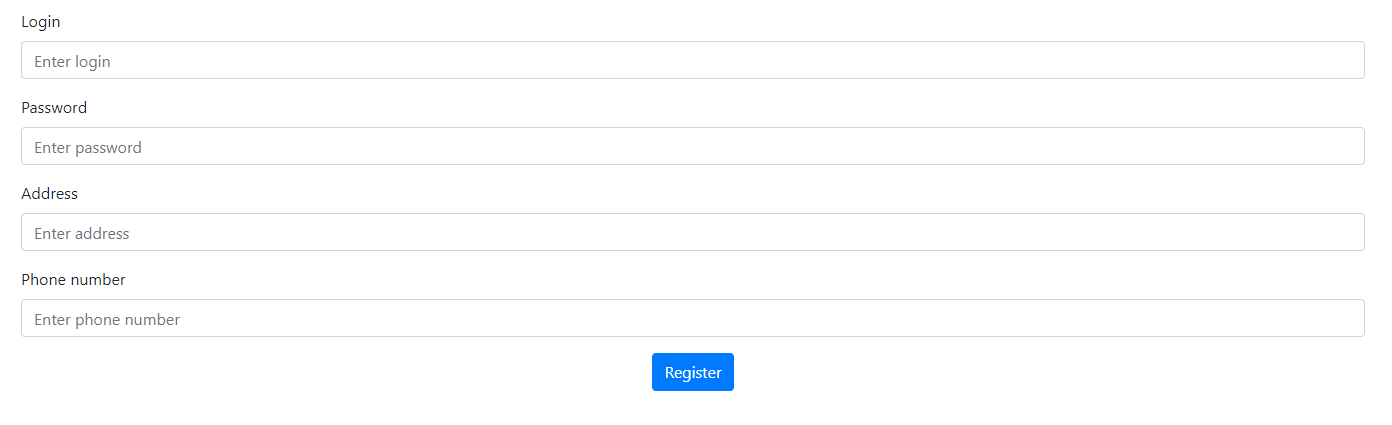
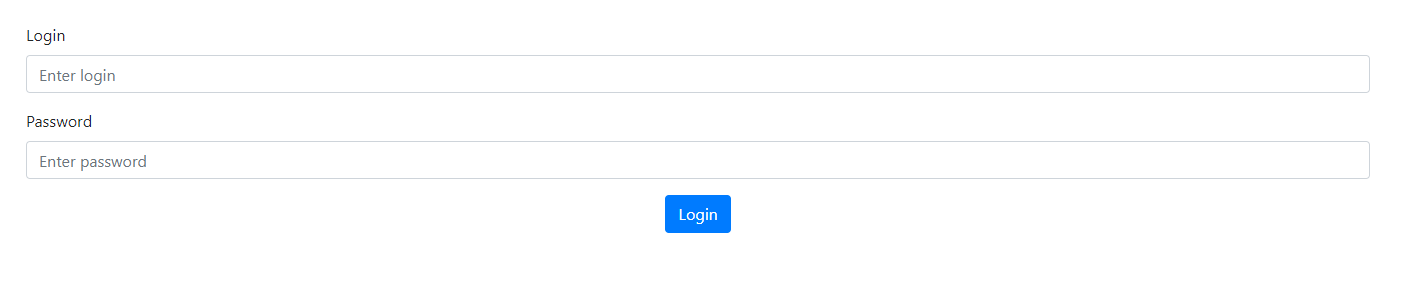
**ЗВІТ 5-6 ЛР**

На цих лабораторних ми розробили NodeJS + Express аплікуху, яка має 2 сторінки: реєстрація + логін та два відповідних ендпоінта (реєстрація + логін):

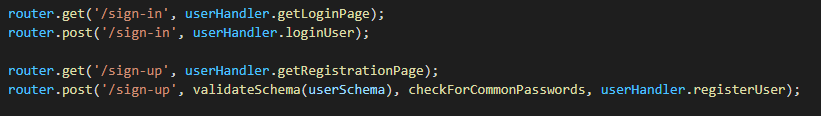


Реєстрація



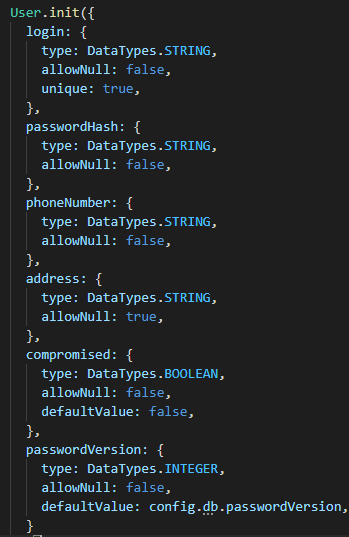
Логін

Просто банально зроблено 2 форми на двох сторінках з відповідними полями.

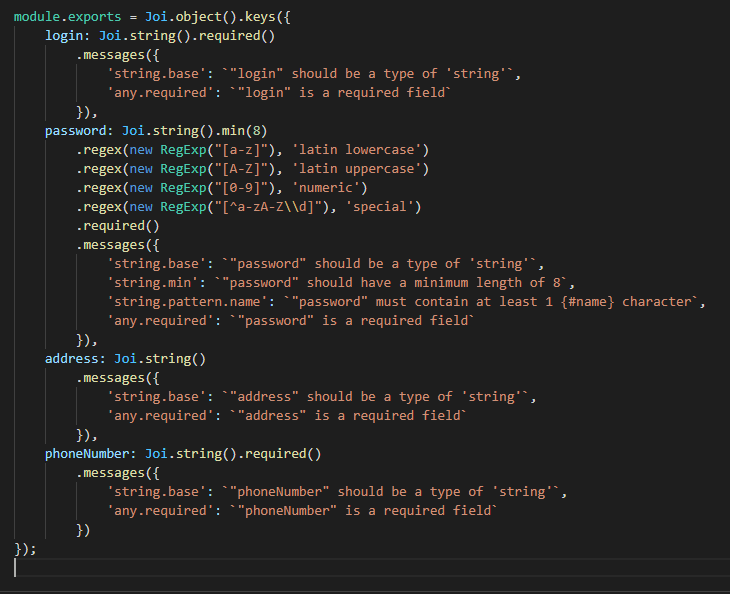


Роути

Також були добавлені поля для версії паролів та булеве поле compromised:



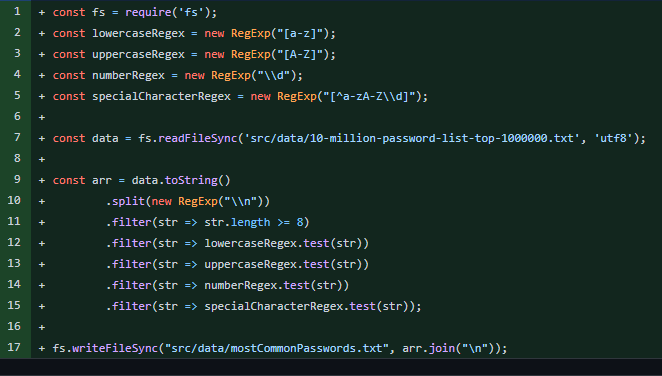
Добавлена валідація на сторінку реєстрації:



Валідація

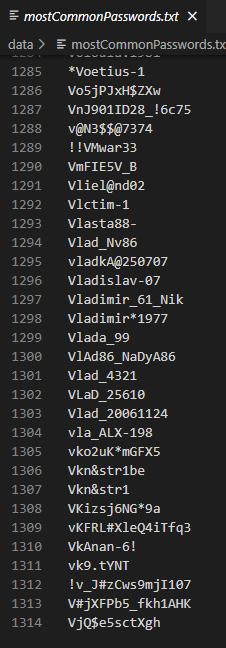
Валідація на пароль: 1 ловеркейс символ, 1 апперкейс символ, 1 цифра та один спец символ. Також мінімальна довжина 8 символів.

Також використовується ще одна перевірка для пароля: було взято 1,000,000 найпопулярніших паролів, відфільтровано по нашим вимогам до паролів:



Створення файлу найпопулярніших паролів

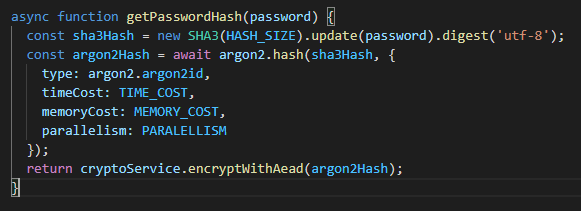
Вийшов файл з 1314 паролями:

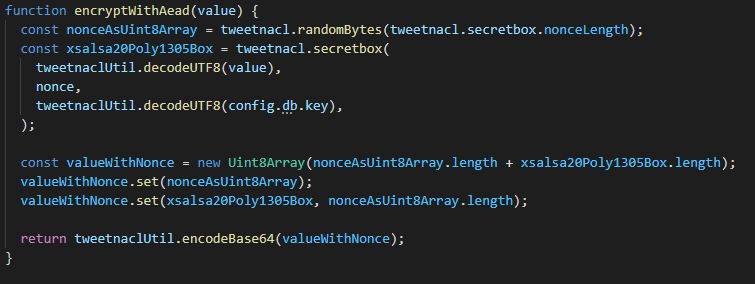


І добавлена перевірка, якщо введений пароль еквівалентний паролю з цього файла, то викидається помилка про те, що цей пароль не є надійним.



Для зберігання паролів була вибрана наступна схема: SHA3(512) -> argon2id -> XSalsa20-Poly1305.





Для аргон2 вибрані наступні опції (мінімальні по OWASP cheatsheets):



Також вразливі поля (address, phone) були зашифровані тією ж самою AEAD (XSalsa20-Poly1305). Ключ як і всі інші конфіги (дані про БД) були збережені на локальному файлі. Зберігаємо ключ як plain text, що є дуже погано (ми знаємо). Як аналог можна використовувати envelope encryption, де ми маємо багато DEK (ключі для шифрування даних) та наприклад один KEK (ключ для шифрування DEK), який ми зберігаємо на KMS наприклад в Amazon. DEK зашифровані KEK, отже наші ключі не зберігаються у відкритому вигляді. Взагалі зберігання ключів на KMS є досить гарною ідеєю, але всеодно якщо зловмисник влізе на наш сервер він зможе запросто отримати доступ до всіх ключів (банально кинути запити до KMS з нашого сервера). Також можна зберігати ключі на криптографічному модулі. Плюси використання KMS: центральне управління ключами, легка заміна ключів, безпечна генерація ключів, зловмиснику важче отримати доступ до них.

Якщо ж дивитись з точки зору того, що в нас просто злили БД (ключ залишився невідомим), можна сказати що ми знаходимось у захисті. Тільки нам надійшла ця інформація, ми ставимо наші поля compromised в true та просимо юзерів змінити паролі. Також для виявлення зловмисників можна використовувати honey pot, який дозволяє аналізувати наміри зловмисників (дивитись як вони пробують взламати сайт чи софт). Ще одне рішення для виявлення того що щось пішло не так, можна створити наприклад 100 юзерів з легкими паролями і якщо хтось на них залогіниться, відразу сигналізувати про це і думати хто це зайшов і з якими намірами.

