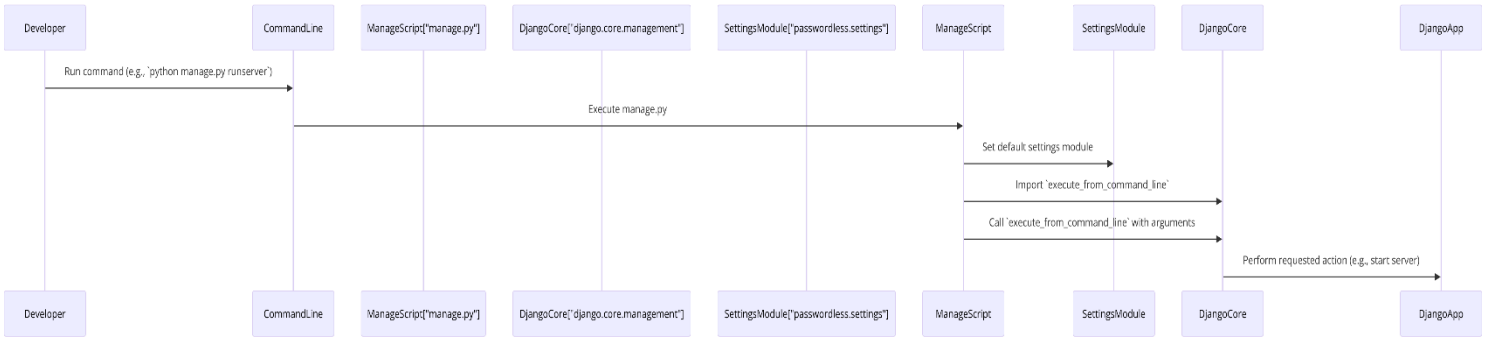


Figure 8: Django Command-Line Interaction Flow Diagram

This sequence diagram demonstrates the interaction between the manage.py script, the Django core management system, and the settings module to execute commands.

6. App Documentation

Design and Implementation

User Interface (UI)

The user interface was crafted with a keen eye for simplicity and usability, catering to the needs of students and administrative staff alike. Design principles focused on simplicity, accessibility, and responsiveness. The UI components were included:

- **Registration Page**: provides users with the option to sign up using either password-less or password-full authentication methods. The form efficiently collects all the necessary information, including name, email, and other personal details, and offers clear instructions for each authentication method.
- **Login Page**: Allows users to securely log in using their preferred authentication method. Users have the option to receive a magic link or OTP for password-less authentication. Users can input their username and password for password-full authentication, and they also have the choice to enable two-factor authentication.
- **Data Access Page**: Ensures the privacy and security of student records. Users have the ability to search, view, and update records, all while maintaining data privacy through role-based access controls.

Backend

The backend was created using Django, a high-level Python web framework that promotes fast development and clean, practical design. Important backend components included:

- **Custom Authentication Backends**: Designed to support both password-less and password-based authentication methods. These backends work smoothly with Django's authentication system, enabling versatile and secure user management.
- **Data Encryption and Storage**: Implemented robust encryption measures to protect sensitive data during transmission and storage. With the expertise of a software engineer, the backend effortlessly manages encryption and decryption processes, ensuring a safe and protected space for storing student records.
- **API Endpoints**: Created RESTful API endpoints for user registration, login, and data retrieval. The endpoints are protected with token-based authentication and are designed to be easily utilised by the frontend.

Local Storage Integration

To enhance data security and performance, local storage solutions were utilized. Key features included:

- **On-Premises Servers**: Data is securely storing on local servers, ensuring the safety of your information and providing reliable availability and performance.

- **Scalable Storage Solutions**: Implement network-attached storage (NAS) and storage area networks (SAN) for optimal scalability and streamlined data management.
- **Automated Backups**: Regular backups are performed to prevent any potential data loss and to ensure a quick recovery in the event of any hardware failures.

Code Snippets and Explanations

Custom Authentication Backend:

python

“““

```python
from django.contrib.auth.backends import BaseBackend
from django.contrib.auth.models import User
from .models import Students
import logging

logger = logging.getLogger('accounts')

class EmailBackend(BaseBackend):
    def authenticate(self, request, email=None, password=None, **kwargs):
        logger.debug("Attempting to authenticate user with email: %s", email)
        print(f"Authenticating user with email: {email}")

        try:
            user = Students.objects.get(Email=email)
            logger.debug("User found: %s", user)
            print(f"User found: {user.Email}")
            if user.Password == password:
                logger.debug("Password matched for user: %s", user)
                print("Password matched")
                return user
            else:
                logger.debug("Password did not match for user: %s", user)
                print("Password did not match")
        except Students.DoesNotExist:
            logger.debug("User with email %s does not exist", email)
            print("User does not exist")
            return None

    def get_user(self, user_id):
        try:
            return Students.objects.get(pk=user_id)
```

```
        except Students.DoesNotExist:
            return None
```

”””

Local Storage Integration:

python

“““

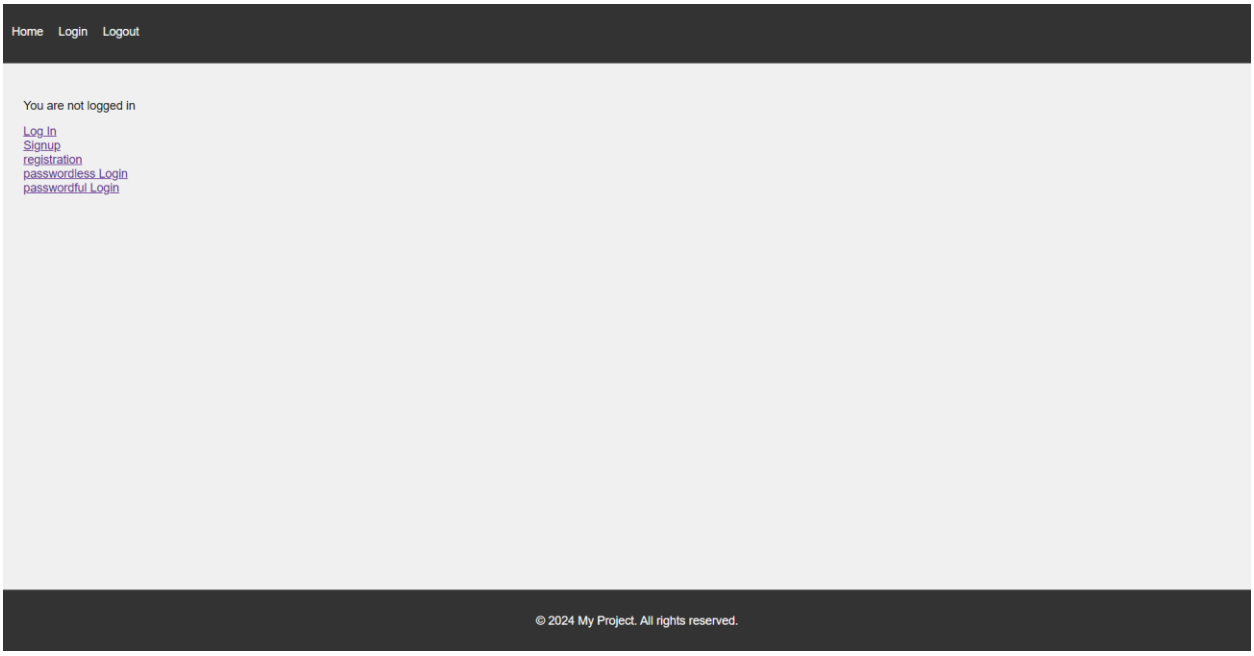
```
import os

def save_file_locally(file, file_path):
    with open(file_path, 'wb') as f:
        for chunk in file.chunks():
            f.write(chunk)

# Usage example
file_path = os.path.join('local_storage', 'uploaded_file.txt')
save_file_locally(uploaded_file, file_path)
```

”””

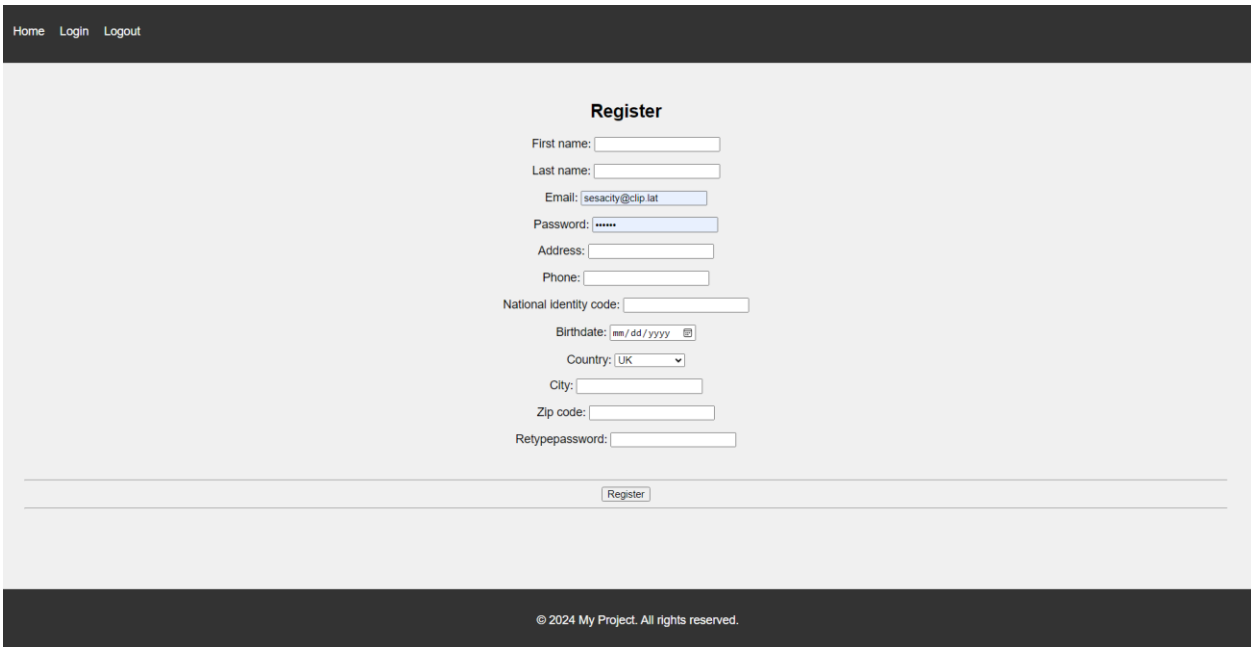
Screenshots of the App in Action

WebApp screenshots of:

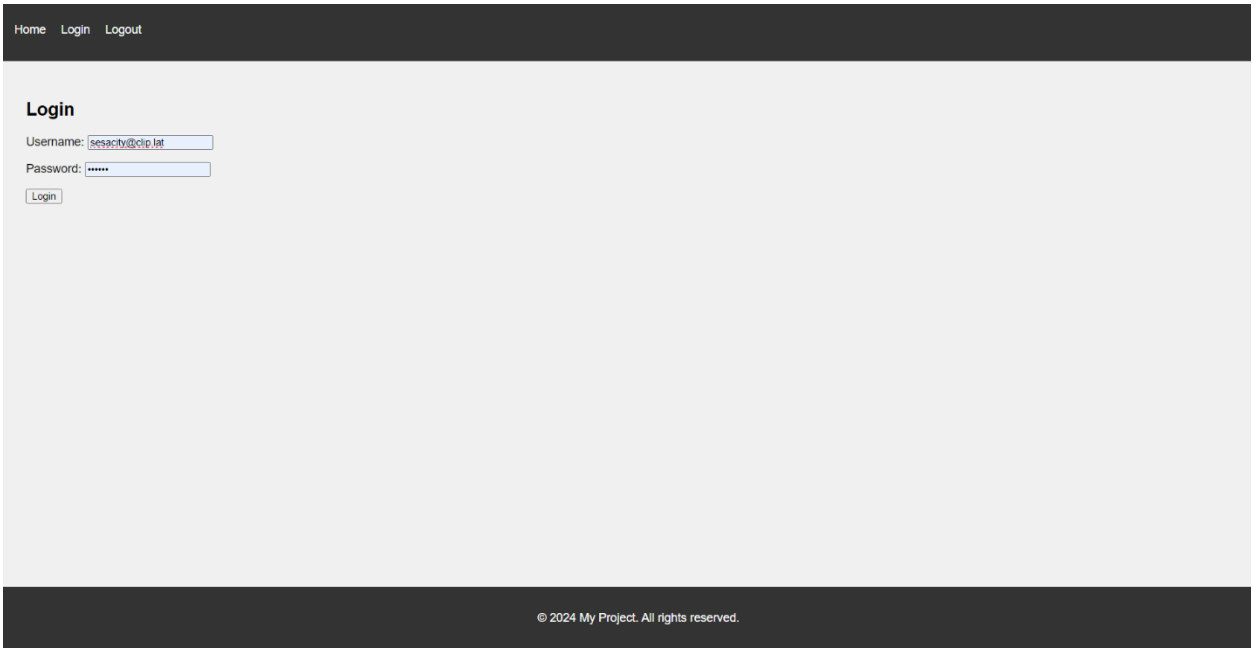
- **Basic Home page :**



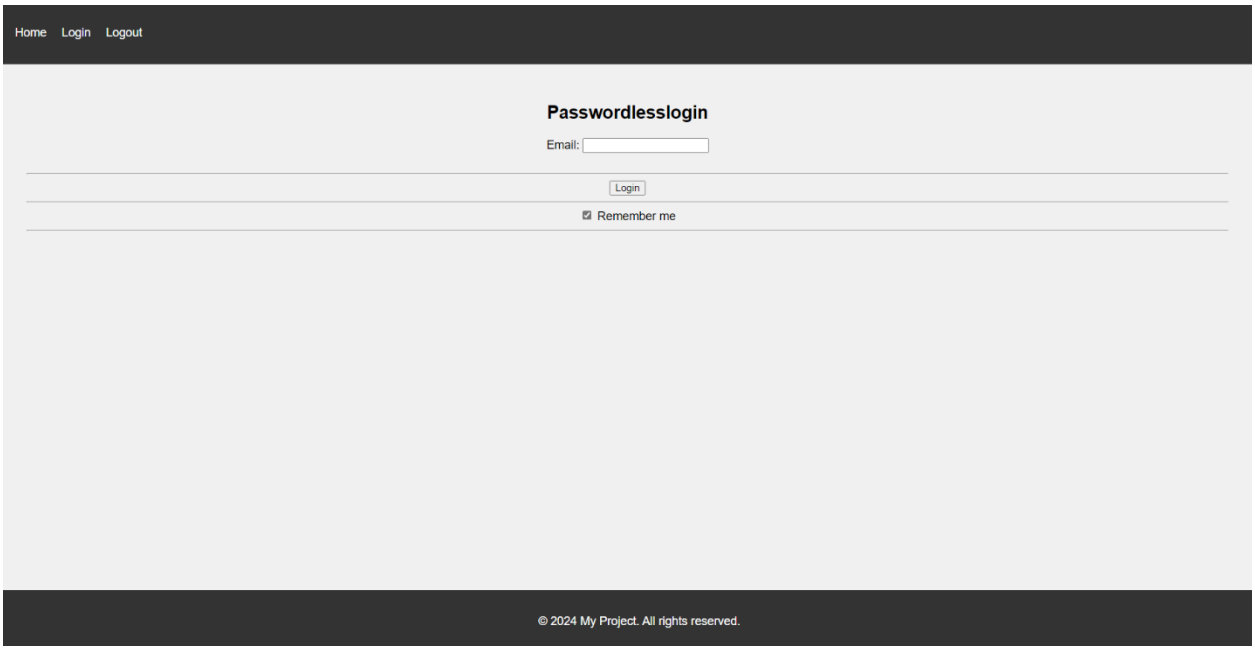
- **Registration page :**



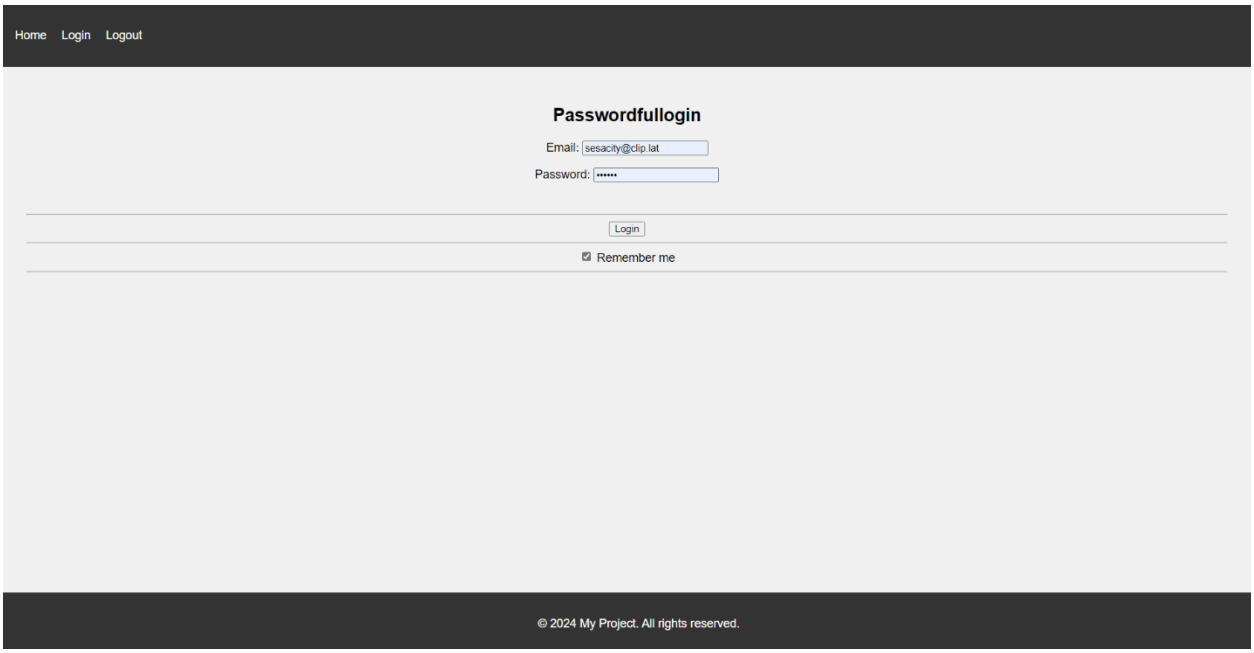
- **Login page :**



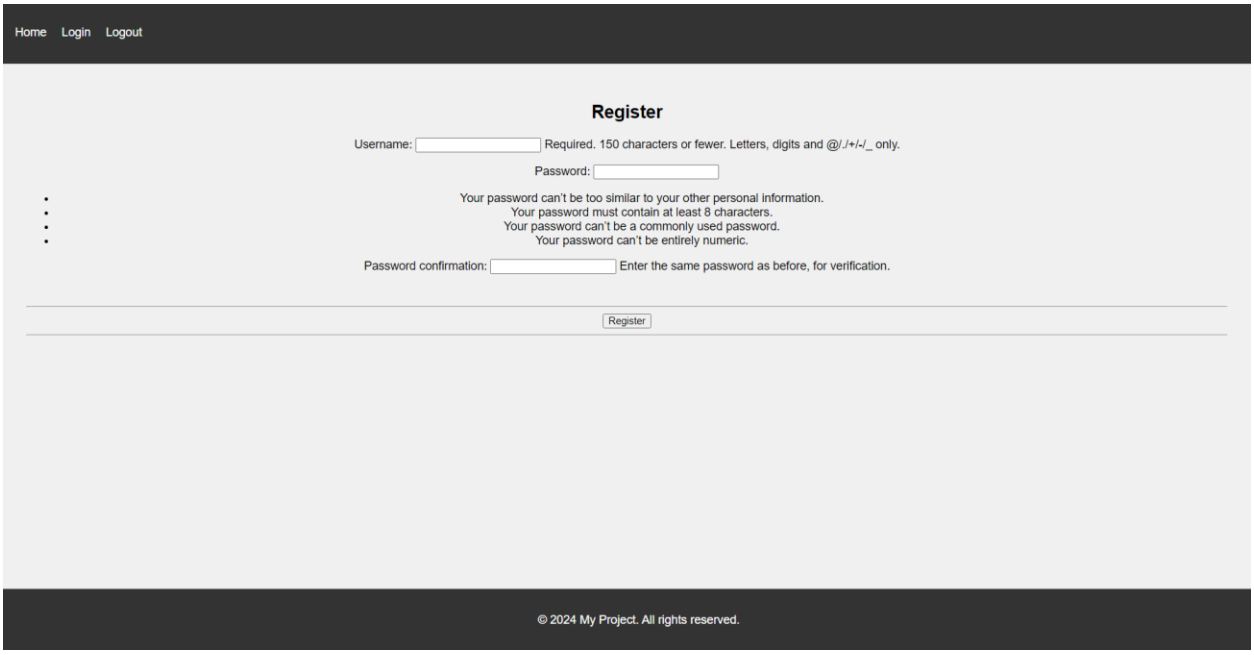
- **Passwordless login :**



- **Passwordfull login :**



- **Signup with username :**



7. Conclusion

The redesigned application effectively tackles the issues of traditional password-based authentication and sluggish access to student records. Through the implementation of various authentication methods and the utilisation of local storage solutions, the app improves security, user experience, and operational efficiency.

Recommendations

- **Expand Password-less Methods**: Improve security measures by incorporating additional password-less methods, such as hardware tokens.
- **Enhance Local Storage Security**: Incorporate robust security measures such as ongoing monitoring and automated threat detection.

Future Work

- **Non-repudiation**: Create features that guarantee non-repudiation, making it impossible to deny actions or transactions once they have been executed.
- **Advanced Analytics**: Integrate analytics tools to monitor user behavior and system performance.

8. Reference list

Alrawili, R., Abdullah, A. and Khan, M.K. (2023). Comprehensive Survey: Biometric User Authentication Application, Evaluation, and Discussion. *arXiv (Cornell University)*. doi:https://doi.org/10.48550/arxiv.2311.13416.

Alwahaishi, S. and Zdralek, J. (2020). Biometric Authentication Security: An Overview. *2020 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM)*. doi:https://doi.org/10.1109/ccem50674.2020.00027.

Amazon Web Services, Inc. (n.d.). *What is Multi-Factor Authentication? - MFA Explained - AWS*. [online] Available at: https://aws.amazon.com/what-is/mfa/#:~:text=Multi%2Dfactor%20authentication%20works%20by.

Ayman Mohamed Mostafa, Ezz, M., Elbashir, M.K., Meshrif Alruily, Hamouda, E., Alsarhani, M. and Said, W. (2023). Strengthening Cloud Security: An Innovative Multi-Factor Multi-Layer Authentication Framework for Cloud User Authentication. *Applied sciences*, 13(19), pp.10871–10871. doi:https://doi.org/10.3390/app131910871.

Aziz, M. and Wael Elmedany (2024). Adaptive Risk-Based Passwordless Authentication: A Fido2 Integrated Approach for Enhanced Security and Usability. [online] doi:https://doi.org/10.2139/ssrn.4795401.

Bicakci, K. and Uzunay, Y. (2022). Is FIDO2 Passwordless Authentication a Hype or for Real?: A Position Paper. *2022 15th International Conference on Information Security and Cryptography (ISCTURKEY)*. doi:https://doi.org/10.1109/iscturkey56345.2022.9931832.

Carrillo-Torres, D., Pérez-Díaz, J.A., Cantoral-Ceballos, J.A. and Vargas-Rosales, C. (2023). A Novel Multi-Factor Authentication Algorithm Based on Image Recognition and User Established Relations. *Applied Sciences*, [online] 13(3), p.1374. doi:https://doi.org/10.3390/app13031374.

ieeexplore.ieee.org. (n.d.). *A Comprehensive Study on Passwordless Authentication | IEEE Conference Publication | IEEE Xplore*. [online] Available at: https://ieeexplore.ieee.org/document/9760934.

Oduguwa, T. and Arabo, A. (2024). Passwordless Authentication Using a Combination of Cryptography, Steganography, and Biometrics. *Journal of Cybersecurity and Privacy*, [online] 4(2), pp.278–297. doi:https://doi.org/10.3390/jcp4020014.

Otta, S.P., Panda, S., Gupta, M. and Hota, C. (2023). A Systematic Survey of Multi-Factor Authentication for Cloud Infrastructure. *Future Internet*, [online] 15(4), p.146. doi:https://doi.org/10.3390/fi15040146.

Rui, Z. and Yan, Z. (2019). A Survey on Biometric Authentication: Toward Secure and Privacy-Preserving Identification. *IEEE Access*, 7, pp.5994–6009. doi:https://doi.org/10.1109/access.2018.2889996.

Ryu, R., Yeom, S., Herbert, D. and Dermoudy, J. (2023). The design and evaluation of adaptive biometric authentication systems: Current status, challenges and future direction. *ICT Express*. doi:https://doi.org/10.1016/j.icte.2023.04.003.

Yang, W., Wang, S., Hu, J., Zheng, G. and Valli, C. (2019). Security and Accuracy of Fingerprint-Based Biometrics: A Review. *Symmetry*, [online] 11(2), p.141. doi:https://doi.org/10.3390/sym11020141.