MINDFUL

Course: Data Integrity and Authentication

Team Members:

- Yehia Ahmed Tawfiq - 2205126
- Maryam Waheed Zamel - 2205154
- Amina Ahmed Ferra - 2205225
- Mayssoune Hussein Elmasry - 2205251
- Hanin Mohamed Hamoda – 2205232

Project Overview

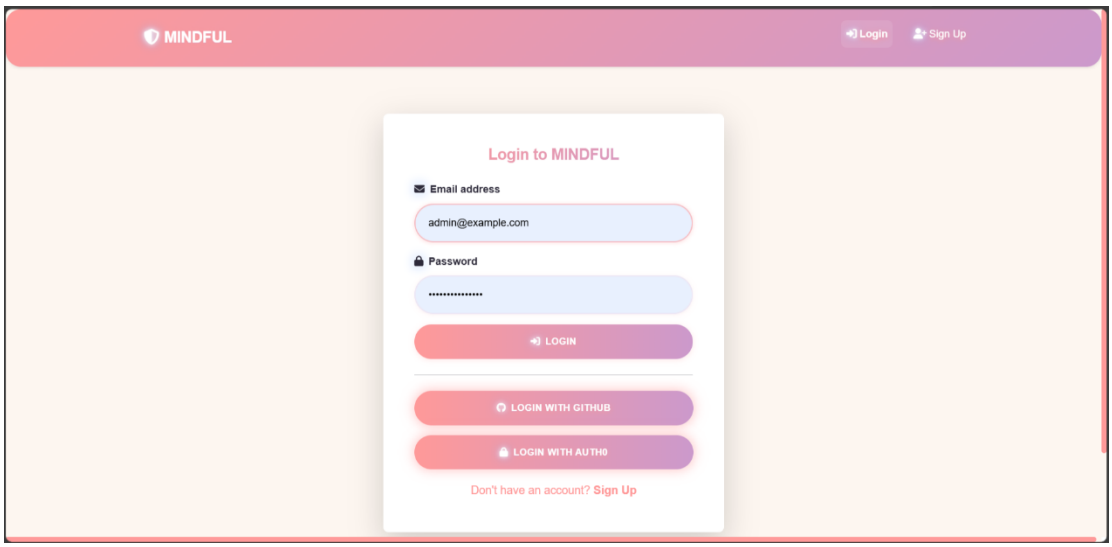
Mindful is a full-stack secure web application designed to manage sensitive health data and wellness records while ensuring data integrity, privacy, and access control. It combines modern authentication mechanisms, secure document handling, and interactive health tracking to simulate a real-world patient support system used in clinics, therapy centres, and corporate wellness programs.

Technologies Used

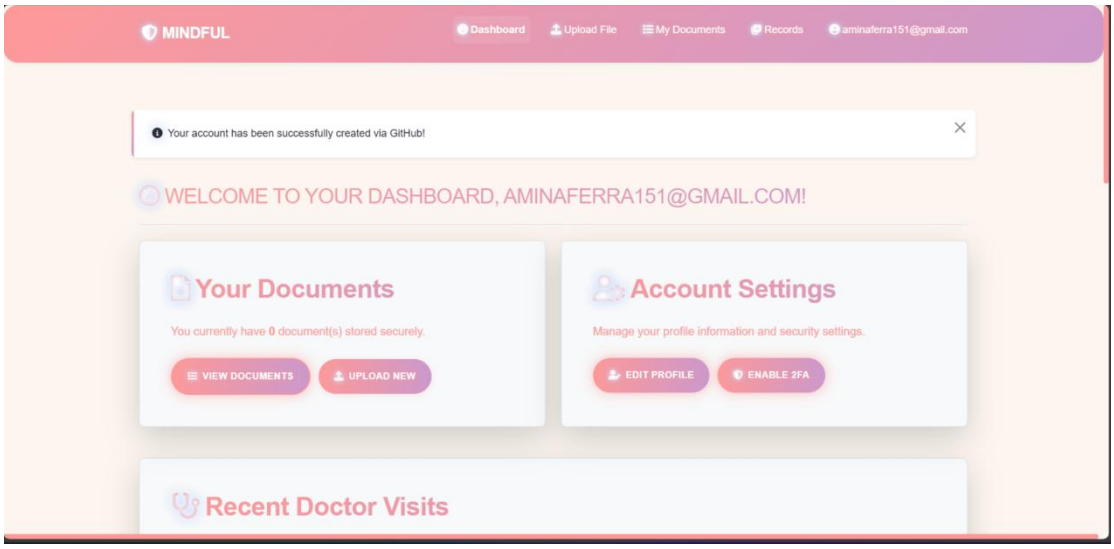
- ***Frontend:*** HTML, CSS, Bootstrap
- ***Backend:*** Python Flask
- ***Database:*** SQLite + SQLAlchemy
- ***Security:*** OpenSSL, SHA-256, AES, HMAC, HTTPS
- ***Authentication:*** OAuth, Okta, 2FA (Google Authenticator)

Now, we are going to Show screenshots of each implemented feature with clear explanation of the encryption and authentication flow & A Wireshark capture summary demonstrating secure communication.

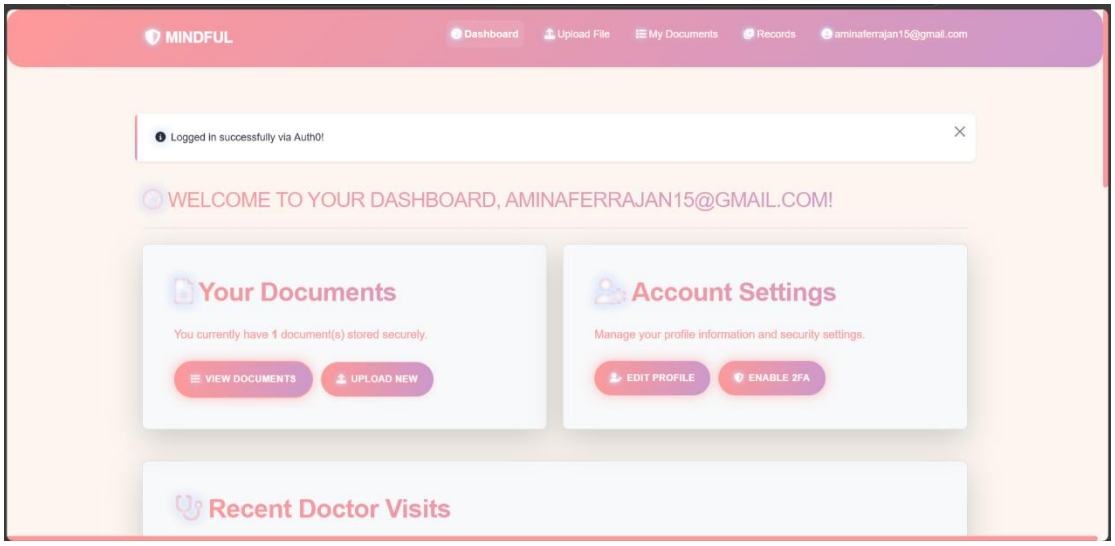
1-Login: user/admin can login and the system check if the Credentials are correct from the database & also can login via GitHub or Auth0 Not just manually.



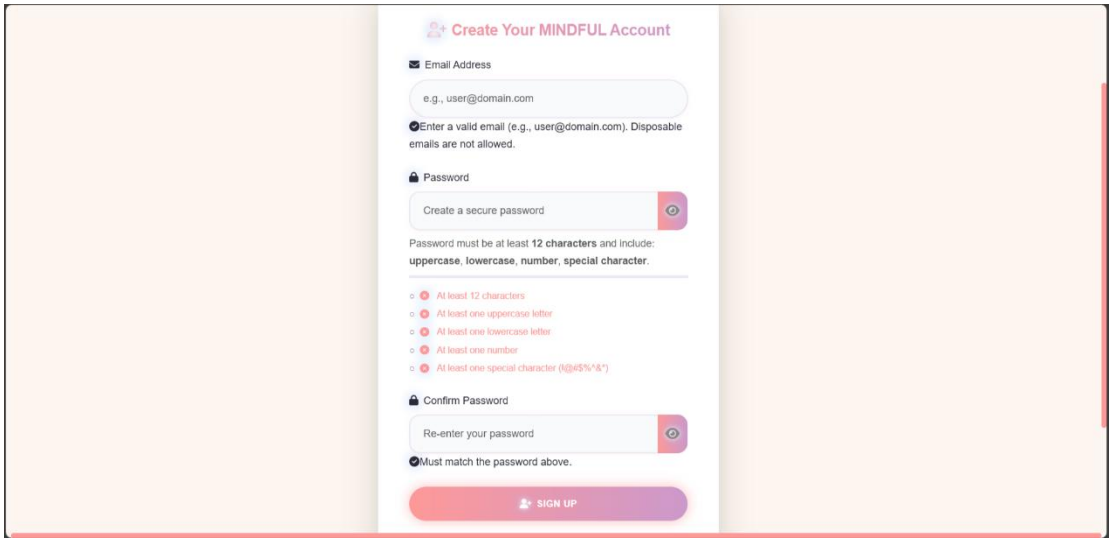
2- OAuth 2.0 Login via GitHub: Enables users to authenticate securely using popular external providers, reducing password management burden.



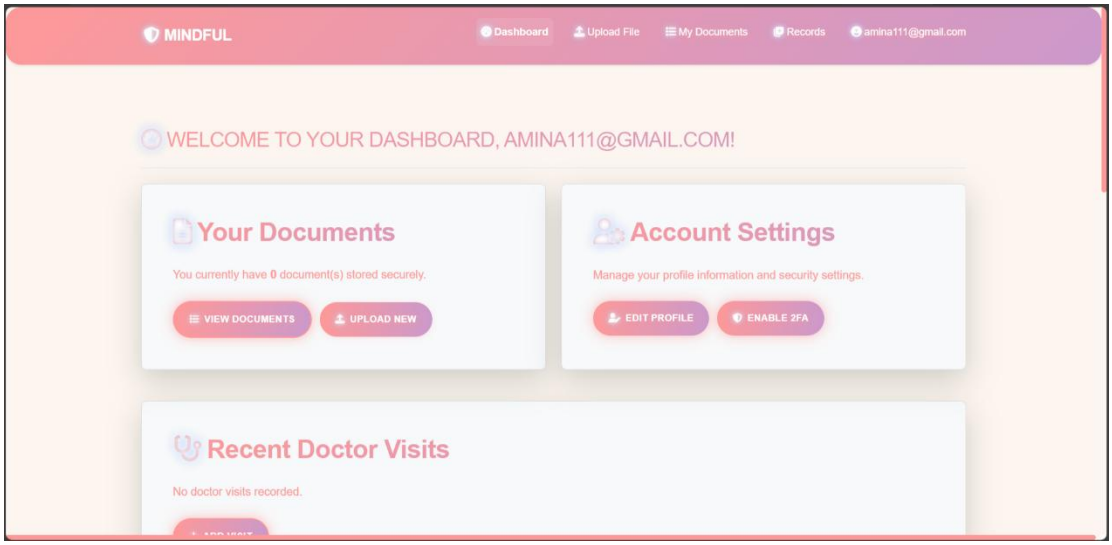
3- **Login via Okta**: Same as GitHub, enables users to authenticate securely



4- **Sign up**: If the user is new to the Website, he/she must Sign up first to create a new account and there are Policies in creating an account they must follow such as: Email format, Password Policies, Password must match to confirm it.

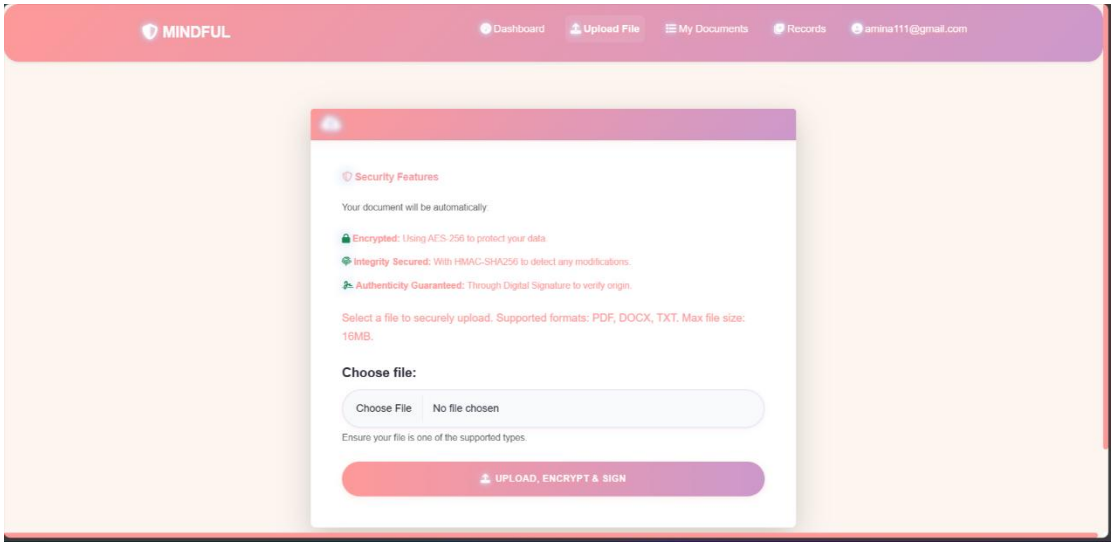


5- **User Dashboard**: when a new user is logged in, he/she enters this dashboard and can enable 2fa to enforce it each time he/she logs in.

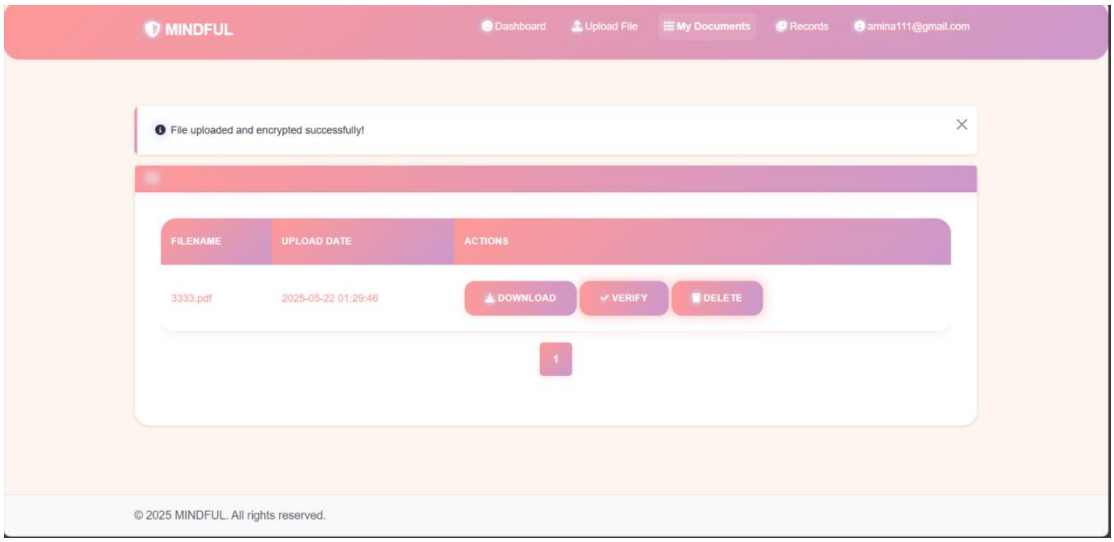


6- **Document Vault**:

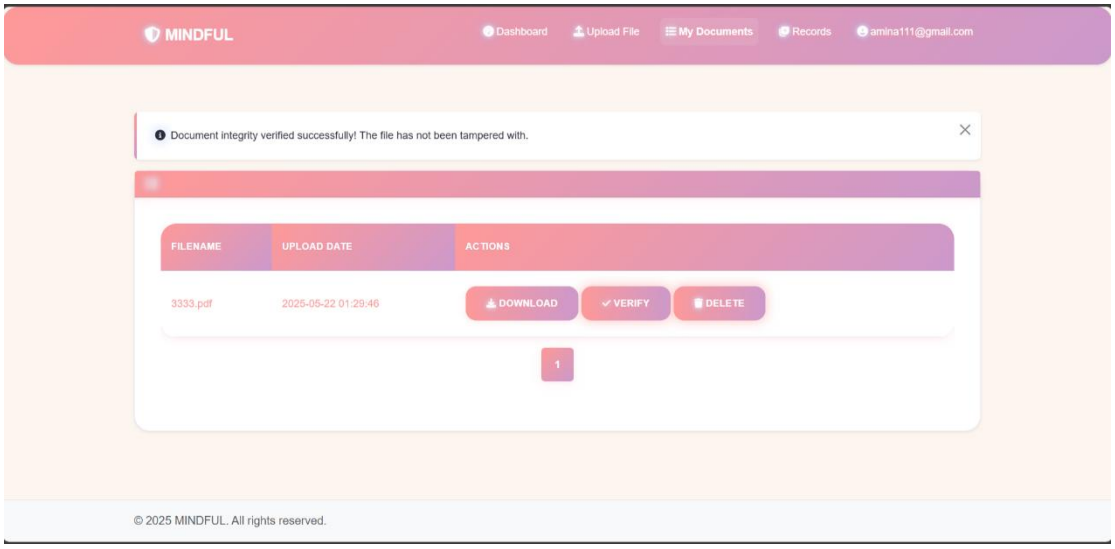
- **Document Upload:** Users can upload PDF, DOCX, and TXT files, which are processed securely.
- **AES Encryption:** All files are encrypted using the Advanced Encryption Standard before storage, ensuring confidentiality.
- **SHA-256 Hashing:** A cryptographic hash is computed for each document to uniquely identify and verify its integrity.
- **HMAC Verification:** the file's integrity is checked to ensure it hasn't been tampered with.
- **Digital Signature:** Uploaded documents are digitally signed, and this signature can be verified later to confirm authenticity.



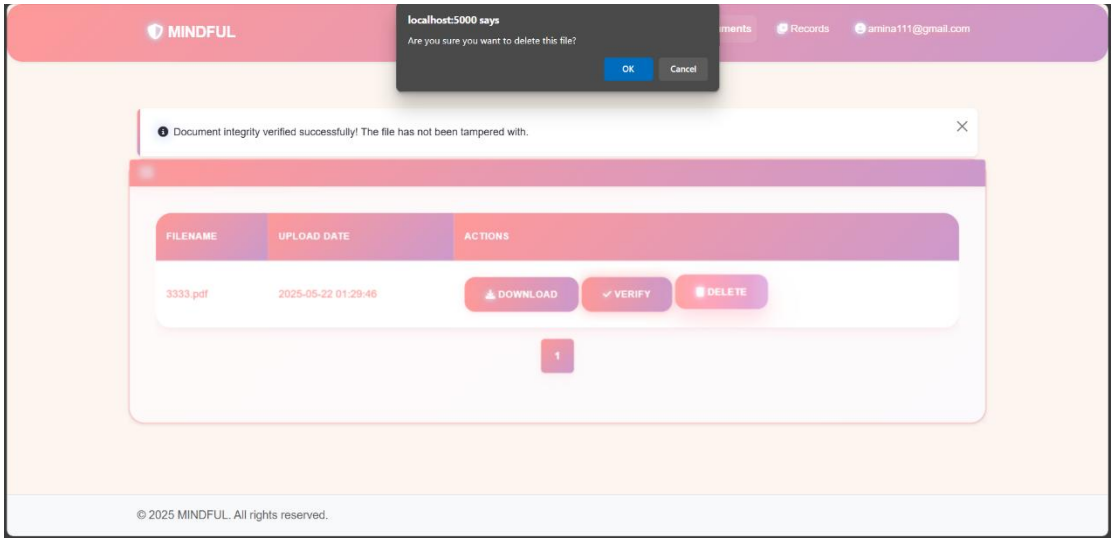
7- Cont. Documents: Users can Download their Documents.



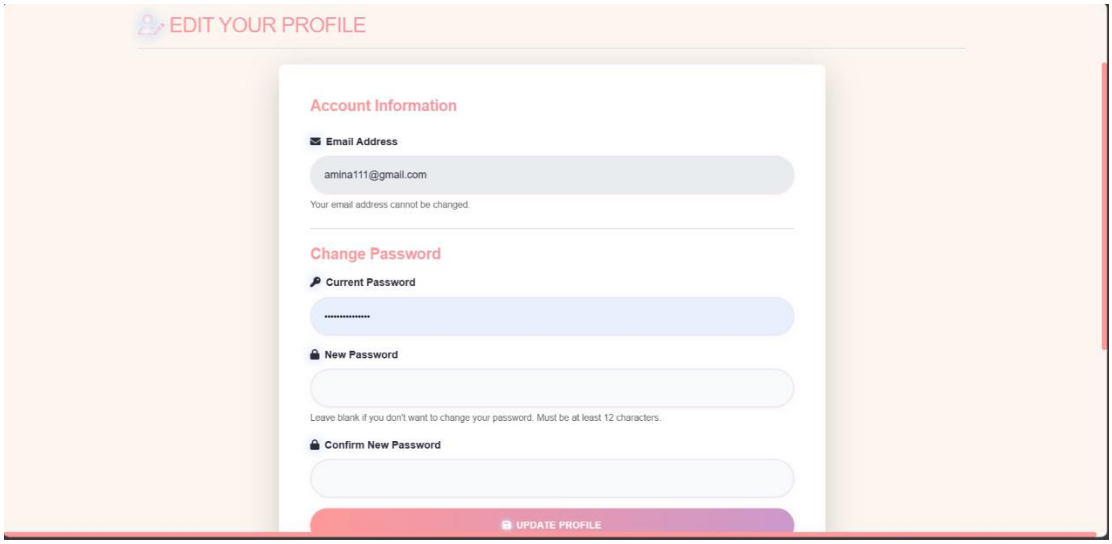
8- **Cont. Documents:** if the user clicked on verify integrity button, the System checks if the document has been tampered or not then shows a Confirmation message.



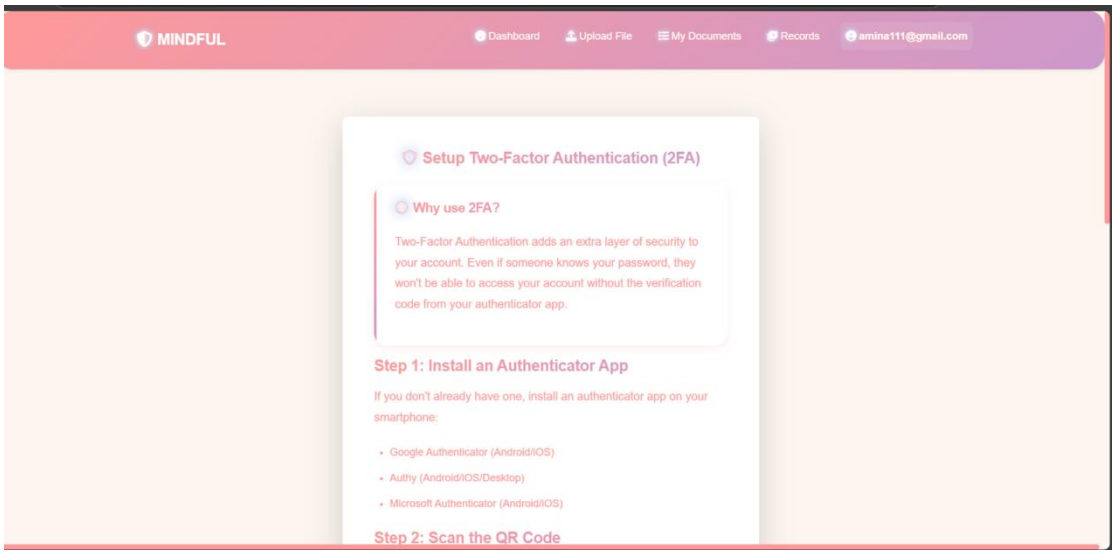
9- **Cont. Documents:** If the user wants to delete a document, the system makes sure that if the user wants to confirm deletion as it cannot be undone.

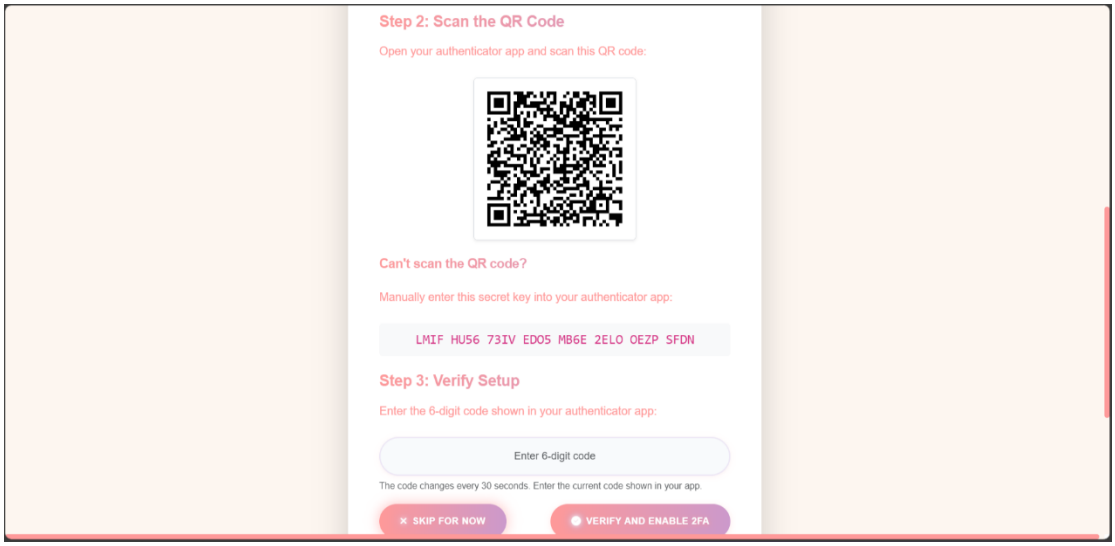


10-Edit Profile: User/Admin can edit their profiles easily as changing the passwords and also there are enforced password policies.

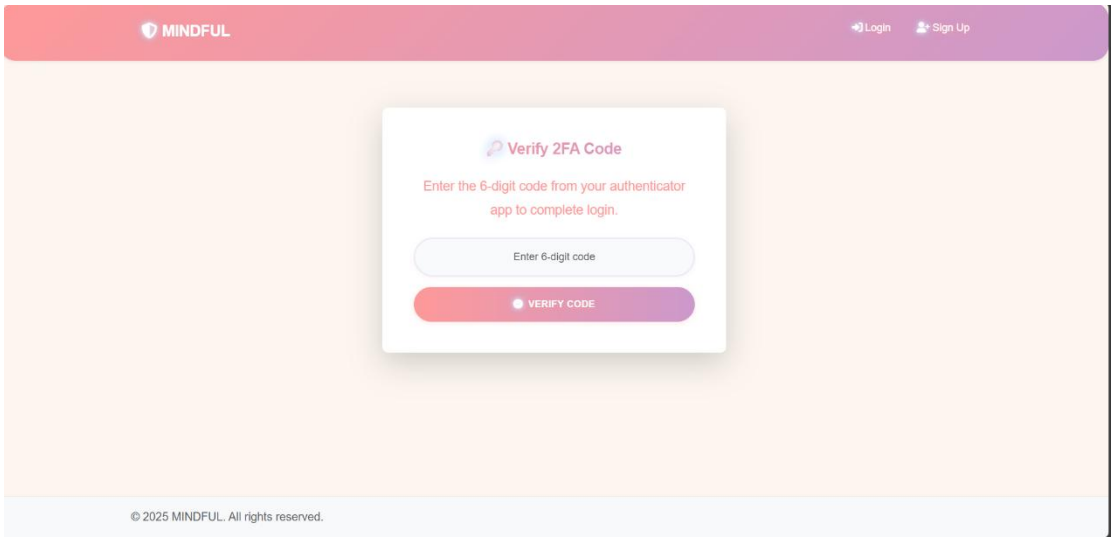


11-2FA: If the new user enables 2fa, he/she will scan the QR code via Google Authenticator then enter the valid code (make sure it is not expired) then click verify.

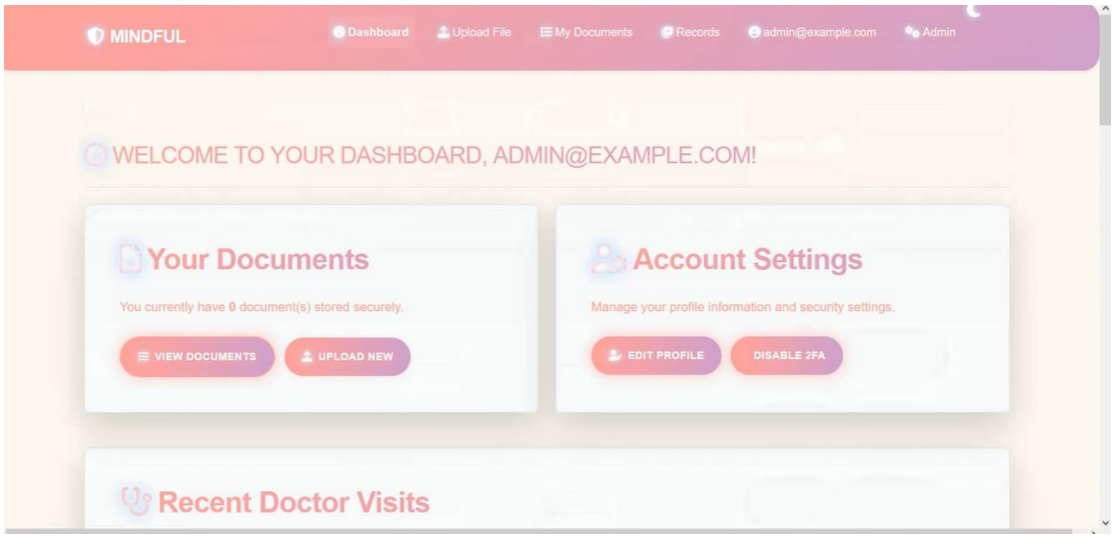




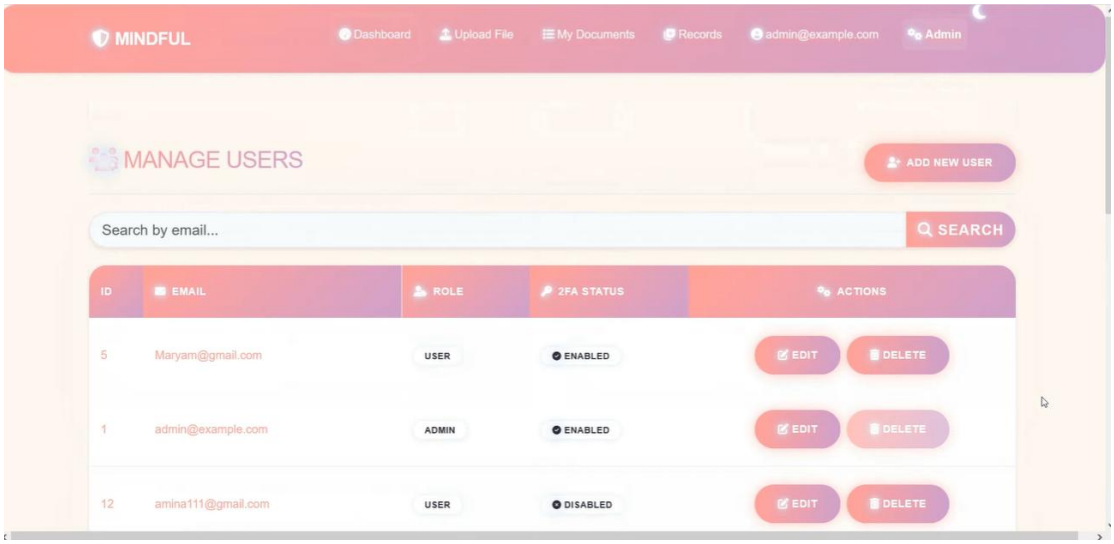
12- Cont. 2FA: After enabling, every time the user logs in, the system will ask him/her for a code to verify.



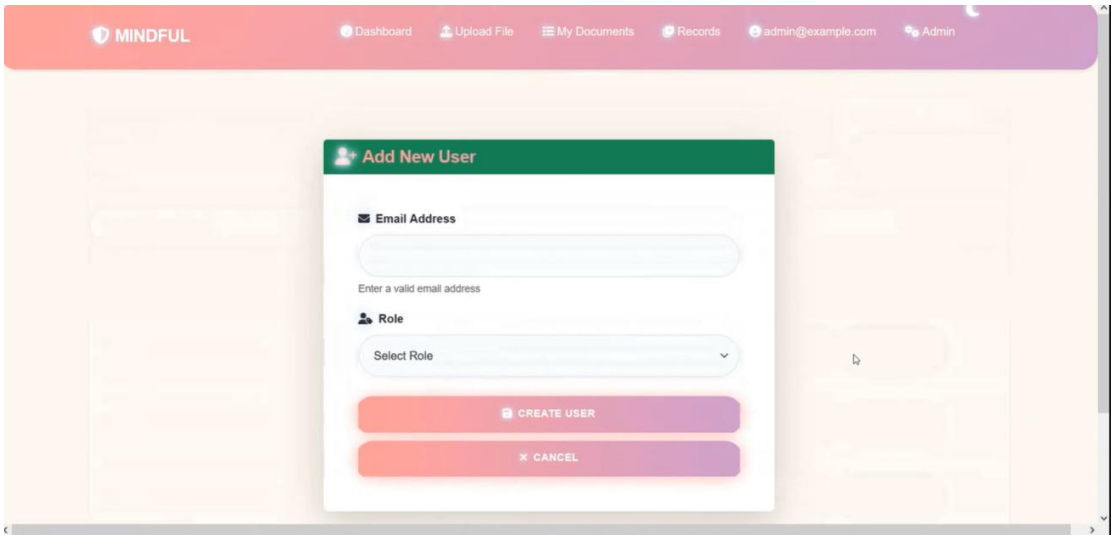
13- Admin Dashboard: the admin can do the same functionalities as the user, but the admin have more control as he has the admin panel that a regular user cannot access.



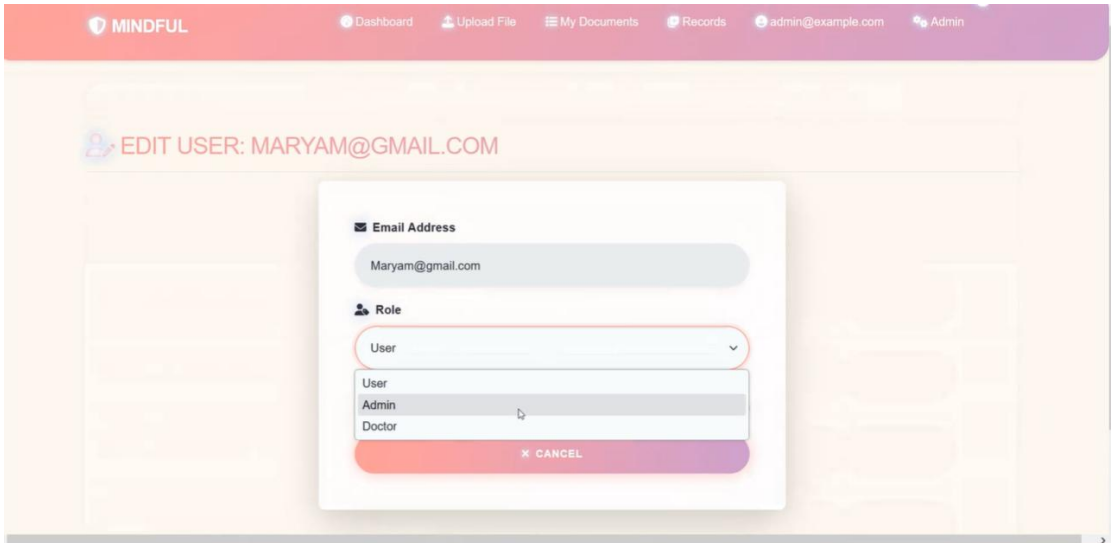
14- User Management: Here the admin can see all the users on the system, also he can edit/add/delete any user.



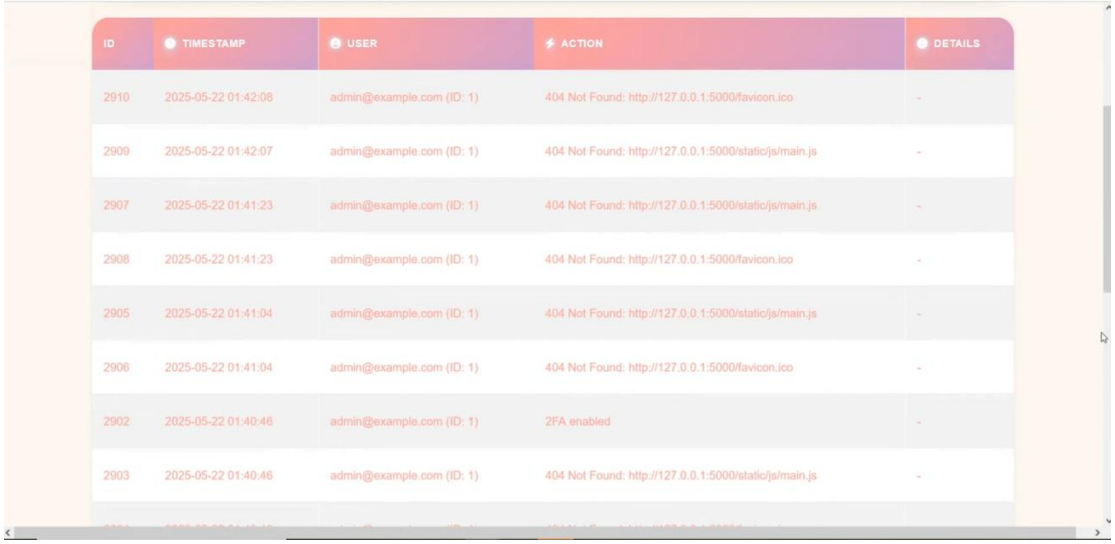
15- Cont. User Management: the admin can add any user and choose his role (all created users via admin is set with a default password that the user can change it later).



16- Cont. User Management: the admin can edit any user by changing his/her role.



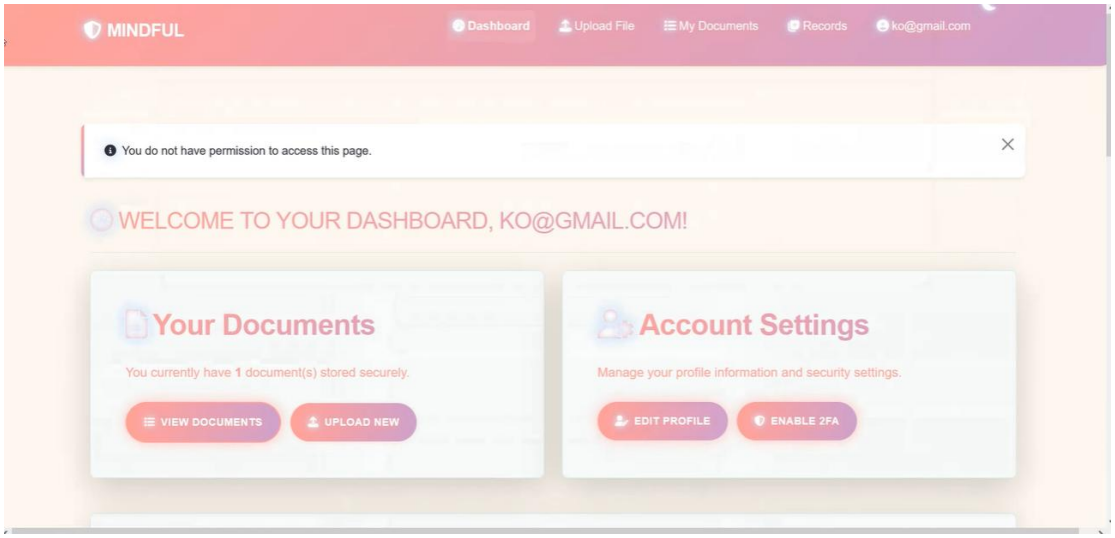
17-Audit logs: here the admin can view all actions that happens on the system (View login history, file uploads, and suspicious actions, session expiration)



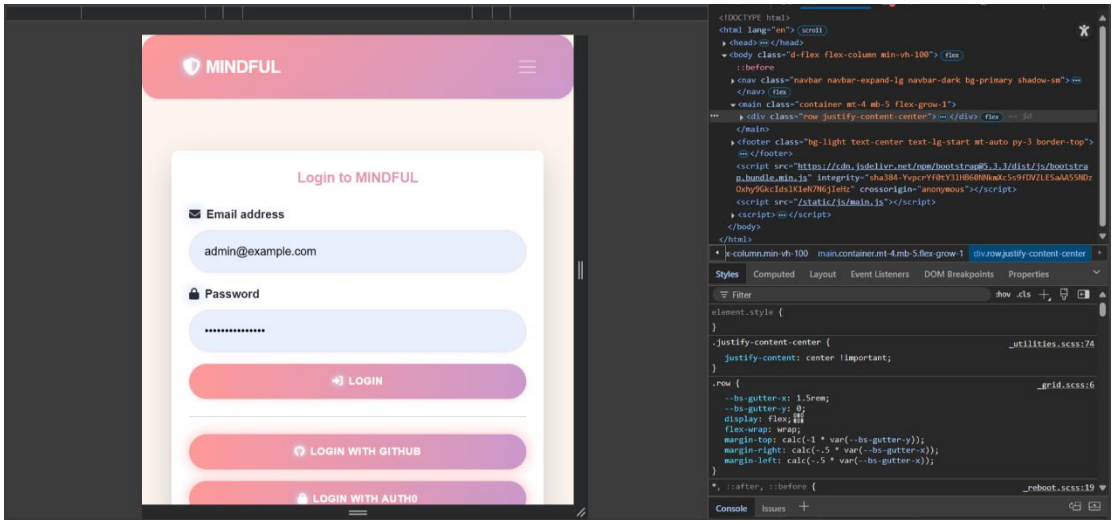
ID	TIMESTAMP	USER	ACTION	DETAILS
2910	2025-05-22 01:42:08	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/favicon.ico	-
2909	2025-05-22 01:42:07	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/static/js/main.js	-
2907	2025-05-22 01:41:23	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/static/js/main.js	-
2908	2025-05-22 01:41:23	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/favicon.ico	-
2905	2025-05-22 01:41:04	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/static/js/main.js	-
2906	2025-05-22 01:41:04	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/favicon.ico	-
2902	2025-05-22 01:40:46	admin@example.com (ID: 1)	2FA enabled	-
2903	2025-05-22 01:40:46	admin@example.com (ID: 1)	404 Not Found: http://127.0.0.1:5000/static/js/main.js	-

18-Secure Admin Endpoints: Access to Admin pages and features is restricted such as http://127.0.0.1:5000/admin/security/audit_logs

If an attacker tries to enter this endpoint it shows him a message that he has no permission to access this route.



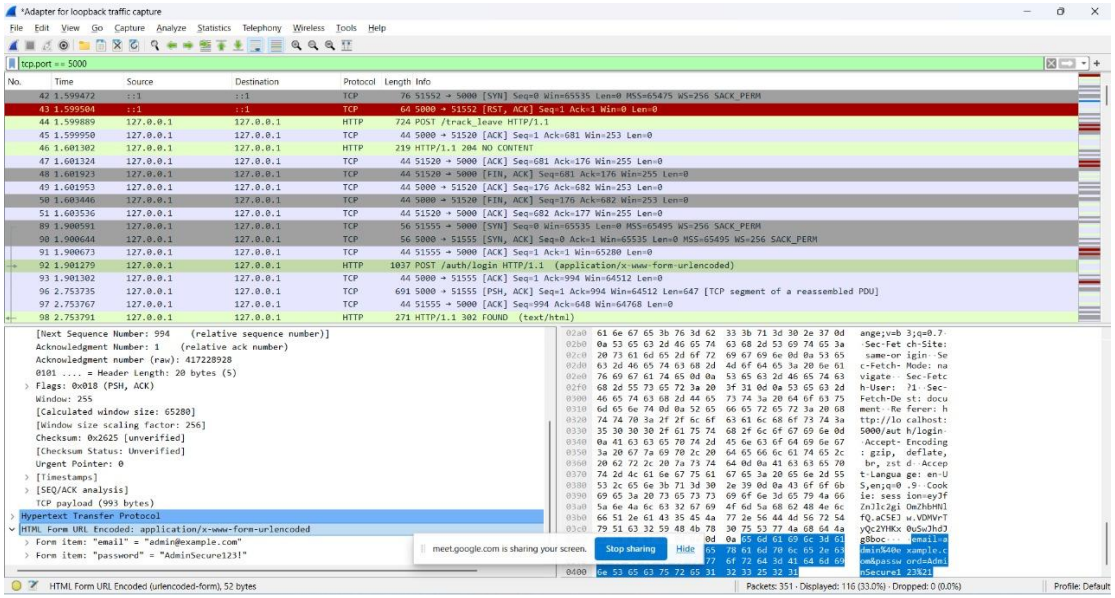
19- UI Requirements: Clean, modern UI using Bootstrap & Responsive (works on desktop and mobile)

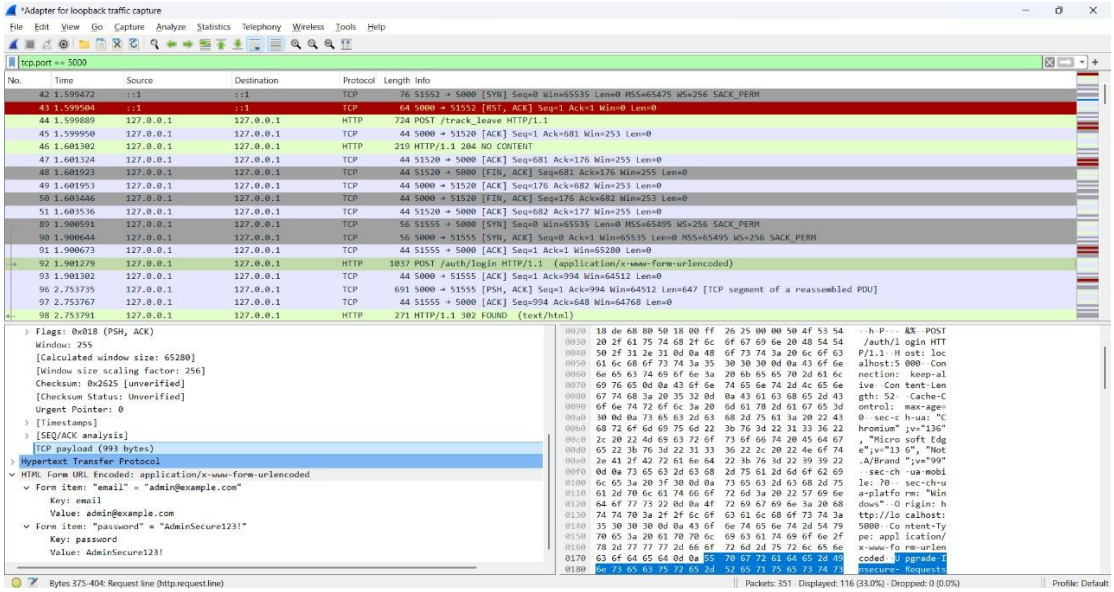


20-we will simulate MITM attacks using Wireshark to demonstrate protection via HTTPS & Shows intercepted vs. protected traffic.

We will run on HTTP first.

Here, as you can see all the credentials are not encrypted anyone can see it so, it can cause man in the middle attack as he can intercept the traffic.





21- HTTPS & Certificate Management: First, Configure local SSL/TLS using OpenSSL Via this command:

(openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout server.key -out server.crt)

Then, Run the app over HTTPS.

As you can see the Credentials is encrypted so, that secures the traffic & protect from MITM.