НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

"На правах рукопису"
УДК _____

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“___”_____2024 р.

Магістерська дисертація

на здобуття ступеня другого (магістерського) рівня вищої освіти
за освітньою програмою “Цифрові технології в енергетиці”
зі спеціальності 122 “Комп’ютерні науки”

на тему “Моделювання систем з безпарольною автентифікацією на основі промислових стандартів”

Виконала: студентка _2__курсу, групи ТР-32мп
___Дяк Анна Михайлівна_____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент каф. цифрових технологій
в енергетиці, к.ф.-м.н., доц., Юрій ТАРНАВСЬКИЙ_ _____
(посада, науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ) (підпис)

Рецензент доцент каф. інженерії програмного
забезпечення в енергетиці, к.т.н., доц., Іван ВАРАВА_ _____
(посада, науковий ступінь, вчене звання, ім’я ПРІЗВИЩЕ) (підпис)

Н.контроль доцент каф. цифрових технологій в
енергетиці, д.ф., Артем ГУРІН_____ _____
(посада, ім’я ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____
(підпис)

Київ - 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – другий (магістерський)
спеціальність 122 “Комп’ютерні науки”
Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА

«___»_____2024 р.

ЗАВДАННЯ
на дипломну роботу студенту

Дяк Анні Михайлівні
(прізвище, ім’я, по батькові)

1. Тема роботи Моделювання систем з безпарольною автентифікацією на основі промислових стандартів

керівник роботи *Тарнавський Юрій Адамович, к.ф.-м.н., доцент*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 11 » листопада 2024 р.
№ 5040-с

2. Термін подання роботи студентом 06.12.2024 року

3. Вихідні дані до роботи Java, Java Script, CSS HTML, середовище розробки IntelliJ IDEA, база даних PostgreSQL.

4. Перелік питань, які потрібно розробити дослідження існуючих стандартів безпарольної автентифікації, аналіз архітектурних рішень для впровадження WebAuthn у сучасні інформаційні системи, розробка моделі безпарольної автентифікації, реалізація тестової системи з імплементацією моделі.

5. Орієнтований перелік ілюстративного матеріалу 1. Постановка задачі моделювання систем з безпарольною автентифікацією на основі промислових стандартів – 3 рисунки, 3. Реалізація системи безпарольної автентифікації – 20 рисунків.

6. Дата видачі завдання «15» вересня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури за темою дипломної роботи	1 вересня 2024	
2	Аналіз наявних рішень в області безпарольної автентифікації	15 вересня 2024	
3	Порівняльний аналіз методів безпарольної автентифікації	18 вересня 2024	
4	Побудова архітектурної моделі системи безпарольної автентифікації	25 вересня 2024	
5	Розробка ПЗ за архітектурною моделлю	30 вересня 2024	
6	Тестування розробленого ПЗ	15 жовтня 2024	
7	Підготовка матеріалів магістерської дисертації	20 жовтня 2024	
8	Передзахист	28 листопада 2024	
9	Захист	16 грудня 2024	

Студент _____ Анна ДЯК
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник роботи _____ Юрій ТАРНАВСЬКИЙ
(підпис) (ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Актуальність теми. У сучасному цифровому суспільстві традиційні паролі часто стають об'єктом фішингових атак та несанкціонованого доступу, що підкреслює необхідність впровадження більш надійних методів автентифікації. Тому дослідження та впровадження безпарольних систем автентифікації на основі WebAuthn є актуальними для підвищення рівня кібербезпеки та зниження ризиків, пов'язаних із використанням паролів.

Мета дипломної роботи. Розробка програмних засобів безпарольної автентифікації в інформаційних системах з метою підвищення рівня безпеки користувацьких даних.

Завдання дослідження. Дослідити існуючі стандарти безпарольної автентифікації, проаналізувати архітектурні рішення для впровадження WebAuthn у сучасні інформаційні системи, розробити моделі безпарольної автентифікації та реалізувати тестову систему з імплементацією моделі.

Методи дослідження включають експериментальне моделювання для тестування безпарольної системи автентифікації з різними токенами в умовах, наближених до реальних. Також застосовано архітектурне моделювання, що дозволило оцінити сумісність стандарту WebAuthn із існуючими системами. Емпіричне тестування проводилось із використанням реальних токенів для перевірки надійності автентифікації. Метод аналізу системної безпеки допоміг виявити переваги та ризики стандартів WebAuthn і FIDO2, а аналіз продуктивності дозволив оцінити ефективність системи під час обробки запитів.

Наукова новизна одержаних результатів полягає в розробці та впровадженні архітектурної моделі системи безпарольної автентифікації на основі стандарту WebAuthn, що забезпечує підвищений рівень безпеки користувацьких даних та знижує ризики, пов'язані з використанням традиційних паролів.

Практичне значення одержаних результатів роботи полягає у впровадженні системи безпарольної автентифікації на основі стандарту WebAuthn, що підвищує безпеку користувацьких даних та знижує ризики,

пов'язані з використанням традиційних паролів. Розроблена архітектурна модель та реалізована серверна частина системи можуть бути інтегровані в сучасні інформаційні системи, забезпечуючи сумісність із існуючими інфраструктурами та підвищуючи загальний рівень кібербезпеки. Це сприяє захисту персональних даних користувачів та зменшенню вразливостей до фішингових атак і несанкціонованого доступу.

Апробація результатів дослідження. Основні положення роботи обговорювались на:

Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем: матеріали міжнародної науково-практичної конференції, 13–16 червня 2024. Луцьк-Світязь. 2024.

Публікації. Результати роботи були опубліковані на:

Тарнавський Ю.А., Дяк А.М. Моделювання систем з безпарольною автентифікацією на основі промислових стандартів. Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем: тези І міжнародної наукової-практичної конференції, 13-16 червня 2024. Луцьк-Світязь. 2024. - С.27.

Повний обсяг дисертації складає з 97 сторінок, в тому числі 88 сторінок основного тексту, 27 таблиць, 26 рисунків, 4 сторінки списку використаних джерел у кількості 40 найменувань.

Ключові слова: безпарольна автентифікація, інформаційна безпека, біометричні дані, криптографія, токени безпеки, користувацькі дані, кіберзагрози.

ABSTRACT

Relevance of the topic. In today's digital society, traditional passwords are often the target of phishing attacks and unauthorized access, emphasizing the need for more reliable authentication methods. Therefore, research and implementation of passwordless authentication systems based on WebAuthn are relevant to enhancing cybersecurity and reducing risks associated with password use.

The objective of the thesis is a development of software tools for passwordless authentication in information systems to improve the security of user data.

Research tasks are to investigate existing standards of passwordless authentication, analyse architectural solutions for implementing WebAuthn in modern information systems, develop passwordless authentication models and implement a test system with the model implementation.

Research methods. The research employed experimental modeling to test the passwordless authentication system with various tokens in near-real conditions. Architectural modeling was applied to assess the compatibility of the WebAuthn standard with existing systems. Empirical testing with real tokens was conducted to verify authentication reliability. Security analysis identified the strengths and risks of WebAuthn and FIDO2 standards, while performance analysis evaluated the system's efficiency in processing requests.

The scientific novelty of the obtained results lies in the development and implementation of an architectural model for a passwordless authentication system based on the WebAuthn standard, ensuring enhanced user data security and reducing the risks associated with traditional passwords.

The practical significance of the obtained work results lies in implementing a passwordless authentication system based on the WebAuthn standard, which increases user data security and reduces password-related risks. The developed architectural model and implemented server side can be integrated into modern information systems, ensuring compatibility with existing infrastructures and enhancing overall

cybersecurity. This contributes to the protection of user data and reduces vulnerabilities to phishing attacks and unauthorized access.

Approbation of the results. The main provisions of this work were discussed at:

Issues in Computer Science, Software Modeling, and Digital Systems Security: Proceedings of the International Scientific-Practical Conference, June 13–16, 2024. Lutsk-Svityaz. 2024.

Publications. The results were published in:

Tarnavskyi Y.A., Diak A.M. Modeling Passwordless Authentication Systems Based on Industrial Standards. Issues in Computer Science, Software Modeling, and Digital Systems Security: Proceedings of the First International Scientific-Practical Conference, June 13–16, 2024. Lutsk-Svityaz, 2024. - P. 27.

The total length of the dissertation is 97 pages, including 88 pages of main text, 27 tables, 26 figures, and 4 pages of references totaling 40 sources.

Keywords: passwordless authentication, information security, biometric data, cryptography, security tokens, user data, cybersecurity.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ ТА ТЕРМІНІВ

FIDO – Fast Identity Online – технічна специфікація для автентифікації користувачів, яка використовується в таких ситуаціях, як вхід за відбитком пальця або двофакторна автентифікація.

WebAuthn – Web Authentication – це вебстандарт, створений консорціумом W3C спільно з альянсом FIDO, що надає вебсайтам та онлайн-сервісам можливість автентифікувати користувачів за допомогою криптографії з відкритим ключем.

CTAP – Client to Authenticator Protocol – це специфікація, розроблена Альянсом FIDO, яка визначає, як зовнішні автентифікатори - такі як апаратні ключі безпеки або мобільні пристрої - взаємодіють з клієнтськими платформами, такими як браузери або операційні системи.

TPM – Trusted Platform Module – це спеціальний мікроконтролер, призначений для захисту обладнання за допомогою вбудованих криптографічних ключів.

YubiKey – це апаратний пристрій автентифікації, розроблений Yubico для підвищення безпеки комп'ютерів, мереж та онлайн-сервісів.

MitM – Man-in-the-Middle – це тип кібернападу, коли зловмисник перехоплює і потенційно змінює комунікацію між двома сторонами, які вважають, що спілкуються безпосередньо одна з одною

Автентифікатор – це пристрій або додаток, який підтверджує особу користувача, не вимагаючи традиційного пароля

ВСТУП

Сучасні методи захисту інформаційних систем є необхідною умовою для безпечного функціонування цифрового суспільства, особливо в умовах зростання кіберзагроз. Захист персональних даних та конфіденційної інформації стає пріоритетним завданням, що сприяє пошуку нових підходів у сфері автентифікації. Одним з перспективних напрямів є використання безпарольних систем автентифікації, які значно підвищують захищеність облікових записів користувачів. Важливим інструментом у цьому процесі є стандарт WebAuthn, розроблений для безпечної автентифікації з використанням асиметричної криптографії та біометрії. Цей стандарт пропонує ефективну альтернативу традиційним паролям, які мають численні недоліки, зокрема вразливість до фішингових атак та ризик несанкціонованого використання.

Традиційні способи автентифікації, що базуються на паролях, часто піддаються атакам, таким як перехоплення чи підбір, що дозволяє неавторизованим особам отримувати доступ до облікових записів. У зв'язку з цим впровадження безпарольних технологій автентифікації є важливим завданням у кібербезпеці. Хоча стандарти FIDO2 та WebAuthn значно розширили можливості безпарольної автентифікації, їх ефективна інтеграція в реальні інформаційні системи потребує додаткових досліджень, особливо з огляду на сумісність з існуючими інфраструктурами, де можуть виникати технічні виклики.

Мета дипломної роботи на тему: “Моделювання систем з безпарольною автентифікацією на основі промислових стандартів” полягає у розробці програмних засобів безпарольної автентифікації в інформаційних системах з метою підвищення рівня безпеки користувацьких даних.

Для досягнення мети було сформовано наступні **завдання**:

- дослідження існуючих стандартів безпарольної автентифікації;
- аналіз архітектурних рішень для впровадження WebAuthn у сучасні інформаційні системи;
- розробка моделі безпарольної автентифікації;

- реалізація тестової системи з імплементацією моделі.

Методи дослідження включають наступні:

- метод експериментального моделювання для проведення серії тестувань розробленої системи безпарольної автентифікації з різними апаратними токенами для перевірки її працездатності та ефективності в умовах, наближених до реальних;
- метод архітектурного моделювання для створення архітектурної моделі системи, що дозволяє дослідити взаємодію її компонентів та оцінити сумісність стандарту WebAuthn з існуючими інформаційними системами;
- метод емпіричного тестування дозволяє провести тестування працездатності реалізованої системи з використанням реальних токенів, таких як YubiKey, для визначення ефективності автентифікації та оцінки захищеності від можливих атак;
- метод аналізу системної безпеки використаний для аналізу стандартів WebAuthn та FIDO2 з точки зору захисту від фішингових атак і компрометації даних, що дозволяє визначити основні переваги та ризики використання безпарольної автентифікації;
- метод аналізу продуктивності, який допомагає виміряти основні показники продуктивності системи під час обробки запитів на автентифікацію, що дало змогу оцінити її ефективність та здатність витримувати навантаження.

Результати дослідження мають **практичне значення** для організацій, які прагнуть підвищити рівень безпеки своїх систем, замінивши традиційні паролі безпарольними методами автентифікації. Розроблена архітектурна модель і впроваджені технології дозволять організаціям реалізувати безпарольний вхід для користувачів, підвищуючи захищеність персональних даних.

Апробація результатів дослідження. Основні положення роботи доповідались і обговорювались на:

Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем: матеріали міжнародної науково-практичної конференції, 13–16 червня 2024. Луцьк-Світязь. 2024.

Публікації. Результати роботи були опубліковані на:

Тарнавський Ю.А., Дяк А.М. Моделювання систем з безпарольною автентифікацією на основі промислових стандартів. Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем: тези І міжнародної наукової-практичної конференції, 13-16 червня 2024. Луцьк-Світязь. 2024.- С.27.

Магістерська дисертація складається зі списку термінів, скорочень та умовних позначень, вступу, 4 розділів, висновків до кожного розділу, загальних висновків, списку використаних джерел, який містить 40 елементів та додатку А на сторінці 94. Загальний обсяг дисертації складає 97 сторінок. Основний зміст викладено на 88 сторінках. Робота містить 27 таблиць та 26 рисунків.

1 ПОСТАНОВКА ЗАДАЧІ МОДЕЛЮВАННЯ СИСТЕМ З БЕЗПАРОЛЬНОЮ АВТЕНТИФІКАЦІЄЮ НА ОСНОВІ ПРОМИСЛОВИХ СТАНДАРТІВ

У цьому розділі розглядаються основні аспекти проблеми використання паролів у сучасних інформаційних системах та необхідність переходу до безпарольних методів автентифікації. Окрім того, наведено огляд існуючих технологій безпарольної автентифікації, що базуються на промислових стандартах, таких як WebAuthn та FIDO2. Додатково проаналізуємо готові рішення, що базуються на даних стандартах. Метою даного розділу є формулювання основних вимог до систем безпарольної автентифікації та визначення завдань для моделювання таких систем на основі сучасних стандартів безпеки.

1.1 Проблеми паролів у сучасних інформаційних системах

У сучасному цифровому середовищі паролі залишаються одним із найпоширеніших методів автентифікації, проте їх використання породжує низку значних проблем, які створюють ризики для безпеки даних як для користувачів, так і для організацій. Основні виклики, пов’язані з паролями, полягають у їхній вразливості до різних видів атак, незручності у використанні, а також необхідності управління великою кількістю облікових записів.

По-перше, вразливість до атак є однією з найбільших загроз для безпеки паролів. В сучасних умовах хакери використовують різні методи для отримання доступу до паролів, включаючи фішинг, атаки типу brute-force (груба сила), та перехоплення даних [1]. Фішинг, зокрема, є дуже поширеним методом, який дозволяє зловмисникам отримувати конфіденційну інформацію, вводячи користувачів в оману через підроблені веб-сайти або електронні листи. Це

призводить до витоків даних, що може мати катастрофічні наслідки як для окремих осіб, так і для великих організацій.

Наступною проблемою є повторне використання паролів на різних інформаційних системах. Багато користувачів, згідно дослідження Google – більше половини, використовують однакові або подібні паролі для кількох облікових записів, що значно збільшує ризик компрометації [2]. Якщо один обліковий запис зламано, зловмисники можуть легко отримати доступ до інших сервісів, що використовують той самий пароль. Така практика є небезпечною, особливо в корпоративних середовищах, де доступ до важливих даних може бути здійснений через скомпрометований особистий обліковий запис працівника.

Окрім ризиків з безпеки, паролі створюють суттєві проблеми з точки зору зручності та користувацького досвіду. Користувачам часто доводиться запам'ятовувати велику кількість складних паролів для різних облікових записів, що є незручним та призводить до використання слабких або легко передбачуваних паролів. Це також сприяє використанню таких небезпечних практик, як записування паролів або зберігання їх у незахищених місцях.

Регулярна зміна паролів, яка рекомендована багатьма організаціями, також додає незручностей для користувачів. Часті зміни паролів часто призводять до того, що користувачі використовують незначні варіації старих паролів або записують нові паролі, щоб уникнути їх забування, що знижує загальну безпеку системи.

З точки зору організацій, управління паролями стає ще однією проблемою. Для забезпечення належного рівня безпеки компанії повинні впроваджувати політики складності паролів, частого їх оновлення та багатофакторної автентифікації. Проте, навіть за цих умов, злом облікових записів через скомпрометовані паролі залишається поширеною загрозою.

Зрештою, економічний аспект також є важливим чинником. Витрати на відновлення після компрометації даних через втрачені або викрадені паролі можуть бути значними. Згідно зі звітами про інциденти кібербезпеки, паролі часто є слабкою ланкою, яка веде до масштабних витоків даних та втрат репутації організацій.

Таким чином, проблема паролів у сучасних інформаційних системах є важливою як з точки зору кібербезпеки, так і з позиції зручності користувачів. Рішення, засновані на використанні безпарольної автентифікації, такі як WebAuthn та FIDO2, відкривають нові можливості для захисту систем від зловмисників, підвищуючи рівень безпеки та зменшуючи залежність від людського фактора. Впровадження цих технологій є кроком до побудови більш надійних і зручних для користувачів систем автентифікації, що робить цю тему надзвичайно актуальною для дослідження та розробки.

1.2 Стандарт FIDO

Стандарт FIDO (Fast Identity Online) є важливою технологією для безпарольної автентифікації, яка допомагає зменшити залежність від паролів та підвищити безпеку в інформаційних системах. Створений у 2012 році, FIDO Alliance об'єднав ключових гравців в індустрії для розробки відкритих і сумісних технічних специфікацій, що дозволяють використовувати біометричні методи автентифікації та криптографію з відкритими ключами. Цей стандарт був заснований на основі ідеї, що паролі є небезпечними та незручними, і вимагає заміни на більш безпечні та зручні рішення.Принцип роботи ключів зображено на рисунку 1.1

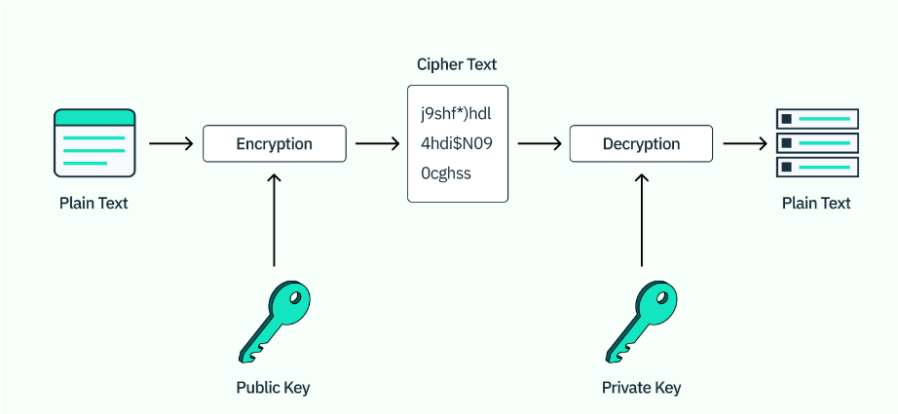


Рисунок 1.1 – Абстрактна діаграми роботи ключів в стандарті FIDO

За рисунком 1.1, автентифікація за стандартом FIDO використовує пару ключів — приватний ключ зберігається на пристрої користувача, а публічний ключ передається серверу. Приватний ключ використовується для підписання викликів (challenge), які сервер відправляє для перевірки автентичності користувача. Завдяки цьому механізму сервер може впевнитися, що користувач є тим, за кого себе видає, без необхідності збереження пароля на сервері

Оскільки приватний ключ зберігається на пристрої користувача або в апаратному модулі безпеки, процес автентифікації стає локалізованим і більш захищеним від віддалених атак. Для підтвердження особи користувач може використовувати біометричні дані, такі як відбитки пальців або розпізнавання обличчя, що робить процес зручним і безпечним.

FIDO Alliance не лише розробляє технічні специфікації, а й сертифікує продукти, що використовують ці стандарти, забезпечуючи їх якість і відповідність вимогам безпеки. Усі специфікації FIDO також подаються для стандартизації через організації, що сприяє популяризації безпарольної автентифікації серед розробників та компаній.

1.3 Стандарт WebAuthn

WebAuthn (Web Authentication) — це стандарт, розроблений консорціумом W3C, який забезпечує безпечну автентифікацію користувачів без використання паролів.

WebAuthn дозволяє веб-додаткам автентифікувати користувачів через асиметричні криптографічні ключі замість традиційних паролів. Процес автентифікації відбувається через створення пари ключів: приватного та публічного. Приватний ключ зберігається на пристрої користувача (наприклад, на токені або апаратному ключі), а публічний ключ передається на сервер [3].

Процес реєстрації користувача передбачає створення нових облікових даних на основі наданого імені користувача. Це основа безпарольного входу, який підтримує WebAuthn. Його можна детальніше розглянути на рисунку 1.2.

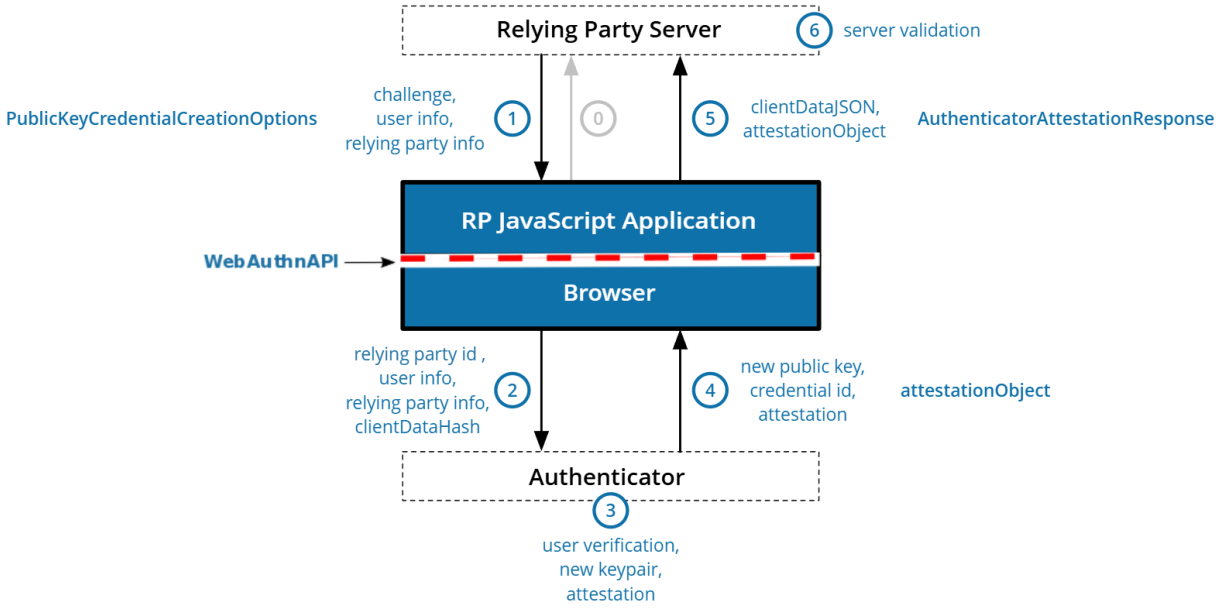


Рисунок 1.2 – Послідовна діаграма процесу реєстрації на основі WebAuthn

Розглянемо опис послідовності кроків на діаграмі:

1. Ініціація процесу реєстрації. Користувач заходить на веб-сайт, який підтримує WebAuthn, і натискає кнопку "Зареєструватися". Ця дія ініціює запит до сервера автентифікації (Relying Party), який відповідає за обробку запитів на реєстрацію автентифікаторів.

2. Генерація виклику сервером. Сервер створює випадковий набір даних, відомий як "виклик" (challenge), який використовується для унікальності запиту. Разом із викликом сервер формує об'єкт PublicKeyCredentialCreationOptions, який містить інформацію про сервер (наприклад, доменне ім’я), ідентифікатор користувача, підтримувані криптографічні алгоритми, тайм-аут і налаштування автентифікатора. Цей об’єкт надсилається браузеру користувача.

3. Передача даних до браузера. Браузер отримує дані від сервера і передає їх локальному автентифікатору або зовнішньому апаратному пристрою (наприклад, YubiKey, Touch ID, Face ID).

4. Взаємодія з автентифікатором. Автентифікатор запитує підтвердження дій від користувача. Наприклад, користувач може ввести PIN-код,

надати скан відбитка пальця або скористатися розпізнаванням обличчя. Цей крок гарантує, що операцію ініціює саме користувач.

5. Генерація ключів автентифікатором. Після успішного підтвердження автентифікатор створює пару ключів. Приватний ключ залишається збереженим на пристрої користувача. Публічний ключ і додаткові атестаційні дані передаються браузеру для подальшої обробки.

6. Формування атестації. Автентифікатор формує атестаційний об'єкт (attestationObject), який містить публічний ключ, інформацію про автентифікатор (тип, модель, виробника) і цифровий підпис. Цей об'єкт разом із даними браузера (clientDataJSON) передається на сервер.

7. Передача даних на сервер. Браузер відправляє дані автентифікації на сервер, включаючи атестаційний об'єкт, публічний ключ та унікальний ідентифікатор автентифікатора (id).

8. Перевірка атестації сервером. Сервер перевіряє відповідність виклику (challenge) отриманим даним, дійсність підпису, використовуючи атестаційні дані та інформацію про автентифікатор, щоб переконатися у його легітимності. Після успішної перевірки сервер зберігає публічний ключ, ідентифікатор користувача і пов’язані метадані в базі даних.

9. Завершення реєстрації. Після успішної перевірки сервер повідомляє користувача, що процес реєстрації завершено. Тепер користувач може використовувати автентифікатор для майбутніх сеансів входу в систему.

Цей процес забезпечує безпеку, оскільки приватний ключ залишається на пристрої користувача і ніколи не передається через мережу, а сервер зберігає лише публічний ключ, необхідний для перевірки.

Процес автентифікації можна підсумувати, як зазначено на рисунку 1.3.

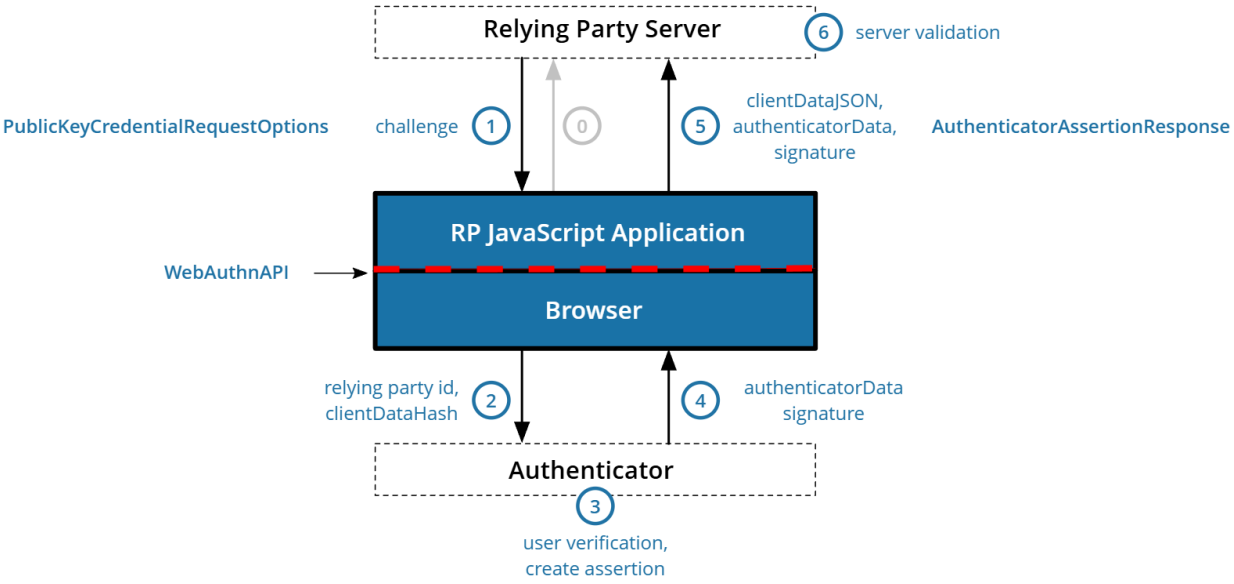


Рисунок 1.3 – Абстрактний опис процесу автентифікації на основі стандарту WebAuthn

Розглянемо опис послідовності кроків на діаграмі:

1. Ініціація процесу автентифікації. Користувач заходить на веб-сайт і натискає кнопку "Увійти". Ця дія створює запит до сервера автентифікації (Relying Party), який перевіряє, чи існує обліковий запис користувача, і ініціює процес автентифікації.

2. Генерація виклику сервером. Сервер генерує випадковий набір даних, відомий як "виклик" (challenge), який забезпечує унікальність запиту та запобігає повторним атакам. Сервер також відправляє інформацію про обліковий запис користувача (наприклад, ідентифікатор зареєстрованого автентифікатора) разом із об’єктом PublicKeyCredentialRequestOptions, який включає: challenge – унікальний виклик для цього сеансу, rpId – ідентифікатор сервера (домен), allowCredentials – список ідентифікаторів автентифікаторів, які можна використовувати для автентифікації, timeout – час очікування відповіді, userVerification – вимога до перевірки користувача (наприклад, біометрія або PIN).

3. Передача даних клієнту. Сервер передає об’єкт PublicKeyCredentialRequestOptions до браузера або клієнтського додатка. Браузер ініціює виклик автентифікатора через WebAuthn API.

4. Запит до автентифікатора. Браузер надсилає отримані дані локальному автентифікатору або зовнішньому пристрою (наприклад, YubiKey, Touch ID, Face ID). Автентифікатор ідентифікує користувача та запитує підтвердження дії.

5. Перевірка користувача автентифікатором. Користувач підтверджує свою особу за допомогою біометричних даних (відбиток пальця, розпізнавання обличчя тощо), ведення PIN-коду чи використання фізичного пристрою (наприклад, торкання до YubiKey).

6. Генерація підпису автентифікатором. Автентифікатор використовує зареєстрований приватний ключ для створення цифрового підпису на основі виклику (challenge) від сервера, даних клієнта (clientDataJSON), інших специфічних параметрів сеансу.

7. Повернення відповіді клієнту. Автентифікатор передає браузеру підписані дані разом із додатковою інформацією: authenticatorData – дані про автентифікатор, signature – цифровий підпис та clientDataJSON – дані, отримані від браузера.

8. Передача відповіді на сервер. Браузер надсилає ці дані назад на сервер. Сервер отримує об'єкт PublicKeyCredential, що включає ідентифікатор автентифікатора (id), дані підпису (authenticatorData, signature) та JSON-дані клієнта (clientDataJSON).

9. Перевірка підпису сервером. Сервер порівнює отриманий виклик (challenge) із тим, що було згенеровано під час початку сеансу. Далі сервер перевіряє цифровий підпис, використовуючи публічний ключ, збережений під час реєстрації. В кінці аналізує authenticatorData для підтвердження коректності відповіді автентифікатора.

10. Завершення автентифікації: Якщо всі перевірки успішно пройдено, сервер підтверджує автентифікацію користувача. Користувач отримує доступ до системи або захищених ресурсів.

Після аналізу діаграм, що відображають структуру та процеси автентифікації за допомогою WebAuthn, варто звернути увагу на загальні переваги й недоліки цього стандарту.

WebAuthn пропонує значні переваги, особливо з точки зору безпеки та зручності. Найбільш помітною перевагою є використання асиметричної криптографії, яка забезпечує високий рівень захисту. Оскільки приватний ключ зберігається на апаратному автентифікаторі, а публічний ключ передається на сервер, доступ до облікового запису захищений навіть у разі компрометації сервера [4]. Крім того, WebAuthn значно підвищує стійкість до фішингових атак, оскільки доменне ім’я, необхідне для автентифікації, зберігається на апаратному токені, а не на віртуальному сервері, що унеможливлює підробку. Також WebAuthn зменшує ризики пов'язані з витоком конфіденційних даних, оскільки система використовує лише публічні ключі і не зберігає біометричну інформацію на сервері.

Додатковою перевагою є сумісність з багатьма браузерами і пристроями, що забезпечує гнучкість та уникнення залежності від одного постачальника. Користувачі можуть самі обирати засоби автентифікації, такі як відбитки пальців або апаратні токени, що підвищує зручність та контроль над процесом автентифікації. Крім того, відсутність паролів робить взаємодію з системою менш стресовою для користувачів, оскільки їм не потрібно запам'ятовувати або керувати складними паролями.

Попри численні переваги, WebAuthn має і деякі суттєві недоліки. Обмежена крос-пристроєва сумісність є одним із головних викликів — користувачі не можуть легко переносити облікові дані з одного автентифікатора на інший. Це означає, що у випадку втрати або пошкодження автентифікатора може виникнути серйозна проблема з відновленням доступу до облікового запису. Окрім того, високі витрати на впровадження можуть бути перешкодою для багатьох організацій, оскільки потрібні апаратні пристрої, а також технічний персонал для їх обслуговування [5]. Також цей стандарт вимагає високого рівня технічної компетенції як від користувачів, так і від організацій для забезпечення належного функціонування та безпеки [6].

У розділі розглянуто ключові аспекти проблеми використання паролів у сучасних інформаційних системах, їхні основні вразливості та ризики, які виникають через залежність від цього методу автентифікації. Паролі довели свою

ненадійність через вразливість до атак, зокрема фішингу, brute-force, і повторного використання, що створює суттєві загрози для безпеки даних як для окремих користувачів, так і для організацій. Ці проблеми підкреслюють необхідність впровадження альтернативних методів автентифікації, які забезпечують як зручність, так і підвищений рівень захисту.

Аналіз сучасних стандартів, таких як FIDO2 і WebAuthn, демонструє їх потенціал у вирішенні цих проблем. Завдяки використанню асиметричної криптографії, ці технології дозволяють зберігати приватні ключі на пристроях користувачів, що усуває залежність від паролів і знижує ризики компрометації даних. Особлива увага приділена WebAuthn, який забезпечує сумісність з широким спектром браузерів і пристроїв, дозволяючи користувачам використовувати біометричні дані або апаратні токени для автентифікації.

Попри численні переваги, впровадження безпарольних методів вимагає вирішення певних викликів, зокрема щодо сумісності між пристроями, вартості реалізації та необхідності високого рівня технічної компетенції. Тим не менш, переваги у вигляді стійкості до фішингових атак, підвищення зручності для користувачів і забезпечення конфіденційності даних значно переважають.

Таким чином, безпарольна автентифікація на основі промислових стандартів є важливим кроком у напрямку до створення безпечних і зручних для користувачів інформаційних систем. Подальші дослідження та моделювання цих систем дозволять вдосконалити їхню інтеграцію в сучасні платформи та сприяти їхньому широкому впровадженню.

2 МЕТОДИ ТА ЗАСОБИ РОЗРОБКИ СИСТЕМ БЕЗПАРОЛЬНОЇ АВТЕНТИФІКАЦІЇ

У цьому розділі розглядаються основні методи та інструменти, що використовуються для розробки систем безпарольної автентифікації. Оскільки безпека і зручність користувачів є ключовими аспектами сучасних інформаційних систем, важливо використовувати новітні технології, які дозволяють усунути паролі як слабку ланку. Безпарольна автентифікація заснована на криптографічних методах, таких як використання публічних і приватних ключів, а також на сучасних засобах автентифікації, зокрема біометричних даних та апаратних токенів. Розробка таких систем потребує як математичних методів для забезпечення безпеки, так і програмних засобів для реалізації цих методів.

2.1 Основи безпарольної автентифікації

Основою будь-якої системи безпарольної автентифікації є асиметрична криптографія, яка забезпечує надійну автентифікацію без необхідності використання паролів. Ключі passkeys (або ключі доступу) — це метод, який застосовує пару криптографічних ключів: приватного, що зберігається на пристрої користувача, та публічного, що зберігається на сервері [7]. Під час автентифікації приватний ключ використовується для підписання запиту, а сервер перевіряє цей підпис за допомогою публічного ключа. Такий механізм гарантує, що дані користувача захищені на всіх етапах взаємодії .

Passkeys можуть бути двох типів:

1. Синхронізовані (synced) passkeys. Зберігаються на пристрої користувача та синхронізуються між його пристроями через захищені хмарні сервіси. Це дозволяє користувачу використовувати один passkey на різних пристроях, забезпечуючи зручність та безпеку [8].

2. Вбудовані (device-bound) passkeys. Зберігаються локально на конкретному пристрої та не синхронізуються з іншими пристроями. Вони забезпечують високий рівень безпеки, але обмежують використання лише на одному пристрої [9].

У контексті безпарольної автентифікації використовуються два типи автентикаторів:

1. Платформні автентикатори (platform authenticators). Вбудовані в пристрій користувача, такі як сканери відбитків пальців або розпізнавання обличчя на смартфонах чи ноутбуках [10]. Вони забезпечують зручну автентифікацію без додаткових пристроїв, але обмежені конкретною платформою.

2. Роумінгові автентикатори (roaming authenticators): зовнішні пристрої, такі як апаратні токени (наприклад, YubiKey), які можуть підключатися до різних пристроїв через USB, NFC або Bluetooth. Вони забезпечують високий рівень безпеки та універсальність, дозволяючи користувачу автентифікуватися на різних пристроях та платформах [11].

Приватний ключ є таємним і ніколи не покидає пристрій користувача, що знижує ймовірність його компрометації. Наприклад, у системах на основі WebAuthn або FIDO2 цей ключ зберігається або в апаратному токені (наприклад, YubiKey), або в захищеній області пристрою, яка відповідає за зберігання ключів і інші конфіденційні дані (наприклад, Trusted Platform Module — TPM на комп’ютері або Secure Enclave на смартфонах) [12].

Публічний ключ передається серверу під час реєстрації користувача. Коли користувач ініціює автентифікацію, сервер генерує випадковий набір даних, який відправляється назад на пристрій користувача. Пристрій підписує цей набір даних приватним ключем і відправляє підпис назад на сервер. Сервер, використовуючи збережений публічний ключ, перевіряє підпис і, якщо він є коректним, надає доступ користувачу.

Такий підхід забезпечує надійний захист, оскільки навіть у випадку компрометації сервера, зловмисники не зможуть отримати доступ до приватних ключів користувачів. Це робить системи безпарольної автентифікації стійкими до

багатьох атак, таких як фішинг або атаки типу "людина посередині" (Man-in-the-Middle) [13].

Крім того, в таких системах для автентифікації можуть використовуватись біометричні дані (відбитки пальців, розпізнавання обличчя тощо). Однак важливо зазначити, що ці дані також зберігаються лише на пристрої користувача і ніколи не передаються на сервер, що забезпечує додатковий рівень захисту [14].

Переваги асиметричної криптографії в безпарольній автентифікації включають:

- захист від фішингу. Оскільки автентифікація базується на підписанні випадкових даних приватним ключем, зловмисник не може підробити підпис навіть за наявності публічного ключа;
- захист від крадіжки паролів. Оскільки в процесі автентифікації не використовуються паролі, їх не можна перехопити або вкрасти;
- покращена безпека. Навіть якщо сервер зламаний, зловмисник не отримає доступ до приватних ключів користувачів, оскільки вони залишаються на пристрої користувача.

Ця архітектура робить асиметричну криптографію ідеальним рішенням для безпарольної автентифікації, яка дозволяє підвищити рівень безпеки та зручності для користувачів у сучасних інформаційних системах.

У системах безпарольної автентифікації також використовується метод "магічних посилань" (Magic Links). Цей підхід передбачає, що після введення користувачем своєї електронної адреси або імені користувача на порталі входу, система надсилає на вказану адресу одноразове посилання [15]. Користувач, перейшовши за цим посиланням, автоматично входить до свого облікового запису без необхідності введення пароля. Такий метод спрощує процес автентифікації, усуваючи потребу в запам'ятовуванні паролів, і може використовуватися як основний спосіб входу або як додатковий фактор автентифікації [16]. Однак, варто зазначити, що безпека цього методу залежить від надійності електронної пошти користувача, оскільки компрометація поштової скриньки може призвести до несанкціонованого доступу. Розглянути порівняння між методами passkey і magic link можна в таблиці 2.1.

Таблиця 2.1 - Порівняння методів безпарольної автентифікації

Параметр	Magic Links	Passkeys
Механізм автентифікації	Одноразове посилання, надіслане на електронну пошту або через SMS	Асиметрична криптографія з використанням пари приватного та публічного ключів
Зручність для користувача	Висока, не потребує запам'ятовування паролів, але залежить від доступу до електронної пошти або телефону	Висока, особливо при використанні біометричних даних; не потребує доступу до додаткових пристроїв
Безпека	Середня. Залежить від безпеки електронної пошти або телефону; можливі атаки типу "людина посередині"	Висока. Захищена від фішингу та атак типу "людина посередині" завдяки використанню криптографії
Використання біометрії	Немає	Можливе. Підтримує відбитки пальців, розпізнавання обличчя тощо
Сумісність	Широка. не потребує спеціального обладнання	Потребує підтримки з боку пристроїв та сервісів; поступово впроваджується
Термін дії	Обмежений. Посилання зазвичай мають короткий термін дії	Постійний. Ключі зберігаються на пристрої та можуть використовуватися багаторазово

Хоча магічні посилання пропонують зручний спосіб автентифікації без паролів, вони мають певні обмеження щодо безпеки та залежності від доступу до електронної пошти або телефону. Passkeys, завдяки використанню асиметричної криптографії та можливості інтеграції з біометричними методами, забезпечують вищий рівень безпеки та зручності для користувачів. Вони стійкі до фішингових атак і не потребують доступу до додаткових пристроїв або сервісів під час автентифікації. Крім того, passkeys можуть синхронізуватися між різними пристроями користувача, що підвищує зручність використання. Тому, для розробки сучасних систем безпарольної автентифікації, passkeys є більш надійним та перспективним вибором.

2.2 Алгоритми та технології для реалізації безпарольної автентифікації

Одним із найбільш поширених стандартів для реалізації безпарольної автентифікації є FIDO2, який включає технології WebAuthn та CTAP (Client to Authenticator Protocol). WebAuthn забезпечує автентифікацію через веб-додатки та браузери за допомогою асиметричної криптографії, тоді як CTAP дозволяє використовувати апаратні токени або біометричні засоби для підтвердження особи користувача.

FIDO2 підтримує використання як апаратних токенів (наприклад, YubiKey), так і біометричних методів (відбитки пальців, розпізнавання обличчя), що значно підвищує зручність для користувачів і забезпечує високий рівень безпеки. Важливо зазначити, що процес автентифікації за допомогою цих методів захищений від фішингових атак, оскільки домен, до якого здійснюється вхід, зберігається на токені або пристрої, а не на сервері.

Крім того, FIDO2 забезпечує масштабованість та сумісність з різними платформами, що дозволяє легко інтегрувати його в існуючі системи та додатки. Це робить FIDO2 привабливим рішенням для організацій будь-якого розміру, які

прагнуть підвищити рівень безпеки та зручності автентифікації для своїх користувачів.

2.3 Програмні засоби для розробки систем безпарольної автентифікації

Для розробки таких систем використовуються різноманітні програмні засоби, які дозволяють інтегрувати безпарольну автентифікацію в сучасні інформаційні системи.

2.3.1 Мова програмування Java

Мова програмування Java була обрана для розробки нашого проєкту завдяки своїм численним перевагам, які роблять її ідеальним вибором для створення надійних та масштабованих застосунків. Java — це високорівнева об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems у 1995 році [17]. Однією з ключових особливостей Java є її кросплатформеність, що дозволяє запускати програми на різних операційних системах без необхідності внесення змін у код. Це досягається завдяки використанню віртуальної машини Java (JVM), яка інтерпретує байт-код незалежно від платформи.

Java була обрана для розробки завдяки кільком ключовим перевагам:

- платформна незалежність. Завдяки JVM, Java-код може виконуватися на будь-якій операційній системі без необхідності модифікацій;
- об'єктно-орієнтована парадигма. Java підтримує об'єктно-орієнтоване програмування, що сприяє модульності, повторному використанню коду та легшому обслуговуванню програм;
- безпека. Java має вбудовані механізми безпеки, які захищають від багатьох поширених вразливостей, таких як переповнення буфера;

- велика кількість інтеграцій з іншими інструментами. Java має багатий набір бібліотек, фреймворків та інструментів, що прискорюють процес розробки.

Порівняно з іншими мовами програмування, такими як C++ та Python, Java має свої унікальні переваги. Наприклад, на відміну від C++, Java забезпечує автоматичне управління пам'яттю через механізм збору сміття (Garbage Collection), що зменшує ризик витоків пам'яті та підвищує стабільність застосунків [18]. Хоча Python відомий своїм простим синтаксисом, Java пропонує строгішу систему типізації, що допомагає виявляти помилки на етапі компіляції та забезпечує більшу надійність коду.

Для кращого розуміння позиції Java серед інших мов програмування, розглянемо порівняльну таблицю 2.2.

Таблиця 2.2 – Порівняння Java з іншими мовами програмування

Характеристика	Java	C++	Python
Парадигма	Об'єктно-орієнтована	Об'єктно-орієнтована, процедурна	Об'єктно-орієнтована, процедурна
Платформна незалежність	Висока (через JVM)	Низька (залежить від компілятора)	Висока (інтерпретована мова)
Управління пам'яттю	Автоматичне (збірка сміття)	Ручне	Автоматичне (збірка сміття)
Швидкість виконання	Висока	Дуже висока	Середня
Складність синтаксису	Середня	Висока	Низька
Основні сфери застосування	Веб-розробка, мобільні додатки, корпоративні системи	Системне програмування, ігри, драйвери	Веб-розробка, наука, машинне навчання

Java також відзначається високою продуктивністю завдяки компіляції в байт-код та використанню JIT-компіляції (Just-In-Time), що дозволяє оптимізувати виконання програм під час їх роботи [19]. Хоча Java може бути повільнішою за C++ у деяких обчислювально інтенсивних задачах, її продуктивність є достатньою для більшості корпоративних застосунків. В роботі Java була використана як основний інструмент написання програмного коду, який потім легко можна було інтегрувати з такими фреймворками, як Spring та Hibernate.

2.3.2 Фреймворк Spring Boot

Spring Boot – це фреймворк для мікросервісної архітектури, який базується на Spring Framework, але значно спрощує його використання завдяки принципам конфігурації за замовчуванням і автоматичної конфігурації. Spring Boot був створений для прискорення розробки та зменшення складності створення сучасних Java-додатків. Головна мета Spring Boot – забезпечити готову основу для запуску проєктів без необхідності налаштовувати багато конфігурацій вручну [20]. Це робить його надзвичайно популярним серед розробників, які хочуть швидко створювати додатки, зокрема для веб, хмарних платформ і мікросервісів [21].

Однією з ключових переваг Spring Boot є його можливість інтеграції з іншими популярними технологіями. Наприклад, фреймворк підтримує більшість реляційних баз даних, таких як PostgreSQL, MySQL, H2, та нереляційних баз, таких як MongoDB та Redis. Spring Boot використовує Spring Data для роботи з базами даних, що дозволяє зосередитися на бізнес-логіці замість написання складного SQL-коду. Крім того, його підтримка RESTful API, інтеграція з Hibernate, а також модулі для хмарних обчислень роблять його надзвичайно гнучким [22].

Однією з основних особливостей Spring Boot є його інтеграція з Spring Initializr. Цей онлайн-інструмент дозволяє створити базовий проєкт за кілька

хвилин, вибравши необхідні залежності та структуру. Spring Boot також має вбудований вебсервер (Tomcat або Jetty), що дозволяє запускати додаток безпосередньо зі звичайного JAR-файлу. Це суттєво спрощує розгортання і тестування додатків у локальних або хмарних середовищах.

Spring Security є основним модулем безпеки у Spring Boot, який надає засоби для реалізації автентифікації та авторизації. Він підтримує інтеграцію з FIDO2 та WebAuthn, дозволяючи налаштовувати безпарольну автентифікацію користувачів через криптографічні ключі або апаратні токени. Завдяки Spring Security розробники можуть реалізовувати багатофакторну автентифікацію з використанням токенів, таких як YubiKey, або біометричних даних, що підвищує рівень безпеки додатків. Фреймворк дозволяє налаштувати різні варіанти політик безпеки, що є важливим при впровадженні складних систем автентифікації [23].

У дипломній роботі Spring Boot використовувався для спрощення та прискорення розробки, надаючи автоматичну конфігурацію та вбудований сервер, що дозволило швидко створювати та розгортати застосунок.

Spring Security інтегрувався для забезпечення безпеки, реалізуючи механізми аутентифікації та авторизації, що гарантувало захист даних і контроль доступу користувачів.

2.3.3 Бібліотека Yubico

Yubico надає бібліотеки та SDK для роботи з апаратними токенами, такими як YubiKey, що є одним із ключових інструментів для реалізації безпарольної автентифікації. YubiKey – це апаратний пристрій, який зберігає приватні криптографічні ключі і використовується для автентифікації користувачів за допомогою криптографії з відкритими ключами. Бібліотека Yubico дозволяє легко інтегрувати підтримку цих токенів у додатки на Java, спрощуючи процес розробки та налаштування криптографічних операцій. Крім того, бібліотеки Yubico підтримують багатофакторну автентифікацію (2FA), де використовується фізичний пристрій для підтвердження входу.

Бібліотеки Yubico знаходять широке застосування в корпоративних рішеннях для захисту даних. Зокрема, вони дозволяють реалізувати сильну автентифікацію для доступу до внутрішніх систем, електронної пошти або VPN. Це особливо корисно для компаній, які працюють з конфіденційною інформацією і прагнуть зменшити ризики витоку даних [24].

В стартап-проєкті бібліотека Yubico, яка написана на мові Java, дозволила реалізувати основну логіку безпарольної автентифікації, що включає роботу з криптографічними ключами.

2.3.4 Фреймворк Hibernate

Hibernate – це популярний фреймворк для об’єктно-реляційного відображення (Object-Relational Mapping, ORM) у Java, який допомагає розробникам працювати з базами даних, використовуючи об’єктно-орієнтований підхід [25]. Його головна мета – спростити розробку додатків, які працюють з великими обсягами даних [26].

Основні особливості Hibernate включають такі:

- автоматичне ORM. Hibernate дозволяє створювати Java-об'єкти, які автоматично відображаються у таблиці бази даних. Використовуючи анотації або XML-файли, розробники можуть легко налаштувати, як об'єкти співвідносяться зі структурами бази даних;
- HQL (Hibernate Query Language). Hibernate підтримує власну мову запитів, яка схожа на SQL, але дозволяє працювати з об'єктами замість таблиць. Це робить запити більш інтуїтивними для Java-розробників;
- кешування. Hibernate підтримує двохрівневе кешування, що допомагає зменшити кількість запитів до бази даних і підвищує продуктивність програми. Перший рівень кешу є локальним для сесії, а другий — глобальним і може використовувати сторонні рішення, такі як Ehcache або Infinispan;

- підтримка багатьох баз даних. Hibernate сумісний з більшістю популярних реляційних баз даних, таких як PostgreSQL, MySQL, Oracle, та SQL Server, а також легко інтегрується з іншими бібліотеками, такими як Spring Data;
- життєвий цикл об'єктів. Hibernate забезпечує управління життєвим циклом об'єктів через їхні стани: Transient, Persistent і Detached. Це дозволяє гнучко керувати даними, забезпечуючи правильне оновлення та видалення об'єктів у базі даних.

Hibernate має кілька важливих переваг, які роблять його зручним для розробників. По-перше, він значно зменшує кількість SQL-запитів, які потрібно писати вручну, завдяки автоматичному відображенню об'єктів у таблиці бази даних. По-друге, Hibernate забезпечує портативність, дозволяючи легко змінювати базу даних, оскільки його HQL-запити не залежать від конкретного SQL-діалекту. Також фреймворк автоматично управляє транзакціями, забезпечуючи їх синхронізацію і можливість відкату у разі помилок. Нарешті, Hibernate легко інтегрується з іншими популярними інструментами, такими як Spring, що спрощує розробку складних корпоративних систем.

У проєкті фреймворк Hibernate використовувався для об'єктно-реляційного відображення (ORM), що дозволило взаємодіяти з базою даних, відображаючи об'єкти Java на таблиці бази даних і навпаки. Це спростило управління транзакціями та виконання запитів, забезпечуючи об'єктно-орієнтований підхід до роботи з даними.

2.3.5 Збірник проекту Maven

Maven є популярним інструментом для управління залежностями та автоматизації процесу збірки проектів на Java. Він забезпечує ефективне управління бібліотеками, такими як Spring Security, Yubico SDK та Hibernate, що дозволяє розробникам легко інтегрувати їх у свої проекти. Використання Maven спрощує процес налаштування та підтримки систем безпарольної автентифікації,

дозволяючи автоматично керувати оновленням залежностей та конфігурацією проекту [27].

Maven є корисний завдяки об'єктній моделі проекту (Project Object Model, POM), яка є XML-файлом, що містить всю інформацію про проект і деталі конфігурації. POM містить опис проекту, деталі щодо керування версіями та конфігурацією проекту. XML-файл знаходиться у домашньому каталозі проекту. Коли користувач виконує завдання, Maven шукає POM у поточному каталозі [28].

Maven в основному використовується для проектів на основі Java, допомагаючи завантажувати залежності, які відносяться до бібліотек або JAR-файлів. Інструмент допомагає отримати правильні JAR-файли для кожного проекту, оскільки можуть існувати різні версії окремих пакунків. Після Maven завантаження залежностей не потребує відвідування офіційних сайтів різних програм. Інструмент також допомагає створити правильну структуру проекту у вигляді стритів, сервлетів тощо, що є важливим для виконання.

Maven пропонує численні переваги, зокрема автоматизацію таких процесів, як створення, документування, випуск та розповсюдження, що спрощує управління проектами та підвищує їхню продуктивність. Він автоматично обробляє завантаження JAR-файлів і залежностей, забезпечує легкий доступ до необхідних ресурсів і дозволяє розробникам безперешкодно створювати проекти в різних середовищах, не турбуючись про залежності або процеси. Додавання нових залежностей є простим і вимагає лише простого запису у файлі pom.xml.

Однак Maven має і недоліки: він вимагає встановлення та плагіну IDE, не може додавати залежності без наявного коду Maven, і іноді його критикують за повільну роботу.

У роботі Maven використовувався як система керування залежностями та автоматизації процесу збирання. Він спрощував інтеграцію зовнішніх бібліотек і фреймворків, забезпечуючи стандартизовану структуру проєкту та зручне керування версіями залежностей.

2.3.6 База даних PostgreSQL

PostgreSQL — це потужна, масштабована система управління базами даних з відкритим кодом, яка поєднує в собі багатий функціонал, надійність та високий рівень безпеки. Завдяки підтримці стандарту ACID (Atomicity, Consistency, Isolation, Durability), PostgreSQL гарантує цілісність даних і є одним із найкращих виборів для проєктів, де важлива стабільність і точність операцій з базами даних. Її підтримка складних структур даних, таких як JSON, XML і геопросторові дані, робить PostgreSQL особливо привабливою для сучасних програм з різноманітними типами інформації [29].

В контексті систем безпарольної автентифікації PostgreSQL використовується для збереження публічних криптографічних ключів користувачів, пов'язаних метаданих і іншої інформації, необхідної для автентифікації. Це дозволяє легко масштабувати систему, додавати нових користувачів і швидко перевіряти їхні дані при автентифікації. Потужні засоби індексації PostgreSQL, зокрема GiST і GIN, забезпечують швидкий доступ до даних навіть при великому обсязі записів у базі [30].

Завдяки вбудованим засобам шифрування, таким як TLS/SSL, PostgreSQL дозволяє захищати дані на етапах передачі, що важливо для забезпечення конфіденційності та запобігання перехопленням. Крім того, вона підтримує розширені механізми контролю доступу, включаючи рольову модель та гранулярні привілеї на рівні таблиць, рядків і колонок, що додає ще один рівень захисту від несанкціонованого доступу.

Гнучкість PostgreSQL дозволяє легко інтегрувати її в різні середовища, включаючи хмарні платформи та корпоративні системи. Це досягається завдяки підтримці розширень, які розширюють її можливості, наприклад, для роботи з реплікацією, резервним копіюванням, або розподіленими базами даних. Завдяки цьому PostgreSQL може використовуватися як у невеликих системах, так і в масштабних розподілених рішеннях, що обслуговують мільйони користувачів [31].

У дипломній роботі PostgreSQL використовувалася як основна система керування базами даних для зберігання та управління інформацією. Інтеграція PostgreSQL з Java через фреймворки Spring Boot та Hibernate забезпечила взаємодію між застосунком та базою даних, спрощуючи операції створення, читання, оновлення та видалення (CRUD). Це дозволило більше зосередитися на бізнес-логіці, мінімізуючи необхідність написання складних SQL-запитів.

У розділі було розглянуто основні методи та засоби, які є основою для створення сучасних систем безпарольної автентифікації. Системи на основі асиметричної криптографії забезпечують високий рівень безпеки та зручності для користувачів, усуваючи необхідність використання паролів. Впровадження таких рішень базується на ефективному використанні стандартів, таких як FIDO2, WebAuthn, і CTAP, а також інтеграції сучасних апаратних і програмних засобів, включаючи токени YubiKey та біометричні автентикатори.

Крім того, були розглянуті інструменти розробки, такі як Java, Spring Boot, Hibernate, і Maven, які дозволяють створити стійку та продуктивну платформу з можливістю швидко масштабувати.

3 РЕАЛІЗАЦІЯ СИСТЕМИ БЕЗПАРОЛЬНОЇ АВТЕНТИФІКАЦІЇ

Цей розділ містить деталізовану інформацію про процес розробки сервісу на основі WebAuthn, що включає аналіз вимог до програмного забезпечення, архітектуру сервісу, реалізацію схеми БД, інструменти розробки, тестування та налаштування сервісу, вимоги до технічного обладнання та забезпечення безпеки сервісу.

3.1 Аналіз вимог до програмного забезпечення

Сервіс повинен включати такий функціонал:

- реєстрація користувачів через WebAuthn;
- автентифікація через WebAuthn;
- підтримка апаратних токенів (YubiKey, Google Titan Security Key, Kensington VeriMark та інші), які використовуються для підтвердження особи користувача та збереження приватних ключів;
- можливість використання біометричних методів автентифікації (відбитки пальців, розпізнавання обличчя);
- генерація та обробка автентифікаційних токенів для керування сесіями користувачів після успішної автентифікації;
- підтримка перевірки автентичності користувача на основі отриманих даних (публічний ключ і підпис).

Функціонал автентифікованого користувача:

- оновлювати свої дані профілю;
- керування ключами автентифікації;
- вихід з системи та видалення кукі сесії після цього.

3.2 Архітектура програмного забезпечення

Архітектура програмного забезпечення визначає структуру та спосіб організації елементів системи. Вона включає в себе деталізований опис компонентів, їхню взаємодію, а також принципи і правила, які керують процесом їхнього проєктування та подальшої еволюції. Основою архітектури є поділ на функціональні блоки з визначеними ролями, що забезпечує модульність, масштабованість та можливість повторного використання компонентів.

3.2.1 Моделі та взаємодія компонентів

Моделі та взаємодія компонентів є важливою частиною архітектури програмного забезпечення. Вони допомагають розуміти, як складові програмного забезпечення взаємоді.

В архітектурі програмного забезпечення для системи безпарольної автентифікації виділяються кілька ключових компонентів, кожен з яких виконує певні функції та взаємодіє з іншими модулями через чітко визначені інтерфейси. Основні компоненти, що взаємодіють між собою – користувач, клієнт (браузер), веб-сервер та автентифікатор.

Користувач – особа, яка ініціює процес автентифікації для доступу до захищених ресурсів системи [32]. Користувач може вводити облікові дані або використовувати апаратний токен для підтвердження своєї особи. У безпарольній автентифікації користувач виконує ключову роль, адже підтверджує доступ за допомогою фізичних або біометричних факторів.

Клієнт/браузер – інструмент для взаємодії користувача з вебсистемою. Браузер надсилає запити до сервера автентифікації, отримує автентифікаційні виклики та відправляє відповіді автентифікатора на сервер [33]. Браузер підтримує WebAuthn, що дозволяє працювати з апаратними токенами, як-от YubiKey, і підтримує безпечну передачу криптографічних даних, необхідну для сучасних методів автентифікації.

Вебсервер – сервер, який приймає та обробляє запити від клієнта, керує процесом автентифікації, працює з базою даних для перевірки користувацьких облікових даних і валідності токенів [34]. Вебсервер підтримує інтеграцію з автентифікатором для перевірки автентифікаційних даних за стандартами FIDO2/WebAuthn.

Автентифікатор – апаратний або програмний компонент, який генерує та зберігає криптографічні ключі для автентифікації. Апаратні токени, такі як YubiKey або Google Titan Security Key, використовуються для підвищення рівня безпеки та забезпечення можливості автентифікації без паролів [35]. Автентифікатор створює підпис на основі приватного ключа для підтвердження особи користувача та передає його серверу через браузер [36].

На рисунку 3.1 зображено загальну архітектуру та основні зв’язки між компонентами WebAuthn системи.

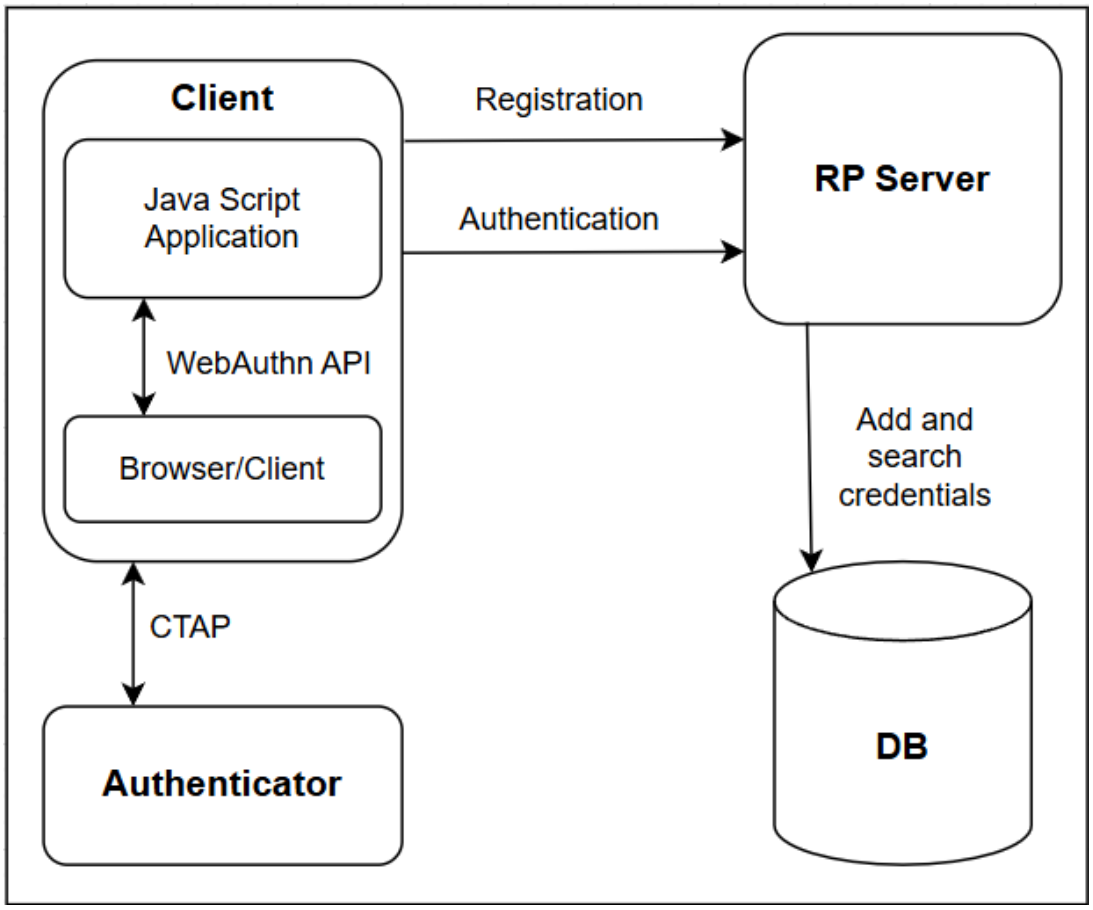


Рисунок 3.1 – Загальна архітектура взаємодії компонентів у системі WebAuthn

Діаграма на рисунку 3.1 включає наступні зв’язки:

- клієнт (браузер) ініціює запити на реєстрацію чи автентифікацію через WebAuthn API, взаємодіючи із JavaScript-додатком;
- автентифікатор взаємодіє з браузером через протокол CTAP, забезпечуючи захищені криптографічні операції;
- вебсервер відповідає за обробку запитів і передачу даних до бази даних, де зберігаються облікові записи, публічні ключі та інша інформація.

На рисунку 3.2 зображена діаграма послідовностей взаємодії даних компонентів.

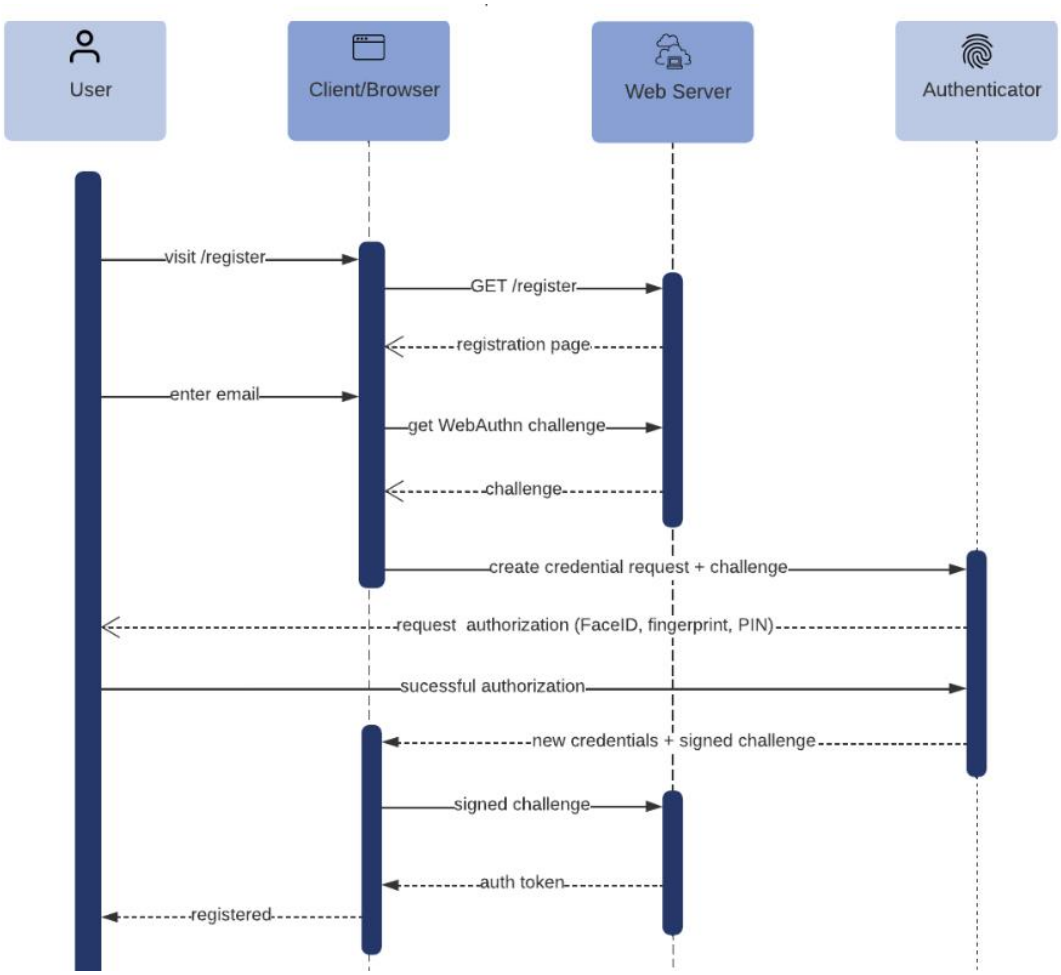


Рисунок 3.2 – Діаграма послідовностей взаємодії компонентів

Зображена діаграма на рисунку 3.2 включає 8 кроків:

1. Користувач заходить на сторінку реєстрації (visit /register) через клієнт/браузер. Клієнт надсилає GET-запит на сервер для отримання сторінки реєстрації.

2. Користувач вводить свою електронну адресу (enter email), і клієнт надсилає цей запит на сервер.

3. Сервер, отримавши запит від клієнта, генерує WebAuthn виклик (challenge) і надсилає його клієнту/браузеру для подальшої автентифікації.

4. Клієнт створює запит на створення облікових даних (credential request) з отриманим викликом, який передається на автентифікатор (Authenticator). Автентифікатор може бути YubiKey, FaceID, або інший апаратний чи біометричний засіб.

5. Користувач проходить авторизацію за допомогою біометричних даних (відбиток пальця, FaceID) або PIN-коду. Після успішної авторизації автентифікатор підписує виклик приватним ключем.

6. Після успішної автентифікації автентифікатор передає назад клієнту нові облікові дані та підписаний виклик.

7. Клієнт/браузер передає ці нові облікові дані та підписаний виклик на сервер. Сервер перевіряє підпис, і якщо все коректно, генерує автентифікаційний токен (auth_token) для користувача.

8. Користувач отримує підтвердження, що реєстрація пройшла успішно, і він зареєстрований у системі.

Ці компоненти спілкуються між собою за допомогою API, які повинні бути розроблені таким чином, щоб забезпечити безпечну та ефективну взаємодію. Важливим є також розробка панелі адміністрування, що дозволяє керувати системою, налаштовувати параметри безпеки, керувати користувацькими обліковими записами та дозволами доступу.

Визначення правильної архітектури є критично важливим для успіху будь-якого проекту розробки програмного забезпечення. Вона має враховувати не лише поточні вимоги, але й майбутнє розширення та адаптацію системи.

3.2.2 Реалізація серверної частини

Реалізація серверної частини системи безпарольної автентифікації вимагає чіткого структурування коду для подальшого масштабування та підтримки безпеки. У цьому контексті тестову систему було організовано за допомогою пакетів, кожен з яких виконує специфічні функції, що сприяє модульності та спрощує процес розробки і підтримки системи. Розглянемо структуру пакетів серверної частини на рисунку 3.3.

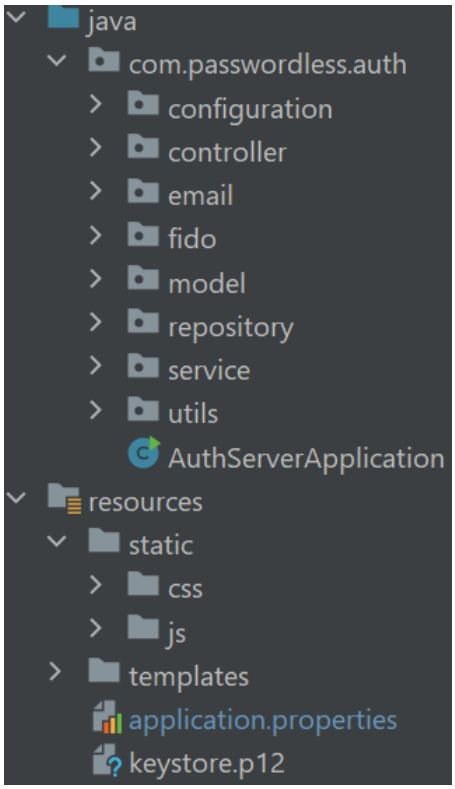


Рисунок 3.3 – Структура пакетів серверної частини

Структура на рисунку 3.3 включає наступні пакети:

- пакет com.passwordless.auth.configuration відповідає за налаштування безпеки та автентифікації в системі, використовуючи Spring Security. Основні компоненти включають WebSecurityConfig, який конфігурує доступ до ресурсів, фільтри для аутентифікації через токени та cookie. Додатково, пакет містить інтеграцію з WebAuthn через конфігурацію довіреної сторони (RelyingPartyConfiguration) для роботи з Yubico;

- пакет com.passwordless.auth.controller реалізує API для управління автентифікацією та реєстрацією користувачів через WebAuthn. Він обробляє HTTP-запити для входу, виходу, реєстрації та відновлення доступу, забезпечуючи інтеграцію з клієнтськими додатками та сервісами автентифікації;
- пакет com.passwordless.auth.email реалізує функціонал для надсилання електронних листів, необхідних для відновлення доступу при втраті пристрою;
- пакет com.passwordless.auth.fido реалізує механізми автентифікації через FIDO2/WebAuthn. Він містить класи для обробки FIDO-токенів, конвертації HTTP-запитів у об’єкти автентифікації, управління процесом перевірки за допомогою FidoAuthenticationManager і обробки успішної автентифікації через FidoLoginSuccessHandler. Цей пакет інтегрує WebAuthn із системою Spring Security, забезпечуючи безпечну автентифікацію без паролів;
- пакет com.passwordless.auth.model містить моделі даних, що використовуються для зберігання інформації в системі. До нього входять класи для представлення ролей, ключів автентифікації, користувачів, токенів доступу, а також запитів на вхід і реєстрацію;
- пакет com.passwordless.auth.repository містить репозиторії для доступу до бази даних, що забезпечують зберігання/управління обліковими записами користувачів та запитами пов’язаними з реєстрацією та автентифікацією;
- пакет com.passwordless.auth.service містить сервіси, які реалізують бізнес-логіку для обробки автентифікації, реєстрації користувачів та управління їх обліковими даними;
- пакет com.passwordless.auth.utils містить утилітарні класи, що забезпечують допоміжну функціональність для роботи з JSON, URL, токенами, обліковими записами користувачів та UUID.

Директорія resources включає дві піддиректорії для реалізації клієнтської частини: static, де зберігаються файли JavaScript і CSS, та templates, призначену для зберігання HTML-файлів.

Для більш детального розуміння взаємодії між компонентами тестової системи, розглянемо діаграму класів на рисунку 3.4, яка відображає їх структуру та зв’язки.

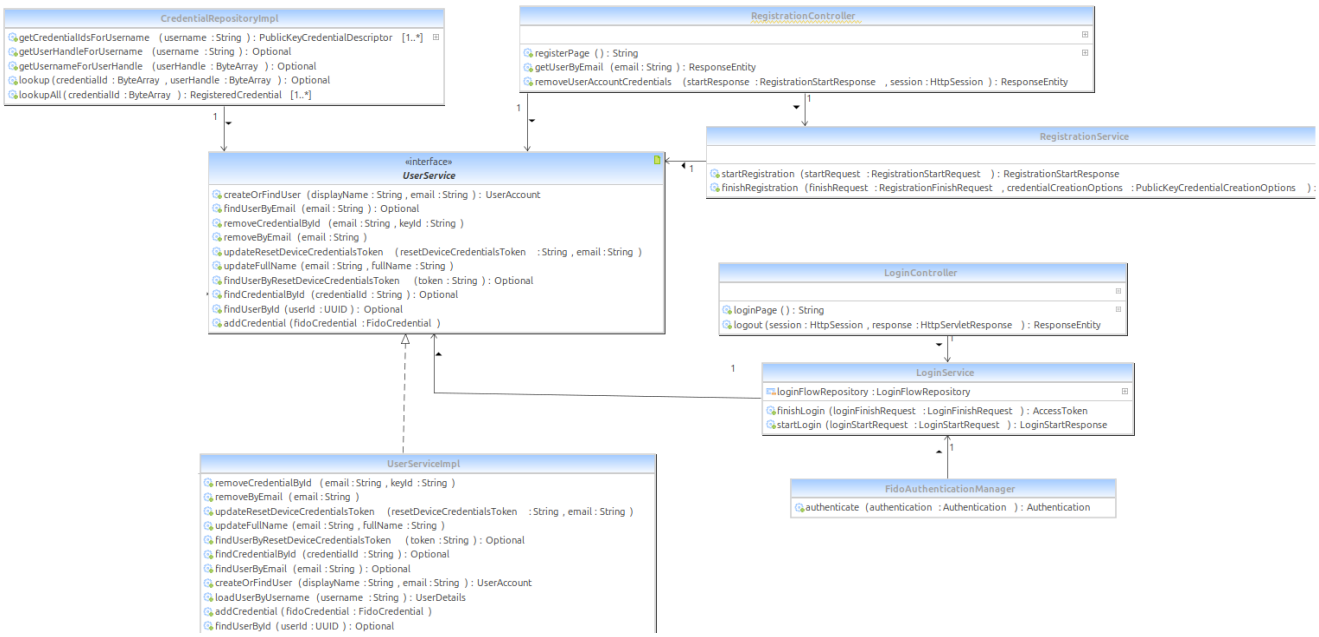


Рисунок 3.4 – Діаграма класів тестової системи

На рисунку 3.4 представлена архітектура основних компонентів системи, що забезпечують підтримку автентифікації без паролів за допомогою протоколу WebAuthn. Вона наочно демонструє зв’язки між контролерами, сервісами та репозиторіями, які спільно формують основу для обробки запитів, виконання бізнес-логіки та доступу до бази даних.

Контролери (LoginController, RegistrationController, UserAccountController) виконують роль точок входу для взаємодії з клієнтами системи. Вони обробляють запити, такі як реєстрація нових користувачів, вхід до системи та управління обліковими записами, передаючи їх до відповідних сервісів.

Сервіси (LoginService, RegistrationService, UserServiceImpl) відповідають за реалізацію основної функціональності. LoginService забезпечує автентифікацію користувачів, включаючи валідацію токенів FIDO2 і генерацію токенів доступу. RegistrationService реалізує процес реєстрації, створюючи публічні ключі автентифікації та асоціюючи їх із користувачами. UserServiceImpl забезпечує роботу з обліковими записами, дозволяючи додавати, оновлювати та видаляти дані користувачів, а також керувати їхніми публічними ключами, що збережені у базі сервера.

Репозиторії (UserAccountRepository, LoginFlowRepository, RegistrationFlowRepository, CredentialRepositoryImpl) відповідають за доступ до бази даних і управління даними. Вони забезпечують ефективне збереження інформації про користувачів та їхні автентифікаційні дані.

3.2.3 Реалізація схеми бази даних

Сервіс передбачає обробку запитів на безпарольну реєстрацію та авторизацію. Для зберігання інформації використовується реляційна база даних PostgreSQL.

Реляційна база даних - це сукупність інформації, яка організовує дані в заздалегідь визначених відносинах, де дані зберігаються в одній або декількох таблицях стовпців і рядків [37].

Для реалізації даного сервісу було створено таблиці бази даних, структура яких показана на рисунку 3.5. На схемі можна побачити 7 таблиць з полями і їх типами даних.

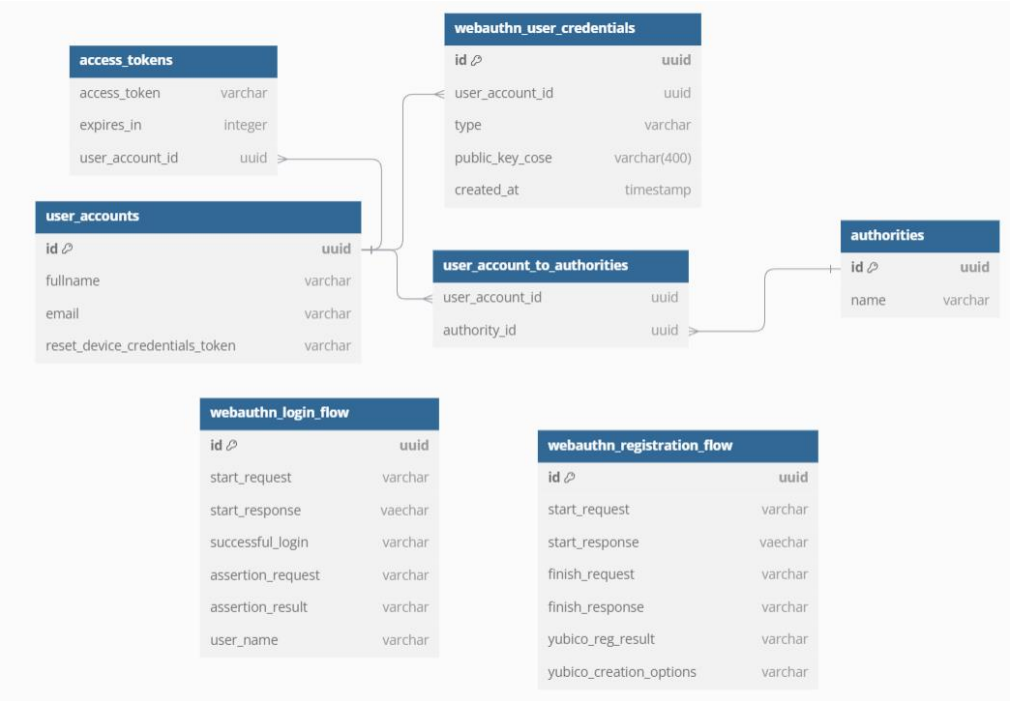


Рисунок 3.5 – Схема бази даних

Таблиця user_accounts зберігає основну інформацію про користувачів, яка є базовою для ідентифікації та автентифікації. У ній містяться такі поля, як унікальний ідентифікатор користувача (user_id), повне ім'я, електронна адреса, а також токен для скидання облікових даних пристрою. Цей токен використовується для відновлення доступу або повторного налаштування автентифікатора, що є важливим для підтримки безпеки та зручності системи.

Таблиця `authorities` містить ролі або права доступу, а таблиця `user_account_to_authorities` з'єднує користувачів з їхніми ролями через зовнішні ключі.

Таблиця webauthn_user_credentials зберігає критично важливі дані для безпарольної автентифікації, такі як публічний ключ, тип облікових даних (наприклад, платформний або роумінговий автентифікатор), а також додаткові метадані, пов'язані з автентифікатором. Завдяки цьому забезпечується унікальність автентифікації кожного користувача, що дозволяє підвищити рівень безпеки системи.

Таблиця webauthn_login_flow веде детальний журнал процесу входу користувача. У ній зберігаються всі запити та відповіді системи під час автентифікації, включаючи час початку та завершення процесу, статус операції та можливі помилки. Це дозволяє адміністраторам системи відстежувати історію входів, аналізувати потенційні збої та забезпечувати контроль за безпекою.

Таблиця `webauthn_registration_flow` зберігає інформацію про процес реєстрації користувачів, включаючи початкові та завершальні запити, а також результати реєстрації через Yubico.

Таблиця `access_tokens` відповідає за збереження токенів доступу, включаючи час дії токена та посилання на користувача, якому він належить.

3.3 Реалізація панелі реєстрації та входу

Перед здійсненням входу в систему користувач повинен зареєструватись на сторінці за шляхом “/auth/webauthn/register”, що зображена на рисунку 3.6.

Додатково потрібно обов’язково заповнити два поля: ім’я та прізвище, пошту.

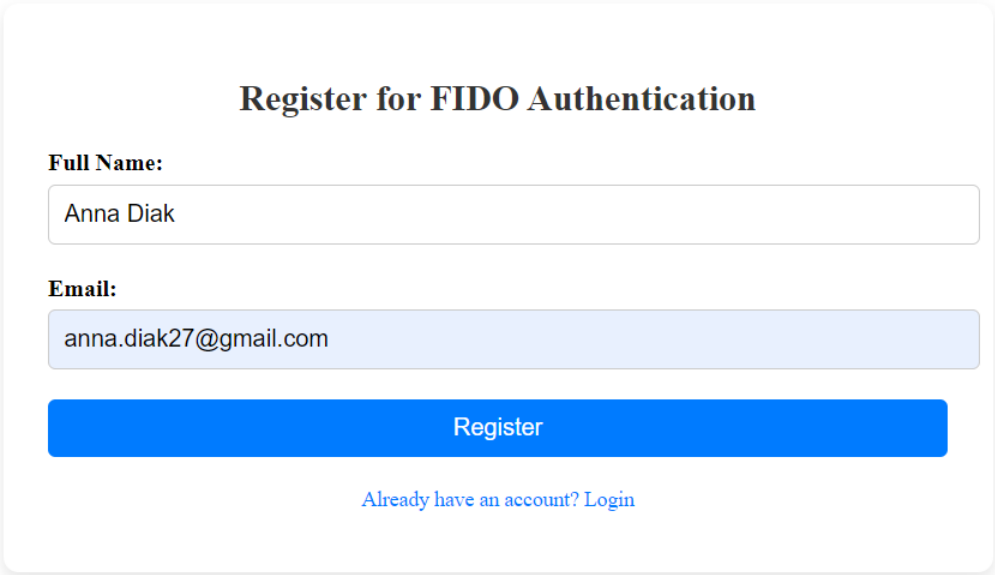


Рисунок 3.6 – Форма реєстрації в систему

Далі, після натискання кнопки “Register”, в користувача з’явиться спливаюче вікно, яке запропонує місце для зберігання ключа (passkey). Загалом існують 3 різні способи:

1. Google Password Manager — дозволяє зберігати ключі в хмарному сховищі Google, що надає можливість входити в систему з різних пристроїв, підключених до одного облікового запису Google. Це менш безпечний варіант порівняно з іншими методами через зберігання ключів у хмарі, але дуже зручний для користувачів, які використовують кілька пристроїв.

2. Windows Hello або зовнішній апаратний ключ — забезпечує зберігання ключа локально на пристрої або на зовнішньому апаратному ключі (наприклад, YubiKey). Цей спосіб вважається значно безпечнішим, оскільки ключ ніколи не передається через мережу і зберігається фізично на пристрої або токені, доступ до якого має лише користувач.

3. Використання іншого телефону, планшету або апаратного ключа — цей метод дозволяє зберігати ключ на іншому пристрої, який може бути використаний

для автентифікації. Користувач може, наприклад, використовувати телефон як засіб автентифікації для входу в систему на комп'ютері. Це забезпечує додаткову гнучкість для тих, хто використовує кілька пристроїв, але хоче зберігати ключ окремо від основного.

Вибір між цими автентифікаторами можна побачити на рисунку 3.7, де користувачу пропонується вибір найзручнішого для нього варіанту зберігання ключів.

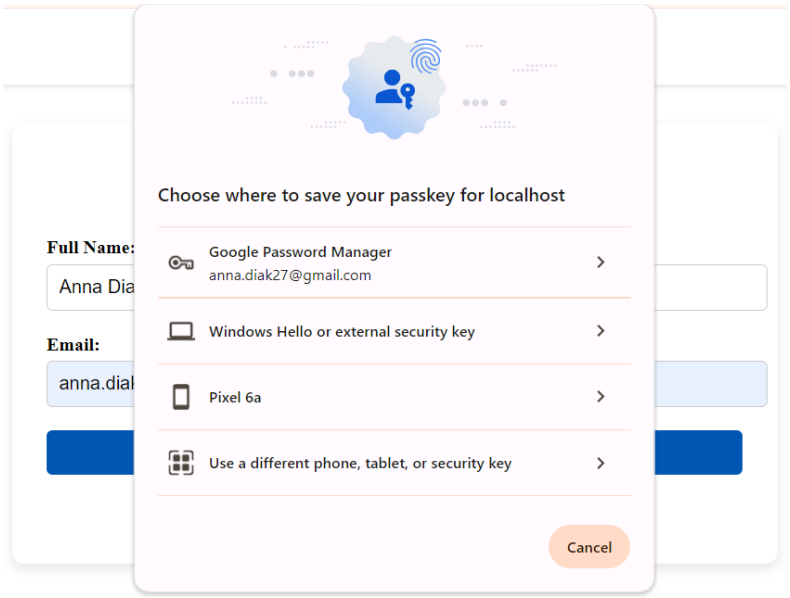


Рисунок 3.7 – Спливаюче вікно з можливістю вибору автентифікатора

Далі в залежності від вибору користувач побачить різне спливаюче вікно.

Якщо користувач вибере Google Password Manager і захоче зберегти ключ у хмарі Google, то з'явиться спливаюче вікно, яке запитає підтвердження входу в обліковий запис Google. Якщо користувач вже авторизований у своєму обліковому записі, система автоматично прив’яже новий ключ до облікового запису.

У цьому ж вікні, яке можна побачити на рисунку 3.8, з'явиться повідомлення, що ключ буде збережено у хмарному сховищі Google і доступний для входу з будь-якого пристрою, підключеного до облікового запису. Для завершення реєстрації користувачу потрібно натиснути "Створити".

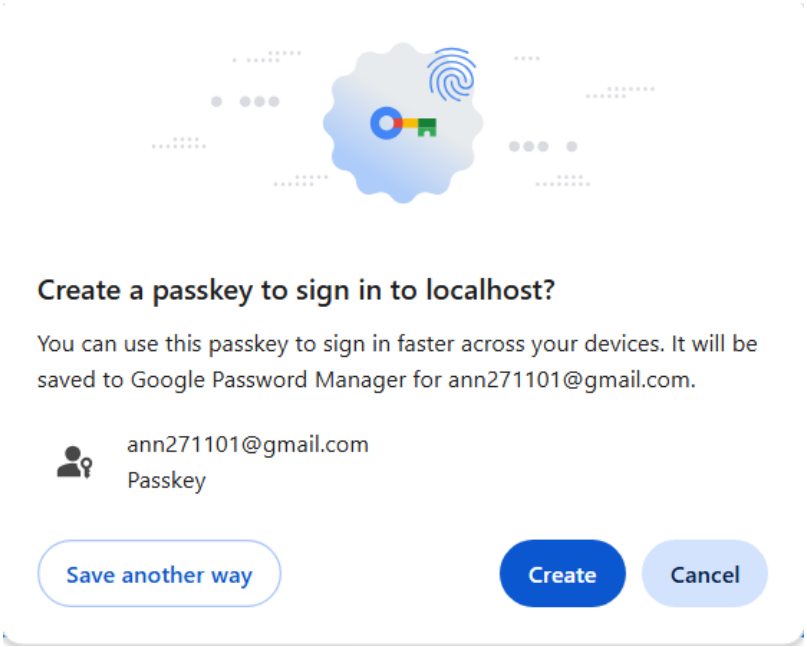


Рисунок 3.8 – Повідомлення про збереження ключа в хмарі Google

Якщо користувач обирає Windows Hello, з'явиться діалогове вікно з біометричною перевіркою або запитом PIN-коду. Для успішної реєстрації ключа користувачеві необхідно підтвердити свою особу через обраний метод сканування відбитка пальця, розпізнавання обличчя або введення PIN. Приклад зображено на рисунку 3.9.

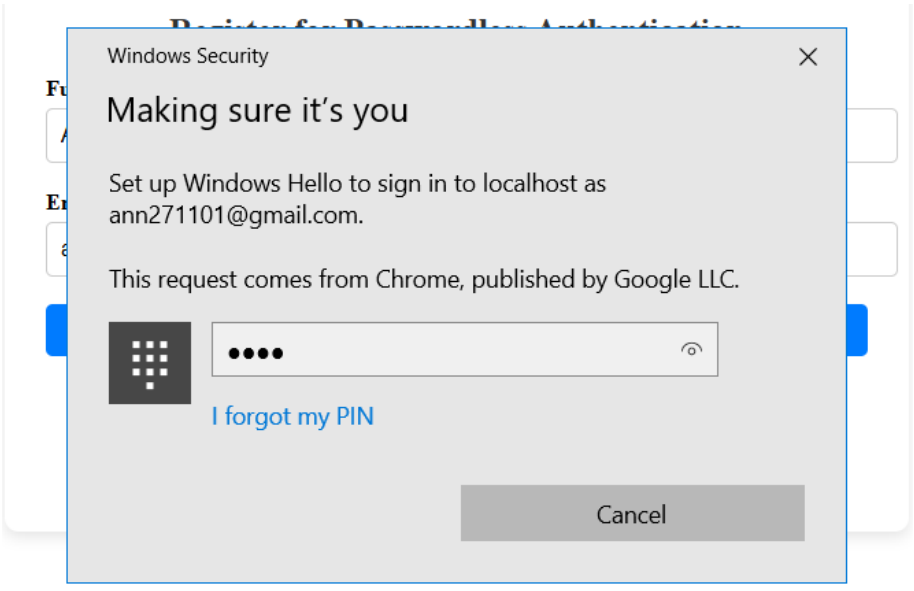


Рисунок 3.9 – Повідомлення про введення PIN у Windows Hello

Якщо обрано зовнішній апаратний ключ, наприклад, YubiKey, з'являється вікно, що зображене на рисунку 3.10, з інструкцією вставити ключ у порт USB або піднести його до пристрою з підтримкою NFC. Це повідомлення допомагає користувачеві виконати необхідні дії для налаштування ключа. У випадку використання NFC-з’єднання користувачу потрібно просто піднести ключ до пристрою, оснащеного NFC-модулем, що робить процес зручним для мобільних пристроїв. Якщо ключ підключається через USB, користувач має вставити його у відповідний порт. Додатково, якщо налаштування апаратного ключа цього вимагають, користувач має торкнутися сенсора на ключі для активації, а також ввести PIN-код або пароль, які додають додатковий рівень безпеки.

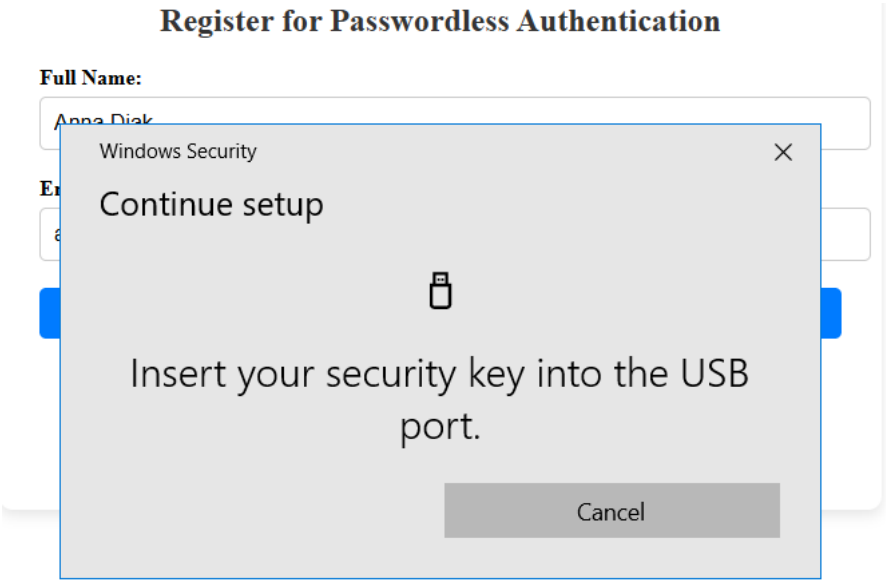


Рисунок 3.10 – Повідомлення про необхідність вставлення ключа в USB порт

Якщо користувач вибере використання іншого пристрою, система запропонує, вікно, зображене на рисунку 3.11, щоб сканувати QR-код за допомогою іншого пристрою (телефону або планшета), на якому буде збережено ключ. QR-код генерується системою для ідентифікації облікового запису користувача та передачі необхідної інформації на інший пристрій. Після сканування QR-коду на додатковому пристрої відкриється вікно з підтвердженням створення ключа, що забезпечує зручність і гнучкість для користувачів, які бажають зберігати свої автентифікаційні ключі на іншому обладнанні.

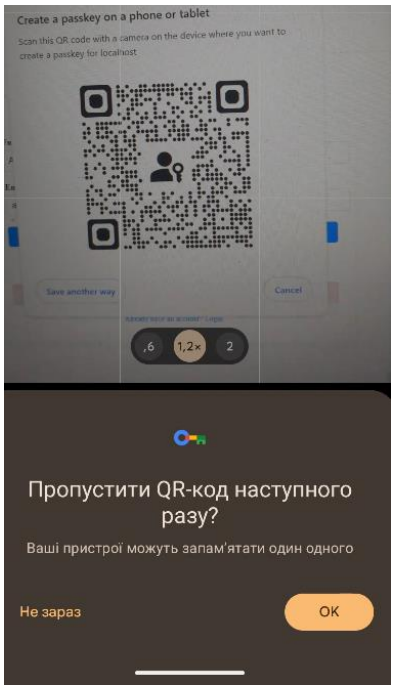


Рисунок 3.11 – Повідомлення про необхідність сканування QR коду

Після сканування QR-коду на іншому пристрої відкриється вікно з підтвердженням створення ключа, яке можна побачити на рисунку 3.12.

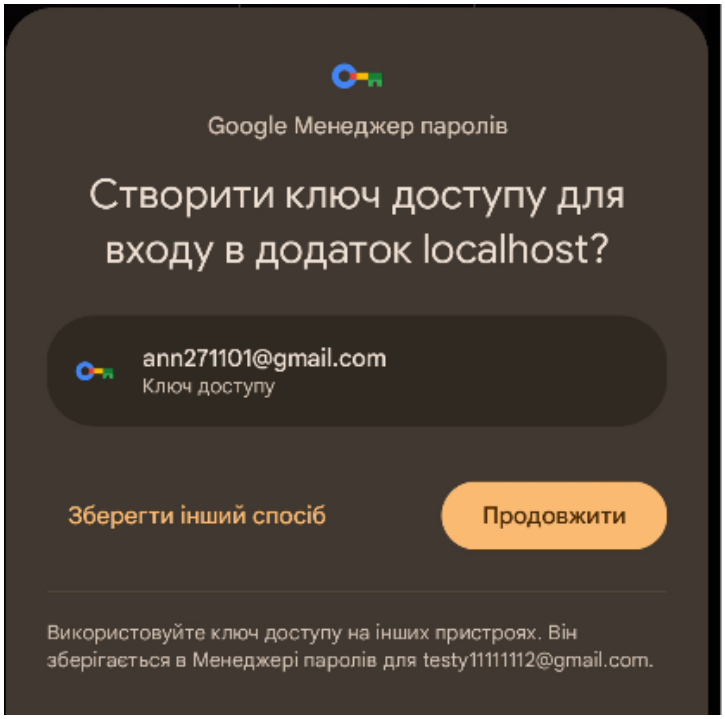


Рисунок 3.12 – Повідомлення про підтвердження створення ключа на пристрої

Коли користувач погодиться на збереження ключа на пристрої, йому також буде запропоновано ввести свої біометричні дані для підтвердження дії.

Після успішної реєстрації в системі будь-яким способом, користувачу буде виведено повідомлення про закінчення реєстрації, що зображено на рисунку 3.13.

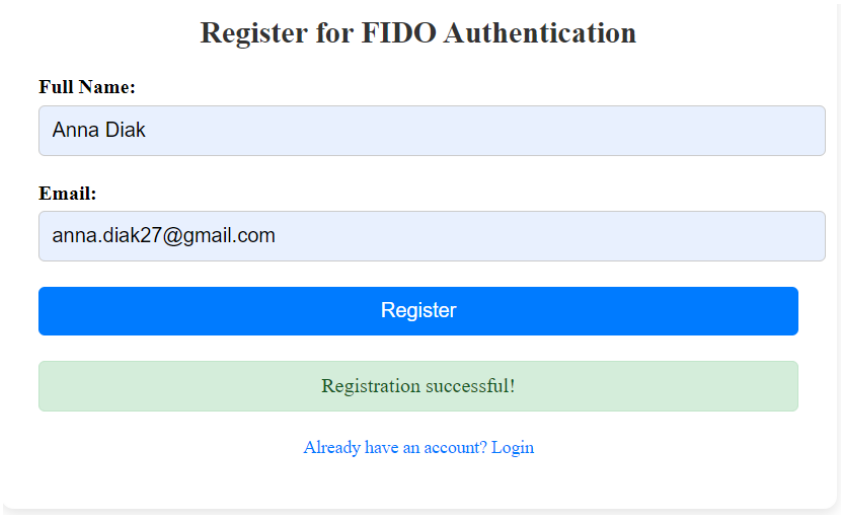


Рисунок 3.13 – Виведення форми після успішної реєстрації

Також слід зазначити, що користувачу для створення ключів відводиться 30 секунд. В разі не вкладання користувача в зазначений проміжок часу або скасування процесу реєстрації, буде виведене відповідне повідомлення про це, що зображено на рисунку 3.14.

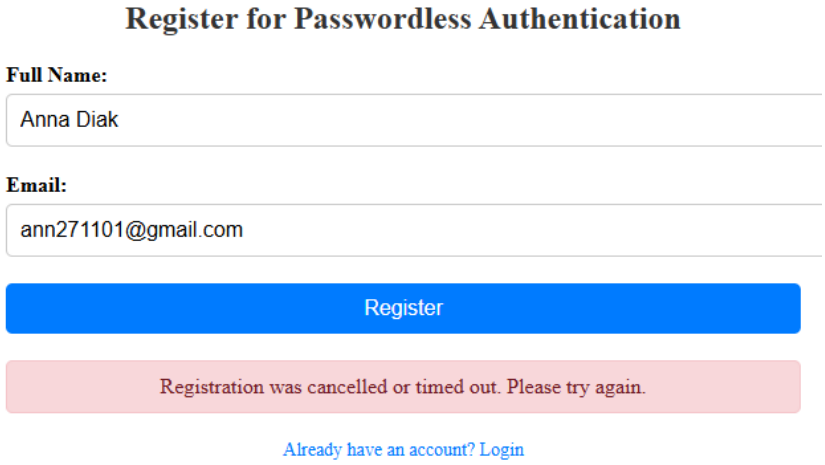


Рисунок 3.14 – Повідомлення про помилку під час процесу реєстрації в системі

Після завершення реєстрації потрібно перейти на сторінку входу, що знаходиться за шляхом “/auth/webauthn/login”. На ній потрібо ввести email та вибрати автентифікатор з якого раніше була здійснена реєстрація. Форму входу можна побачити на рисунку 3.15.

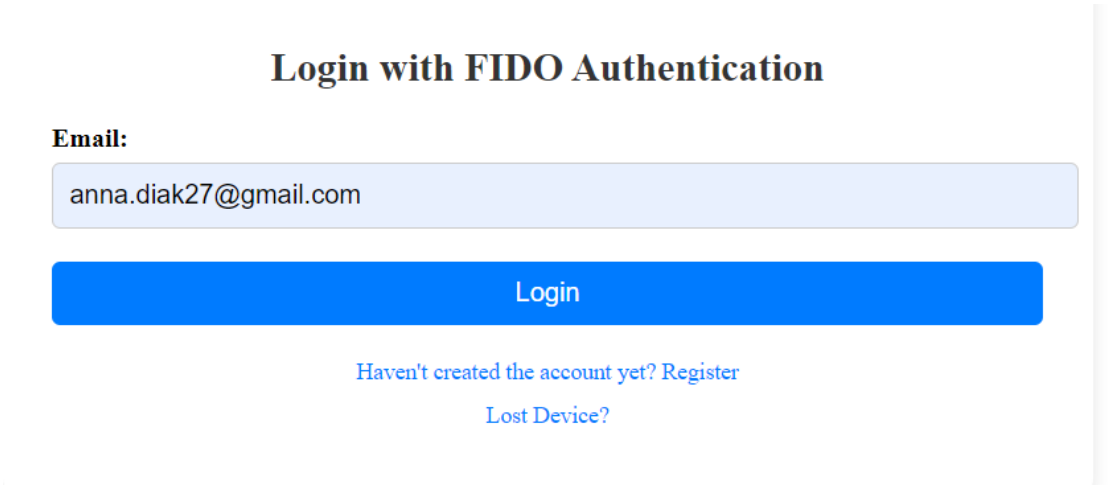


Рисунок 3.15 – Сторінка входу в систему

Коли автентифікація завершиться успішно, користувача буде переадресовано на його особистий обліковий запис.

В разі втрати пристроїв, можна відновити доступ до облікового запису через раніше зареєстровану пошту. Для цього потрібно перейти на сторінку “/auth/webauthn/lost_device”, що зображена на рисунку 3.16 та ввести свою пошту.

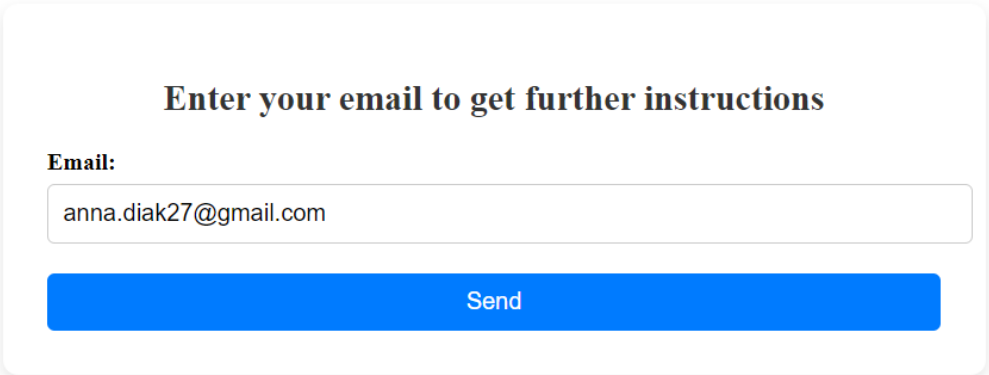


Рисунок 3.16 – Сторінка отримання доступу до облікового запису з нового пристрою через пошту

Після успішного надсилання подальших інструкцій, вони з’являться на раніше введеній пошті. Розглянути інструкції для додавання нових ключів можна на рисунку 3.17.

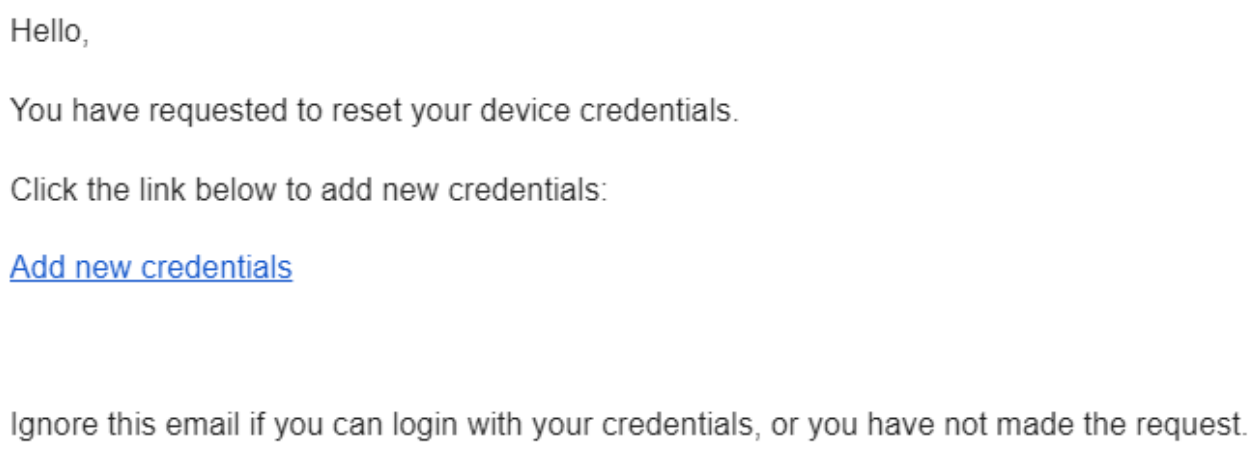


Рисунок 3.17 – Інструкції для отримання доступу для нового пристрою

Після переходу за посиланням ми побачимо форму, що зображена на рисунку 3.18.

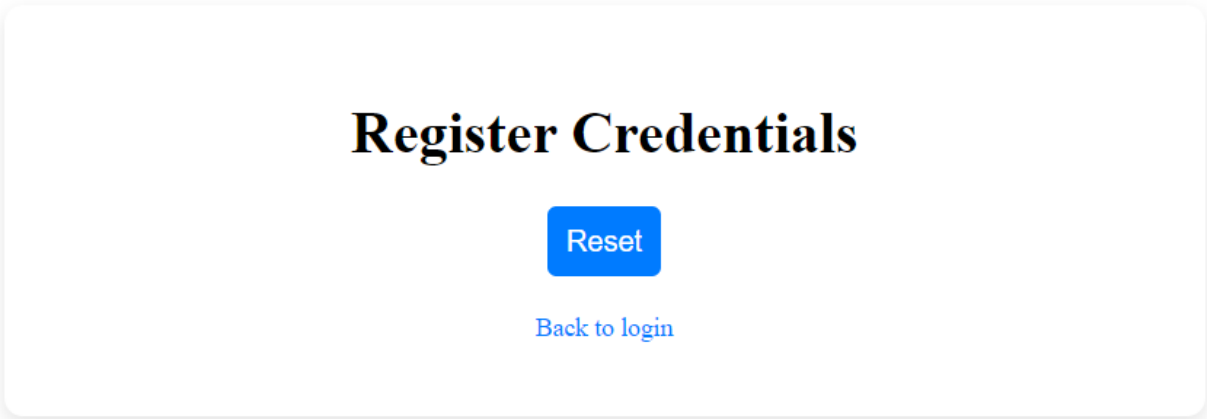


Рисунок 3.18 – Форма для реєстрації нових ключів

Щоб отримати доступ до облікового запису з нового пристрою потрібно натиснути кнопку “Reset”. В результаті буде відображено те саме вікно, що зображено на рисунку 3.7 з можливістю вибору автентифікатора.

На основі рисунків, зображених в даному пункті, які демонструють процеси входу та реєстрації в системі, можна побачити, як клієнтська частина взаємодіє з сервером для виконання операцій автентифікації. Ці процеси забезпечують безпечний обмін даними, включаючи перевірку облікових записів, генерацію викликів WebAuthn та управління ключами автентифікації.

Для забезпечення цієї функціональності система використовує API, які відповідають за різні етапи реєстрації, входу та відновлення доступу. Нижче в таблиці 3.1 наведено детальний опис цих запитів, які реалізують інтеграцію клієнтської частини з сервером.

Таблиця 3.1 – Основні шляхи API для клієнтської взаємодії з серверною частиною

№	**Запит**	**Опис**	**Вхідні дані**	**Вихідні дані**	**Використання**
1.	GET /auth/webauthn/register/user	Перевіряє наявність користувача з вказаним email	email	Об'єкт користувача або null	Перед початком реєстрації
2.	POST /auth/webauthn/register/start	Стартує процес реєстрації, генерує challenge	JSON: { fullName, email}	Об'єкт із challenge та іншими даними	Для ініціації реєстрації
3.	POST /auth/webauthn/register/finish	Завершує реєстрацію, зберігає публічний ключ	JSON: { flowId, credential: {id, response } }	Статус завершення реєстрації	Завершення процесу реєстрації

Кінець таблиці 3.1.

4.	POST /auth/webauthn/login/start	Стартує процес входу, генерує challenge	JSON:{ email }	Об'єкт із даними для автенти фікації	Для ініціації входу
5.	POST /auth/webauthn/login/finish	Завершує процес входу, перевіряє публічний ключ	JSON: { flowId, credential: {id, response } }	Токен доступу (JWT)	Завер-шення автенти-фікації
6.	POST /auth/webauthn/lost_device	Надсилає email для відновлення доступу	JSON:{ email }	Статус: успішно, не знайден, помилка	При втраті пристрою з автенти-фікато-ром

Ці маршрути API забезпечують всі необхідні дії для реєстрації, входу та відновлення доступу користувачів, інтегруючи клієнтську частину з серверною логікою системи. Вони чітко структурують процеси автентифікації та гарантують коректну взаємодію в межах реалізованого функціоналу.

3.4 Реалізація панелі облікового запису користувача

Після успішного входу користувач переадресовується на заданий ендпойнт, який визначається конфігурацією системи. Водночас у тестовій системі користувач має можливість увійти до свого облікового запису через маршрут

/auth/account, де надається функція редагування персональних даних (за винятком електронної пошти). Цю можливість ілюструє рисунок 3.19

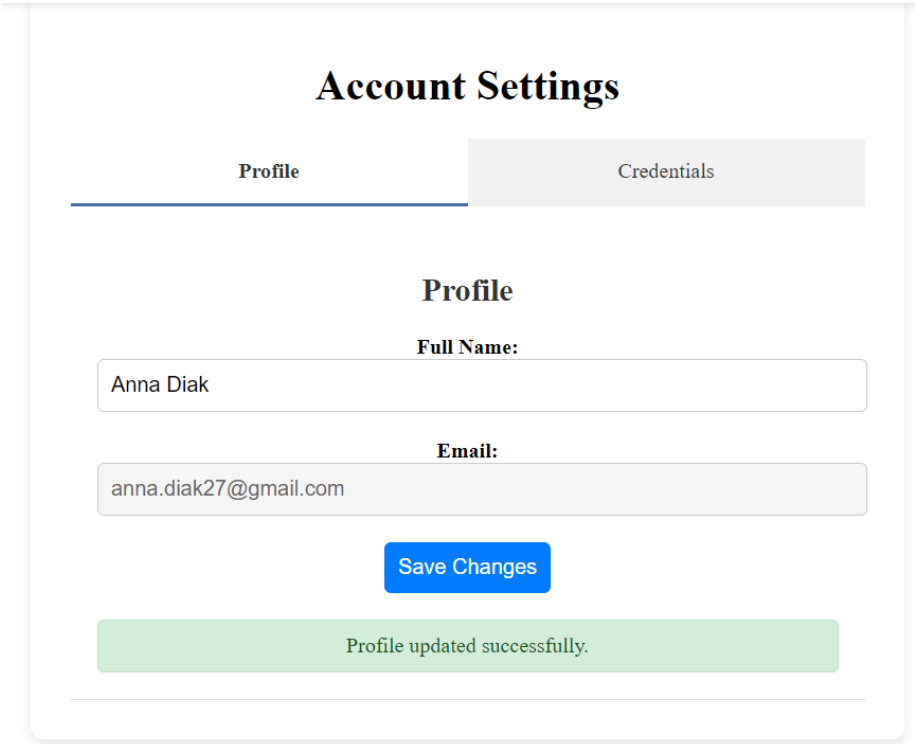


Рисунок 3.19 – Форма для реєстрації нових ключів

Також, користувач має можливість переглядати, додавати та видаляти зареєстровані ключі, які прив'язані до його облікового запису через різні пристрої. Це забезпечує гнучкість у керуванні автентифікаторами, дозволяючи користувачу підтримувати актуальність ключів, видаляти ті, які більше не використовуються, або додавати нові, наприклад, при переході на інший пристрій.

Ключі містять важливу інформацію, яка використовується для безпечної автентифікації. Зокрема, кожен ключ включає такі поля:

- дата створення – це допомагає відстежувати, коли саме ключ було створено, що є корисним для управління строком дії ключів або визначення застарілих записів;
- Key ID – унікальний ідентифікатор ключа, який дозволяє системі ідентифікувати його серед інших ключів, пов'язаних з обліковим записом;

- Public Key – відкритий ключ, що використовується для перевірки підписів при автентифікації. Цей ключ є частиною пари ключів і не потребує захищеного зберігання, оскільки він не розкриває приватну інформацію.

Приватний ключ, який є основою безпеки системи, залишається збереженим лише на пристрої, де його було створено. Через міркування безпеки інформація про точне місце зберігання приватного ключа не відображається у системі. Це рішення запобігає можливим витокам даних, забезпечуючи, щоб приватний ключ залишався доступним лише пристрою, де він був створений.

Користувач може керувати цими ключами через спеціальний інтерфейс у налаштуваннях облікового запису, зображений на рисунку 3.20. Наприклад, якщо користувач видаляє ключ, система гарантує, що цей ключ більше не зможе використовуватись для автентифікації. Аналогічно, при додаванні нового ключа користувачеві буде запропоновано створити пару ключів із дотриманням усіх необхідних заходів безпеки.

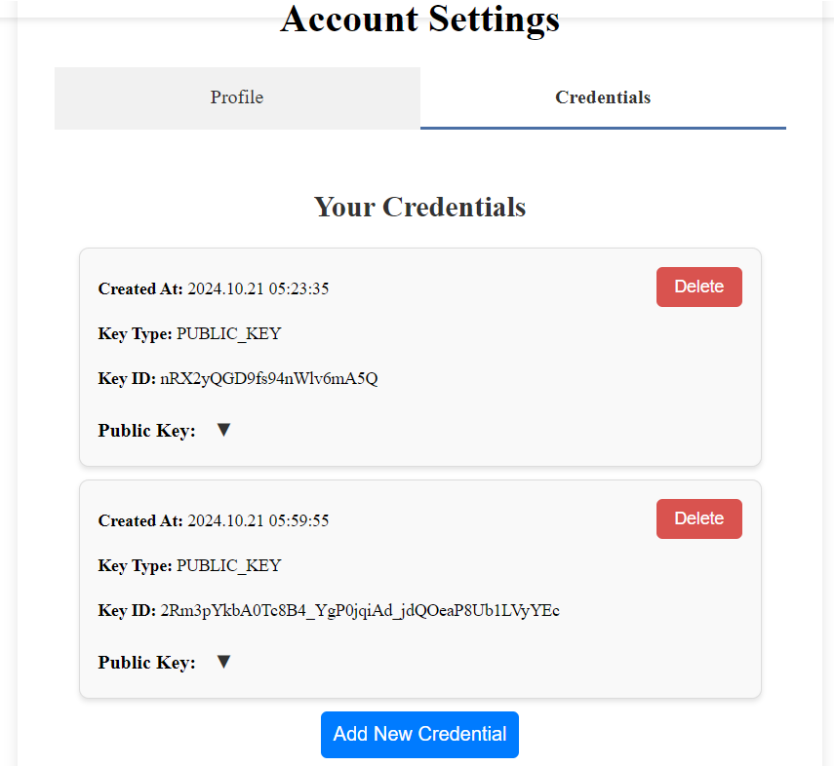


Рисунок 3.20 – Список ключів зареєстрованих в обліковому записі користувача

Даний спосіб є зручним в разі зміни чи втрати пристрою для автентифікації.

3.5 Тестування та налаштування сервісу

Для налаштування та тестування тестової системи необхідно забезпечити коректну конфігурацію змінних оточення, які використовуються для запуску програми. Усі налаштування здійснюються за допомогою скрипта “run_script.sh”, який розташований у корені проєкту. На рисунку 3.21 представлено цей скрипт, що демонструє ключові змінні для роботи системи. Ці змінні включають конфігурацію SSL, підключення до бази даних і параметри переадресації після входу до системи.

```
#!/bin/bash

# environment variables
export SERVER_PORT=2024

export SERVER_SSL_ENABLED=true
export SERVER_SSL_KEY_STORE_TYPE=PKCS12
export SERVER_SSL_KEY_STORE=classpath:/keystore.p12
export SERVER_SSL_KEY_STORE_PASSWORD=changeit
export SERVER_SSL_KEY_ALIAS=demo

export SPRING_DATASOURCE_URL=jdbc:postgresql://localhost:5432/as
export SPRING_DATASOURCE_USERNAME=postgres
export SPRING_DATASOURCE_PASSWORD=postgres
export SPRING_DATASOURCE_DRIVER_CLASS_NAME=org.postgresql.Driver

export SERVER_REDIRECT_URI=/auth/account

 # Replace with your project's directory
PROJECT_DIR="D:\Labs\Diploma 2\authorization-server"
cd "$PROJECT_DIR" || exit

# Run the Spring Boot application
echo "Running the Spring Boot application..."
java -jar target/*.jar \
  --server.port=$SERVER_PORT \
  --server.ssl.enabled=$SERVER_SSL_ENABLED \
  --server.ssl.key-store-type=$SERVER_SSL_KEY_STORE_TYPE \
  --server.ssl.key-store=$SERVER_SSL_KEY_STORE \
  --server.ssl.key-store-password=$SERVER_SSL_KEY_STORE_PASSWORD \
  --server.ssl.key-alias=$SERVER_SSL_KEY_ALIAS \
  --spring.datasource.url=$SPRING_DATASOURCE_URL \
  --spring.datasource.username=$SPRING_DATASOURCE_USERNAME \
  --spring.datasource.password=$SPRING_DATASOURCE_PASSWORD \
  --spring.datasource.driver-class-name=$SPRING_DATASOURCE_DRIVER_CLASS_NAME \
  --server.redirect-uri=$SERVER_REDIRECT_URI
```

Рисунок 3.21 – Сценарій налаштування та запуску тестової системи в скрипті run_script.sh

Конфігурація SSL задається через змінні в таблиці 3.2.

Таблиця 3.2 – Змінні для конфігурації SSL-з'єднання

Змінна	Опис
SERVER_SSL_ENABLED	Визначає, чи буде сервер працювати через HTTPS (повинна завжди бути true)
SERVER_SSL_KEY_STORE_TYPE	Формат keystore-файлу, що містить сертифікати
SERVER_SSL_KEY_STORE	Шлях до keystore-файлу, який містить SSL-сертифікати
SERVER_SSL_KEY_STORE_PASSWORD	Пароль для доступу до keystore-файлу
SERVER_SSL_KEY_ALIAS	Аліас для вибору конкретного ключа в keystore.

Без цих параметрів робота сервісу WebAuthn неможлива, оскільки стандарт вимагає захищеного HTTPS-з'єднання.

Для забезпечення взаємодії з базою даних необхідно налаштувати змінні, що зображені у таблиці 3.3.

Таблиця 3.3 – Змінні для конфігурації бази даних

Змінна	Опис
SPRING_DATASOURCE_URL	URL для підключення до БД, що включає сервер, порт та назву бази
SPRING_DATASOURCE_USERNAME	Ім’я користувача для підключення до БД
SPRING_DATASOURCE_PASSWORD	Пароль користувача для доступу до БД
SPRING_DATASOURCE_DRIVER_CLASS_NAME	Клас драйвера для підключення до БД

Ці параметри дозволяють системі зберігати та отримувати дані користувачів, ключі WebAuthn та інші об'єкти.

В кінці налаштування тестової системи слід не забути про змінну “SERVER_REDIRECT_URI”, яка визначає адресу, на яку користувач буде перенаправлений після успішного входу в систему.

Скрипт “run_script.sh” можна запустити напряму, що автоматично налаштує всі необхідні параметри для старту системи. Для цього достатньо виконати команду `./run_script.sh` в терміналі. Альтернативно, скрипт може бути інтегрований у Docker для більшої зручності розгортання. У цьому випадку конфігурація змінних передається у контейнер через аргументи командного рядка, забезпечуючи гнучке налаштування для різних середовищ.

Робота системи може бути протестована через ключові ендпоінти, такі як реєстрація користувача, вхід у систему та доступ до облікового запису. Тестування забезпечує впевненість у правильності налаштувань і функціонуванні всіх компонентів, включно із безпекою та інтеграцією з базою даних. Представлений рисунок 3.21 зі скриптом деталізує всі ключові елементи конфігурації та є основою для коректної роботи тестової системи.

.

3.6 Вимоги до технічного обладнання

Для забезпечення стабільної та надійної роботи сервісу безпарольної автентифікації на основі WebAuthn необхідно дотримуватися певних технічних вимог, які гарантують ефективність, швидкодію та безпеку системи.

Сервер, на якому розгортається сервіс, повинен мати 4-ядерний процесор для забезпечення швидкої обробки багатьох одночасних запитів від користувачів. Це є критично важливим для підтримки високої продуктивності системи в умовах великого навантаження. Мінімальний обсяг оперативної пам’яті має становити 8 ГБ, що дозволяє системі зберігати у пам'яті необхідні дані для обробки запитів та уникати затримок. Для зберігання даних необхідно забезпечити щонайменше 20

ГБ вільного місця на диску, що дозволить зберігати інформацію про користувачів, автентифікаційні ключі, дані сесій та журнали активності.

Для бази даних PostgreSQL, яка використовується для зберігання автентифікаційних ключів, записів сесій і інших важливих даних, рекомендовано мати сервер із 2-ядерним процесором і мінімум 4 ГБ оперативної пам'яті. Вимоги до дискового простору бази даних залежать від обсягу накопичуваних даних, тому слід передбачити резерв для подальшого розширення, враховуючи можливе збільшення кількості користувачів і сесій.

Для коректної роботи WebAuthn користувачі повинні використовувати сучасні браузери, які підтримують цей стандарт. Нижче в таблиці 3.4 наведено інформацію про підтримку WebAuthn у різних браузерах, що була сформована на основі інформації з сайту caniuse.com.

Таблиця 3.4 – Інформація про підтримку WebAuthn у різних браузерах

Браузер	**Підтримка версій**	**Часткова підтримка версій**	**Відсутня під-тримка версій**
Google Chrome	67 і новіші	Відсутня	4–66
Mozilla Firefox	60 і новіші	2–59	Відсутня
Microsoft Edge	18 і новіші	12–17 (необхідно увімкнути експериментальні функції)	12
Apple Safari	13 і новіші	3.2–12.1 (необхідно увімкнути експериментальні функції)	3.1 і раніше
Opera	54 і новіші	9–53	Відсутня
Opera mini	Відсутня	Відсутня	Відсутня

Таблиця 3.4 демонструє таблицю сумісності WebAuthn з різними версіями популярних браузерів та надає візуальне уявлення про те, які браузери

підтримують WebAuthn, що допомагає користувачам обрати відповідний браузер для роботи з сервісом.

Google Chrome, починаючи з версії 67, браузер повністю підтримує WebAuthn, що забезпечує користувачам можливість безпарольної автентифікації. Версії від 4 до 66 не підтримують цей стандарт, тому користувачам рекомендується оновити браузер до останньої версії.

Mozilla Firefox з версії 60 і новіші повністю підтримує WebAuthn. Версії від 2 до 59 мають часткову підтримку – деякі важливі функції, такі як використання зовнішніх апаратних токенів (наприклад, YubiKey), можуть працювати ненадійно або потребувати додаткових налаштувань.

Microsoft Edge з версії 18 повністю підтримує WebAuthn. Версії від 12 до 17 мають часткову підтримку, оскільки функція WebAuthn була доступна лише в експериментальному режимі. Це означає, що користувачі можуть використовувати базові функції WebAuthn, але для роботи з певними розширеними механізмами автентифікації (наприклад, з біометричними даними або токенами FIDO2) можуть знадобитися додаткові налаштування. Версія 12 взагалі не підтримує WebAuthn.

Apple Safari, починаючи з версії 13 повністю підтримує WebAuthn, забезпечуючи користувачам безпечну автентифікацію. Версії від 3.2 до 12.1 мають часткову підтримку, оскільки WebAuthn був доступний лише як експериментальна функція. Це означає, що в цих версіях деякі функції, такі як інтеграція з біометричними даними на пристрої (наприклад, з Touch ID), можуть працювати з перебоями або вимагати ввімкнення в налаштуваннях розробника. Версії 3.1 і раніше не підтримують цей стандарт взагалі.

Opera повністю підтримує WebAuthn з версії 54. Версії від 9 до 53 мають часткову підтримку, що означає, що деякі розширені функції WebAuthn (наприклад, інтеграція з апаратними токенами та використання біометричних даних) можуть не працювати належним чином або потребувати ввімкнення експериментальних функцій.

Для забезпечення ефективної роботи безпарольної автентифікації на основі WebAuthn, важливо ще також враховувати підтримку автентифікаторів на різних

операційних системах. Нагадаю, WebAuthn використовує два основних типи автентифікаторів, які були зазначені в розділі 1: платформні і роумінгові. Однак підтримка цих автентифікаторів варіюється залежно від браузера, операційної системи та типу підключення (USB, NFC, BLE).

Розглянемо спочатку підтримку платформних автентифікаторів в залежності від браузера та ОС на рисунку 3.22.

	Android 7+	iOS 14.5+	Windows 10 *(with Windows Hello)*	macOS Catalina	macOS Big Sur	Desktop Linux
Chrome	Yes	Yes	Yes	Yes	Yes	-
Safari	N/A	Yes	N/A	No	Yes	N/A
Firefox	No	Yes	Yes	No	No	-
Brave	No	Yes	Yes	Yes	Yes	-
Edge	No	Yes	Yes	Yes	Yes	-
Internet Explorer	N/A	N/A	No	N/A	N/A	N/A

Рисунок 3.22 – Матриця сумісності браузерів та операційних систем, що підтримують вбудовані автентифікатори (Touch ID, Face ID або Windows Hello)

На рисунку 3.22 позначення "Yes" вказує на повну підтримку платформних автентифікаторів у комбінації певного браузера та операційної системи. Позначка "No" свідчить про відсутність підтримки, через що функціонал автентифікації може бути недоступним. "N/A" (Not Applicable) означає, що зазначена комбінація не застосовується або не підтримується. Дефіс використовується для позначення відсутності інформації або невизначеності підтримки в конкретній комбінації.

Ці ж позначення застосовуються і для рисунка 3.23, який відображає сумісність браузерів та операційних систем із роумінговими автентифікаторами, такими як YubiKey і Titan Key.

Аналізуючи матрицю на рисунку 3.22, можна зробити висновок, що підтримка платформних автентифікаторів значною мірою залежить від комбінації браузера та операційної системи. Найкраща сумісність спостерігається у сучасних браузерах, таких як Google Chrome, Safari та Microsoft Edge, на популярних

операційних системах, включаючи Windows 10, macOS Big Sur, Android та iOS [38].

Водночас підтримка на старіших операційних системах або менш поширених браузерах, таких як Firefox на macOS або Desktop Linux, є обмеженою. Це підкреслює необхідність оновлення браузерів і операційних систем до останніх версій для забезпечення максимальної функціональності WebAuthn.

Далі розглянемо підтримку роумінгових автентифікаторів в залежності від браузера та ОС на рисунку 3.23.

	Android 7+	iOS 14.5+	Windows 10 *(with Windows Hello)*	macOS Catalina	macOS Big Sur	Desktop Linux
Chrome	Yes	Yes	Yes	Yes	Yes	-
Safari	N/A	Yes	N/A	No	Yes	N/A
Firefox	No	Yes	Yes	No	No	-
Brave	No	Yes	Yes	Yes	Yes	-
Edge	No	Yes	Yes	Yes	Yes	-
Internet Explorer	N/A	N/A	No	N/A	N/A	N/A

Рисунок 3.23 – Матриця сумісності браузерів та операційних систем, що підтримують зовнішніх апаратні автентифікатори (YubiKey, Titan Key)

З рисунку 3.23 можна зробити висновок, що найбільш стабільна підтримка цих пристроїв забезпечується через USB-з'єднання. Ця форма автентифікації підтримується практично всіма сучасними браузерами на таких популярних платформах, як Windows, macOS, Android та Linux [38].

Однак підтримка NFC і Bluetooth (BLE) є менш поширеною і залежить від браузера та операційної системи. Наприклад, браузери Safari та Firefox мають обмеження в роботі з BLE та NFC, що може вплинути на зручність використання зовнішніх токенів.

Дотримання цих технічних вимог допоможе забезпечити безперебійну роботу вашого сервісу та ефективну обробку запитів користувачів.

Цей розділ пропонує всебічний огляд процедури реалізації системи безпарольної автентифікації з використанням WebAuthn. У ньому описані основні етапи, від аналізу вимог до програмного забезпечення до тестування та налаштування сервісу. Архітектура програмного забезпечення була розроблена таким чином, щоб гарантувати стійку взаємодію між усіма компонентами системи, які включають клієнт, сервер та автентифікатори. Створено схему бази даних для безпечного зберігання ключів автентифікації та ефективного управління обліковими записами користувачів.

Додатково, було реалізовано інтуїтивно зрозумілі панелі реєстрації, входу та управління обліковими записами, що відповідають сучасним стандартам безпеки та забезпечують гнучкість у виборі автентифікаторів. Інтерфейс підтримує різні типи автентифікаторів, включаючи платформні та роумінгові автентифікатори, що забезпечує універсальний та безпечний користувацький досвід [39].

Система пройшла ретельне тестування, яке включало модульні, інтеграційні та навантажувальні тести. Виявлені проблеми були вирішені для забезпечення стабільної роботи в реальних умовах. Були встановлені специфічні технічні вимоги до середовища розгортання, що забезпечило стабільність, продуктивність і відповідність системи сучасним стандартам безпеки.

Результати цього розділу ілюструють готовність системи до розгортання у виробничому середовищі, пропонуючи безпечну і зручну безпарольну автентифікацію для користувачів, зберігаючи при цьому високу надійність і зручність для користувача.

4 РОЗРОБКА СТАРТАП ПРОЕКТУ

Цей розділ присвячено детальному розгляду основних аспектів розробки стартап-проєкту, включаючи визначення концепції, проведення технологічного аудиту, аналіз ринку та конкурентного середовища, оцінку ризиків та можливостей, а також розробку стратегії виходу на ринок та маркетингової програми. Особлива увага приділяється аналізу цільової аудиторії та визначенню сильних і слабких сторін проєкту, що дозволяє сформувати ефективну стратегію позиціонування та забезпечити успішне впровадження продукту на ринок.

4.1 Загальний опис стартап-проєкту

Розглянемо характеристику методів автентифікації у таблиці 4.1.

Таблиця 4.1 - Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Розробка системи безпарольної автентифікації з використанням стандартів WebAuthn і FIDO2	Корпоративні інформаційні системи	Підвищення рівня безпеки доступу до систем, зниження ризиків компрометації даних, усунення залежності від паролів.
	Веб-додатки	Забезпечення швидкого та безпечного входу для користувачів, сумісність із сучасними браузерами та платформами.

Кінець таблиці 4.1

	3. Мобільні додатки	Інтеграція біометричних методів автентифікації (відбитки пальців, розпізнавання обличчя), що підвищує зручність і безпеку для користувачів.
Інтеграція автентифікації за допомогою Passkeys	1. Хмарні сервіси	Забезпечення безпечного доступу до хмарних ресурсів, зниження ризиків витоку даних завдяки криптографічним методам.
	2. Інтернет-магазини	Підвищення довіри клієнтів через безпечну автентифікацію без введення паролів, що мінімізує ризики фішингових атак.
Впровадження Magic Links як додаткового методу автентифікації	1. Одноразовий доступ	Зручний спосіб автентифікації для користувачів, які рідко входять у систему, без необхідності запам'ятовувати облікові дані.

Наступним кроком є оцінка сильних і слабких сторін системи для визначення її потенціалу впровадження на ринок та конкурентоспроможності. Відповідна інформація представлена в Таблиці 4.2.

Таблиця 4.2 - Визначення характеристик ідеї проєкту

№	Економічні характеристики	Концепція (стратегія) конкурентів			Слабкі сторони	Нейтральні сторони	Сильні сторони
		Власний проєкт	Конкурент 1	Конкурент 2			
1	Форма виконання	Асиметрична криптографія, Passkeys	Паролі з MFA	Апаратні токени, FIDO2			+
2	Зручність використання	Інтеграція з біометрією	Залежність від складності паролів	Обмеження для користувачів без токенів			+
3	Кросплатформеність	Висока	Низька	Висока			+
4	Складність впровадження	Помірна	Низька	Висока	+		

Таблиця 4.2 демонструє, що запропонована система безпарольної автентифікації на основі асиметричної криптографії та passkeys має значні переваги над традиційними методами, такими як паролі з багатофакторною автентифікацією (MFA) та апаратні токени FIDO2. Зокрема, вона забезпечує високу зручність використання завдяки інтеграції з біометричними методами, кросплатформеність та помірну складність впровадження. Ці характеристики роблять її привабливим вибором для сучасних інформаційних систем, що прагнуть підвищити рівень безпеки та зручності для користувачів.

4.2 Технологічний аудит ідеї

У цьому розділі проведено аудит технологій, які можуть бути застосовані для реалізації запропонованої в дипломній роботі ідеї системи безпарольної автентифікації. Використані технології можна побачити в Таблиці 4.3

Таблиця 4.3 - Технологічна здійсненність ідеї проекту

Ідея проєкту	Технології її реалізації	Наявність технологій	Доступність технологій
Система безпарольної автентифікації	Мова програмування Java	Наявна	Безкоштовна, доступна
	Фреймворк Spring Framework	Наявна	Безкоштовна, доступна
	ORM фреймворк Hibernate	Наявна	Безкоштовна, доступна
	База даних PostgreSQL	Наявна	Безкоштовна, доступна
	Збірник проукту Maven	Наявна	Безкоштовна, доступна
	Мова програмування JavaScript	Наявна	Безкоштовна, доступна

Усі запропоновані технології, такі як Java, Spring Framework, Hibernate, PostgreSQL, Maven та JavaScript, є відкритими, безкоштовними та доступними для використання. Вони забезпечують високу ефективність розробки, надійність та легкість інтеграції в сучасні інформаційні системи.

Таким чином, об'єднання цих технологій створює потужний інструментарій для реалізації системи безпарольної автентифікації, яка відповідає сучасним вимогам безпеки та зручності.

4.3 Аналіз ринкових можливостей запуску стартап-проєкту

Аналіз ринкових можливостей, які можна використати під час впровадження проєкту, та загроз, що можуть завадити його реалізації, дозволяє спланувати напрями розвитку з урахуванням ринкового середовища, потреб потенційних клієнтів і пропозицій конкурентів. У таблиці 4.4 представлено попередню характеристику ринку для розроблюваного стартап-проєкту.

Таблиця 4.4 – Характеристика ринку систем безпарольної автентифікації

№	Характеристика ринку	Значення
1	Кількість гравців на ринку	Кілька великих компаній та численні стартапи
2	Загальний обсяг продаж на рік	Точні дані відсутні. Ринок зростає з кожним роком
3	Динаміка ринку	Стрімке зростання через підвищення уваги до кібербезпеки
4	Наявність обмежень для входу	Низькі. Необхідність відповідності стандартам безпеки
5	Наявність попиту	Високий попит серед підприємств
6	Середня рентабельність у галузі по ринку	Відсутні точні дані. Залежить від бізнес-моделі та масштабу операцій
7	Специфічні вимоги до сертифікації	Відповідність стандартам FIDO2 та іншим галузевим нормам

Відповідно до проаналізованих характеристик, ринок систем безпарольної автентифікації демонструє значний потенціал для впровадження нового продукту. Стрімке зростання попиту та відносно низькі бар'єри для входу. У таблиці 4.5 представлено детальну характеристику потенційних клієнтів.

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проєкту

№	Потреба, що формує ринок	Користувачі (основні клієнти)	Відмінності у поведінці різних потенційних цільових груп користувачів	Відмінності у поведінці різних потенційних цільових груп користувачів
1	Підви-щення безпеки доступу до систем та даних	Під-приємства різних галузей, фінансові установи, державні організації	Підприємства прагнуть захистити конфіденційні дані. фінансові установи зосереджені на захисті транзакцій; державні організації потребують відповідності нормативним вимогам	Високий рівень безпеки, відповідність стандартам, легкість інтеграції
2	Зручність та швидкість автентифі кації	Кінцеві користувачі, співробітни-ки компаній	Кінцеві користувачі цінують простоту. Співробітники компаній потребують швидкого доступу до робочих ресурсів	Інтуїтивно зрозумілий інтерфейс, мінімізація часу на автентифікацію
3	Зниження витрат на упра-вління паролями	ІТ-відділи компаній	ІТ-відділи прагнуть зменшити навантаження на підтримку	Можливість централізованого управління, зниження кількості запитів на скидання паролів

Задоволення вимог основних груп користувачів є ключовим фактором успіху стартап-проєкту. Розуміння специфічних потреб кожної цільової аудиторії дозволить розробити систему, яка відповідатиме очікуванням користувачів та забезпечить конкурентні переваги на ринку. Таким чином, основними вимогами є підвищення безпеки, зручність та швидкість автентифікації і зниження витрат.

У таблиці 4.6 наведено основні фактори загроз, їхній зміст та можливі реакції компанії.

Таблиця 4.6 – Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Зростаюча вимогливість користувачів	Підвищення очікувань щодо безпеки та зручності автентифікації	Постійне вдосконалення продукту, впровадження нових функцій
2	Посилення конкуренції	Вихід на ринок нових гравців із подібними рішеннями	Розробка унікальних пропозицій, підвищення якості обслуговування
3	Технологічні зміни	Швидкий розвиток технологій, що може зробити продукт застарілим	Інвестування в дослідження та розробки, адаптація до нових технологій
4	Регуляторні зміни	Введення нових законів та стандартів у сфері кібербезпеки	Відстеження змін у законодавстві, забезпечення відповідності продукту новим вимогам
5	Кіберзагрози	Збільшення кількості та складності кібератак	Підсилення захисту, проведення регулярних аудитів безпеки

Визначення цих загроз дозволяє компанії розробити стратегії для їхнього подолання та мінімізації потенційних ризиків.

У таблиці 4.7 представлено основні фактори можливостей, їхній зміст та можливі дії компанії для їхнього використання.

Таблиця 4.7 – Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Зростання попиту на безпечні рішення	Підвищення інтересу до безпарольної автентифікації	Активна маркетингова кампанія, розширення ринку збуту
2	Партнерство з великими компаніями	Можливість інтеграції з продуктами відомих брендів	Встановлення співпраці, розробка спільних рішень
3	Розширення функціоналу	Додавання нових можливостей до продукту	Проведення досліджень, впровадження інновацій
4	Вихід на міжнародний ринок	Вихід на міжнародний ринок	Адаптація продукту до вимог різних ринків, локалізація
5	Залучення інвестицій	Отримання додаткових ресурсів для розвитку	Підготовка інвестиційних пропозицій, участь у стартап-програмах

Використання можливостей зазначених в таблиці 4.7 сприятиме зростанню компанії та зміцненню її позицій на ринку.

Проведемо ступеневий аналіз конкуренції на ринку систем безпарольної автентифікації. Цей аналіз дозволяє зрозуміти поточний стан ринку та визначити стратегії для підвищення конкурентоспроможності компанії в сфері безпарольної автентифікації. Результати наведено в таблиці 4.8.

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку систем безпарольної автентифікації

№	Особливості конкурентного середовища	Прояви характеристики	Вплив на діяльність підприємства (можливі дії компанії для забезпечення конкурентоспроможності)
1	Тип конкуренції - монополістична	Наявність кількох великих гравців та численних стартапів	Вивчення ринку, визначення унікальних переваг продукту, розробка стратегії
2	Рівень конкуренції - міжнародний	Присутність як локальних, так і міжнародних компаній	Аналіз глобальних трендів, адаптація продукту до міжнародних стандартів, розширення на нові ринки
3	Галузева ознака - міжгалузева	Використання технологій у різних сферах (фінанси, охорона здоров'я, IT)	Розробка універсальних рішень, здатних задовольнити потреби різних галузей, гнучкість продукту
4	Конкуренція за видами товарів – товарно-видова	Різноманітність рішень з подібним функціоналом	Підвищення якості та безпеки продукту, впровадження інноваційних функцій, покращення користувацького досвіду
5	Характер конкурентних переваг – нецінова	Конкуренція базується на якості, безпеці та зручності	Інвестування в дослідження та розробки, забезпечення високого рівня обслуговування клієнтів, побудова сильної репутації
6	Інтенсивність конкуренції – висока	Швидкий розвиток технологій та поява нових гравців	Постійний моніторинг ринку, швидка адаптація до змін, впровадження нових технологій та рішень

Дослідження конкурентного середовища систем безпарольної автентифікації вказують на наявність монополістичної конкуренції, де співіснують як великі міжнародні корпорації, так і численні стартапи. Висока інтенсивність конкуренції та стрімкий розвиток технологій зобов'язують компанії постійно моніторити ринок, швидко адаптуватися до змін та впроваджувати новітні рішення, щоб залишатися конкурентоспроможними.

Наступним етапом є проведення аналізу конкурентного середовища за моделлю п'яти сил Майкла Портера, результати якого представлені в таблиці 4.9.

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти у галузі	Потенційні конкуренти	Постачальники	Клієнти	Продукт-замінники
	Наявність кількох великих гравців, таких як Google, Apple та Microsoft, які пропонують власні рішення для безпарольної автентифікації	Можливість входження на ринок нових компаній, зокрема стартапів, які розробляють інноваційні рішення у сфері кібербезпеки	Відсутність критичних постачальників, оскільки більшість технологій є відкритими або доступними для розробки власних рішень	Високий попит на безпечні та зручні методи автентифікації з боку бізнесу та кінцевих користувачі. Клієнти мають можливість вибору	Традиційні методи автентифікації, такі як паролі та двофакторна автентифікація, можуть виступати як замінники, але поступово втрачають популярність через нижчий рівень безпеки

Кінець таблиці 4.9.

Висно-вки	Помірна інтенсив-ність з боку великих техноло-гічних компаній та потенцій-них нових гравців	Низькі бар'єри для входження сприяють появі нових учасників ринку	Відсутність залежності від специфічних постачаль-ників знижує ризики	Високий попит на безпарольні рішення підвищує важливість задоволення потреб клієнтів	Традиційні методи автентифіка-ції втрачають актуаль-ність, що сприяє впрова-дженню безпароль-них технологій

На основі проведеного аналізу конкуренції (таблиця 4.2), характеристик ідеї стартап-проекту (таблиця 4.5), характеристик потенційних клієнтів та їхніх вимог до продукту (таблиця 4.6), а також факторів ринкового середовища (таблиці 4.7 і 4.8), сформульовано та обґрунтовано перелік факторів конкурентоспроможності, представлений у таблиці 4.10.

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспро-можності	Обґрунтування (чинники, що роблять фактор значущим для япорівняння конкурентних проектів)
1	Універсальність	Рішення з безпарольної автентифікації може бути інтегроване в різні системи та платформи

Кінець таблиці 4.10.

2	Зручність використання	Інтуїтивно зрозумілий інтерфейс та простота налаштування дозволяють користувачам швидко адаптуватися до нашого продукту без потреби в спеціальних знаннях чи тривалому навчанні
3	Рівень безпеки	Використання сучасних стандартів безпеки, таких як FIDO2 та WebAuthn, забезпечує надійний захист від фішингових атак та інших загроз, що підвищує довіру користувачів до нашого продукту
4	Кросплатформеність	Підтримка різних операційних систем та пристроїв дозволяє користувачам використовувати наше рішення на будь-якій платформі, що підвищує його привабливість та зручність
5	Масштабованість	Наше рішення легко адаптується до потреб як малого бізнесу, так і великих корпорацій, що дозволяє ефективно обслуговувати різні сегменти ринку та забезпечувати стабільне зростання

Аналіз конкурентоспроможності власного проєкту виявив ключові переваги, такі як універсальність, зручність використання, високий рівень безпеки, кросплатформеність та масштабованість. Ці характеристики дозволяють нашому рішенню не лише конкурувати з наявними альтернативами, але й забезпечувати виняткову цінність для користувачів, підвищуючи їхню довіру та задоволення. Завдяки цьому продукт здатний задовольнити потреби як індивідуальних користувачів, так і великих організацій, що шукають сучасні технології автентифікації.

З метою більш детального розуміння конкурентних переваг, проведено порівняльний аналіз слабких та сильних сторін з урахуванням зазначених факторів. Цей аналіз включає оцінку можливих ризиків, що можуть вплинути на сприйняття продукту на ринку, а також виявлення можливостей для подальшого вдосконалення. Результати порівняльного аналізу представлені у таблиці 4.11, що

дозволяє візуалізувати основні аспекти конкурентоспроможності та підтримувати ефективне стратегічне планування.

Таблиця 4.11 – Сильні та слабкі сторони стартап-проєкту

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг продуктів від конкуретів						
			-3	-2	-1	0	1	2	3
1	Універсальність	20				+			
2	Зручність використання	16					+		
3	Рівень безпеки	18			+				
4	Кросплатформеність	15		+					
5	Масштабованість	17					+		

Далі розглянемо таблицю 4.12 в якій вказано результати SWOT аналіз.

Таблиця 4.12 – SWOT-аналіз стартап-проєкту

Сильні сторони	Слабкі сторони
Інноваційний алгоритм безпарольної автентифікації, що забезпечує високий рівень безпеки та зручності для користувачів.	Відсутність впізнаваності бренду на ринку, що може ускладнювати залучення нових клієнтів.
Кросплатформеність рішення, що дозволяє інтегрувати його в різні системи та платформи	Обмежені фінансові та кадрові ресурси, що можуть впливати на швидкість розвитку та впровадження продукту
Простота використання, що знижує бар'єр входу для нових користувачів	Відсутність масштабних маркетингових кампаній.

SWOT-аналіз виявив, що наш стартап-проєкт має значні сильні сторони, такі як інноваційний алгоритм, кросплатформеність та простота використання. Однак, для успішного виходу на ринок необхідно подолати слабкі сторони, зокрема, підвищити впізнаваність бренду та розширити фінансові і кадрові ресурси. Стрімкий розвиток індустрії та зростаючий попит на безпарольні рішення відкривають значні можливості для зростання, проте конкуренція з боку більш відомих компаній та швидкий технологічний прогрес становлять потенційні загрози, які слід враховувати при розробці стратегії розвитку.

Далі, в таблиці 4.13 розглянемо альтернативи впровадження проєкту на основі SWOT аналізу.

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проєкту

№	Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Інтеграція з популярними платформами (наприклад Microsoft Azure Active Directory)	80%	12 місяців
2	Співпраця з фінансовими установами (банки або інші фінансові організації)	60%	15 місяців
3	Участь у міжнародних виставках та конференціях	50%	6 місяців
4	Розробка безкоштовної пробної версії продукту для залучення нових користувачів	75%	3 місяці
5	Проведення вебінарів та онлайн-семінарів для демонстрації переваг продукту	70%	2 місяці

На цьому етапі проведено всебічний аналіз ринку та продукту. Зокрема, здійснено детальний конкурентний аналіз, визначено ключові ринкові фактори та оцінено їх сприятливість для нашого стартап-проєкту. Також проведено глибоке дослідження концепції та характеристик пропонованого продукту. На основі отриманих даних можна зробити висновок, що поточні ринкові умови є сприятливими для успішного виходу нашого продукту на ринок. Це відкриває значні можливості для подальшого розвитку та масштабування проєкту.

4.4 Розробка стратегії для виходу на ринок

Для ефективного впровадження рішення з безпарольної автентифікації необхідно ідентифікувати основні групи потенційних споживачів, оцінити їхню готовність до впровадження продукту, орієнтовний попит, рівень конкуренції та складність виходу на відповідні сегменти ринку.

У таблиці 4.14 представлено аналіз цільових груп потенційних клієнтів.

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегмент)	Інтенсивність конкуренції в сегменті	Складність входу у сегмент
1	Індивідуальні користувачі	Висока	Середній	Низька	Низька
2	Малі компанії	Висока	Високий	Середня	Середня
3	Великі компанії	Висока	Високий	Висока	Висока

Аналіз цільових груп потенційних споживачів показує, що всі сегменти демонструють високу готовність до впровадження безпарольної автентифікації.

Проте інтенсивність конкуренції та складність входу на ринок зростають від простих користувачів до великих компаній. Це вимагає адаптації маркетингових стратегій та ресурсів для ефективного проникнення в кожен із сегментів. Зокрема, для залучення великих компаній необхідно підкреслити переваги безпарольної автентифікації, такі як підвищення безпеки та зниження витрат на управління паролями. Водночас, для простих користувачів слід акцентувати увагу на зручності та простоті використання продукту.

Для визначення базової стратегії розвитку проєкту з безпарольної автентифікації було проведено аналіз ринкової позиції, цільової аудиторії та конкурентних характеристик. Результати цього аналізу представлені в таблиці 4.15.

Таблиця 4.15 – Визначення базової стратегії розвитку

№	Чи є проєкт «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів або залучати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які саме?	Стратегія конкурентної поведінки
1	Ні	Залучення нових споживачів та перехід існуючих від конкурентів	Використання найкращих практик конкурентів, зокрема щодо надійності та точності	Наступальна

Для ефективного позиціонування нашого програмного продукту на ринку необхідно врахувати вимоги цільової аудиторії та визначити ключові конкурентні переваги. Результати цього аналізу представлені в таблиці 4.16.

Таблиця 4.16 – Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту (три ключових)
1	Надійність, точність, простота використання	Оптимізація витрат	Якість та ефективність	Надійність, точність, простота використання

У цьому розділі визначено стратегію позиціонування програмного продукту, яка базується на трьох ключових аспектах: ефективність, точність та простота використання алгоритму. Ці характеристики відповідають основним вимогам цільової аудиторії та підкреслюють конкурентні переваги стартап-проєкту.

4.5 Розробка маркетингової програми

Для розробки ефективної маркетингової програми необхідно чітко визначити концепцію товару, який пропонується споживачам. Це включає аналіз потреб цільової аудиторії, вигод, які надає продукт, та його конкурентних переваг. У таблиці 4.17 представлено підсумок аналізу конкурентоспроможності товару, що був проведений раніше.

Таблиця 4.17 – Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Якісний захист від загроз	Надійна безпарольна автентифікація, що знижує ризик компрометації облікових записів	Використання передових алгоритмів шифрування та багатофакторної автентифікації для забезпечення високого рівня безпеки
2	Універсальність	Сумісність із різними платформами та пристроями, що забезпечує зручність використання	Гнучка інтеграція з існуючими системами безпеки та можливість налаштування під специфічні потреби клієнтів
3	Підтримка	Регулярні оновлення та оперативна технічна підтримка для користувачів	Наявність цілодобової служби підтримки та швидке реагування на виявлені вразливості або запити клієнтів

У таблиці 4.18 наведено опис трьох рівнів моделі товару для стартап-проекту.

Таблиця 4.18 – Опис трьох рівнів моделі товару для стартап-проєкту

Рівень товару	Сутність та складові

Кінець таблиці 4.18.

<table>
<tr><td>I. Товар за задумом</td><td colspan="2">Система безпарольної автентифікації, що забезпечує безпечний та зручний доступ користувачів до онлайн-сервісів без використання традиційних паролів</td></tr>
<tr><td rowspan="4">II. Товар у реальному виконанні</td><td>Властивості/характеристики:</td><td>Розмір:</td></tr>
<tr><td>Клієнтський додаток</td><td>20 МБ</td></tr>
<tr><td>Серверний модуль</td><td>50 МБ</td></tr>
<tr><td colspan="2">Якість: висока надійність та швидкодія системи, підтримка багатофакторної автентифікації, сумісність із різними платформами та пристроями</td></tr>
<tr><td rowspan="2"></td><td colspan="2">Пакування: електронна ліцензія з унікальним ключем активації, документація та інструкції з налаштування та використання</td></tr>
<tr><td colspan="2">Марка: назва компанії-розробника та назва продукту, що підкреслюють інноваційність та безпеку рішення</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="2">До продажу: демонстраційна версія з обмеженим функціоналом для ознайомлення клієнтів, консультації щодо інтеграції та налаштування системи</td></tr>
<tr><td colspan="2">Захист від копіювання: прив’язка ліцензії до конкретного домену або IP-адреси, активація продукту через онлайн-сервіс з перевіркою автентичності ліцензійного ключа, використання шифрування для захисту коду</td></tr>
</table>

Розглянемо цінові межі визначені за допомогою експертного метода в таблиці 4.19.

Таблиця 4.19 – Формування цінових меж

№	Рівень цін на товари-замінники (за 1000 корист/міс)	Рівень цін на товари-аналоги (за 1000 корист/міс)	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу (за 1000 корист/міс)
1	3000$	6000$	100000$	2000$ - 7000$

У таблиці 4.20 зображено формування системи збуту для продукту.

Таблиця 4.20 – Формування системи збуту

№	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Придбання ліцензії на продукт або укладання договору про надання послуг без передачі права власності на програмне забезпечення	- Надання доступу до API через веб-сервіс; - Розміщення ключів на веб-сервісах;	Однорівневий канал збуту (прямий контакт із кінцевим споживачем)	Канал збуту одного рівня

У таблиці 4.21 наведено концепцію маркетингових комунікацій для продукту.

Таблиця 4.21 – Концепція маркетингових комунікацій

№	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Клієнти шукають нові засоби підвищення безпеки та зручності автентифікації	Професійні конференції, спеціалізовані вебінари, галузеві видання, соціальні мережі (LinkedIn)	Надійність, простота впровадження, сумісність із існуючими системами	Підкреслити переваги безпарольної автентифікації, зокрема підвищення безпеки та зручності для користувачів	"Забудьте про паролі – впроваджуйте сучасні рішення для безпечної та простої автентифікації"

Таблиця 4.21 відображає концепцію маркетингових комунікацій для проєкту безпарольної автентифікації, зосереджуючись на поведінці цільових клієнтів, які шукають нові засоби підвищення безпеки та зручності автентифікації. Основними каналами комунікації є професійні конференції, спеціалізовані вебінари, галузеві видання та соціальні мережі, такі як LinkedIn. Позиціонування продукту базується на надійності, простоті впровадження та сумісності з існуючими системами. Завдання рекламного повідомлення полягає в підкресленні переваг безпарольної автентифікації, зокрема підвищення безпеки та зручності для користувачів.

У цьому розділі проведено комплексний маркетинговий аналіз, що є ключовим етапом у розробці стартап-проєкту. Визначено основну концепцію проєкту, проведено технологічний аудит для оцінки доцільності використання різних технологій, здійснено детальний аналіз ринку, включаючи оцінку

конкурентного середовища та визначення ринкових тенденцій. Також проаналізовано ризики та можливості, визначено сильні та слабкі сторони проєкту, а також проведено аналіз цільової аудиторії. На основі зібраних даних розроблено стратегію виходу на ринок та маркетингову програму, спрямовані на ефективне позиціонування проєкту, залучення цільової аудиторії та досягнення поставлених бізнес-цілей. Результати аналізу свідчать про високу конкурентоспроможність та економічну доцільність проєкту, що підтверджує його готовність до впровадження на ринок.

ВИСНОВКИ

В даній роботі було розроблено та протестовано систему безпарольної автентифікації, що базується на сучасних стандартах WebAuthn та FIDO2. Система забезпечує стійкість до фішингових атак та атак "людина посередині" (MITM), а також підвищує зручність автентифікації для користувачів завдяки інтеграції асиметричної криптографії та використанню апаратних токенів [40].

Для інтеграції WebAuthn було обрано бібліотеку Yubico, яка спростила процес розробки та забезпечила відповідність стандартам безпеки. Створено модель системи безпарольної автентифікації, що включає механізми реєстрації користувачів, автентифікації за допомогою публічних та приватних ключів, а також управління криптографічними ключами.

Розгорнуто тестову систему, у якій налаштовано процеси реєстрації, входу та відновлення доступу. Додатково реалізовано конфігурацію для роботи із захищеним SSL-з'єднанням, інтеграцію з базою даних PostgreSQL, а також підтримку гнучкого налаштування через змінні оточення.

Запропоноване рішення дозволяє значно підвищити безпеку користувачів та зменшити ризики, пов'язані з використанням паролів, оскільки у процесі автентифікації паролі не використовуються взагалі. Крім того, інтеграція біометричних даних та апаратних токенів забезпечує додатковий рівень захисту і робить систему більш зручною для кінцевих користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. America's Password Habits | Security.org. Security.org. URL: https://www.security.org/resources/online-password-strategies/ (дата звернення: 03.10.2024).

2. Das S., Phelan L. A., Hoyos-Rivera J. A. Password Managers Usage and Trust: A US Study of User Behavior, Preferences, and Perceptions. ACM Transactions on Privacy and Security. Vol. 24, no. 3, article 16, July 2021. URL: https://doi.org/10.1145/3447564 (date of access: 09.10.2024).

3. Saul Johnson, Jo˜ao F. Ferreira, Alexandra Mendes, and Julien Cordry. “Skeptic: Automatic, justified and privacy-preserving password composition policy selection”. In: Proceedings of the 15th ACM Asia Conf

4. What is WebAuthn? Authentication standard. Wallarm | Integrated App and API Security Platform. URL: https://www.wallarm.com/what/webauthn-web-authentication (дата звернення: 05.10.2024).

5. Gordin, A. Graur, S. Vlad and C. I. Adomniței, "Moving forward passwordless authentication: challenges and implementations for the private cloud," 20th RoEduNet Conference: Networking in Education and Research (RoEduNet), 2021, p. 1-5, doi: 10.1109/RoEduNet54112.2021.9638271 (date of access: 09.10.2024).

6. Decoding How WebAuthn Works. FusionAuth. URL: https://fusionauth.io/articles/authentication/webauthn (дата звернення: 05.10.2024).

7. Що таке ключ Passkey та як він працює? Просте пояснення | Hideez. Passwordless Workforce Identity Solutions | Hideez. URL: https://hideez.com/uk-ua/blogs/news/what-is-a-passkey (дата звернення: 25.11.2024).

8. Rolfe B. Synced vs Device-Bound Passkeys: How User Convenience and Authentication Experiences Vary. Authsignal - Drop-in Passkeys & Passwordless Authentication. URL: https://www.authsignal.com/blog/articles/synced-vs-device-bound-passkeys-convenience-and-authentication-experiences (date of access: 25.11.2024).

9. Device-Bound vs. Synced Passkeys (SCA & Passkeys I). Corbado - Add passkeys to any new or existing app. URL: https://corbado.com/blog/device-bound-synced-passkeys (date of access: 25.11.2024).

10. HYPR. What is a FIDO Platform Authenticator? | Security Encyclopedia. HYPR: Identity Security & Passwordless Authentication Solution. URL: https://www.hypr.com/security-encyclopedia/platform-authenticator (date of access: 25.11.2024).

11. Platform vs Cross-Platform. Yubico Developers. URL: https://developers.yubico.com/WebAuthn/WebAuthn_Developer_Guide/Platform_vs_Cross-Platform.html (date of access: 25.11.2024).

12. NIST SP 800-53 Rev. 5: Security and Privacy Controls for Information Systems and Organizations. Independently Published, 2022, 462 p.

13. Blokdyk G. Passwordless Authentication, Second Edition. 5STARCooks, 2021. 80-93 p.

14. Методи аутентифікації без пароля - Wise IT Ukraine. Wise IT Ukraine. URL: https://wiseit.com.ua/metody-autentyfikacziyi-bez-parolya-vid-fudo/ (дата звернення: 30.09.2024).

15. A Study on Passwordless Authentication Technology and Its Effects. The International Journal of Reliable Information and Assurance. 2018. Vol. 6, no. 1. URL: https://doi.org/10.21742/ijria.2018.6.1.03 (date of access: 19.11.2024).

16. Passwordless Magic Links vs. Certificates for Secure Access. SecureW2. URL: https://www.securew2.com/blog/passwordless-magic-link-authentication-explained (date of access: 25.11.2024).

17. Java: основні принципи та переваги використання. www.mathros.net.ua - Сайт для студентів спеціальності інформатика. URL: https://www.mathros.net.ua/basic-principles-of-java-programming.html (дата звернення: 25.11.2024).

18. Java: The Complete Reference, Thirteenth Edition. McGraw-Hill Education, 2023.

19. Horstmann C. S. Core Java, Volume II--Advanced Features (11th Edition). Prentice Hall, 2019. 26-28 p.

20. Scarioni C., Nardone M. Pro Spring Security: Securing Spring Framework 5 and Boot 2-based Java Applications. Apress, 2019. 428 p.

21. Spring Boot. Spring Boot. URL: https://spring.io/projects/spring-boot (date of access: 25.11.2024).

22. Spring Data. Spring Data. URL: https://spring.io/projects/spring-data (date of access: 25.11.2024).

23. Macero García M., Telang T. Learn Microservices with Spring Boot 3. 3rd ed. Berkeley, CA : Apress, 2023. 11-18 p.

24. Yubico Developers. Yubico Developers. URL: https://developers.yubico.com/ (date of access: 25.11.2024).

25. Tudose C. Java Persistence with Spring Data and Hibernate. Manning, 2022. 625 p.

26. Hibernate. Everything data. Hibernate. URL: https://hibernate.org/ (date of access: 25.11.2024).

27. Apache Maven Series | Baeldung. Baeldung. URL: https://www.baeldung.com/maven-series (дата звернення: 11.10.2024).

28. Gaba I. What is Maven: Here's What You Need to Know [Updated]. Simplilearn.com. URL: https://www.simplilearn.com/tutorials/maven-tutorial/what-is-maven (date of access: 25.11.2024).

29. What Is PostgreSQL?. Kinsta®. URL: https://kinsta.com/knowledgebase/what-is-postgresql/#what-is-postgresql (date of access: 29.09.2024).

30. Schönig H.-J. Mastering PostgreSQL 12: Advanced Techniques to Build and Administer Scalable and Reliable PostgreSQL Database Applications, 3rd Edition. Packt Publishing, Limited, 2019.

31. Mastering PostgreSQL 15: Advanced techniques to build and manage scalable, reliable, and fault-tolerant database applications, 5th Edition. Packt Publishing, 2023. 522 p.

32. Blog H. H. T. Passwordless Authentication With Passkey: How It Works and Why It Matters—Part 1. Medium. URL:

https://medium.com/@heritage.tech/passwordless-authentication-with-passkey-how-it-works-and-why-it-matters-part-1-dcae2a004988 (date of access: 29.09.2024).

33. Web client definition Glossary. NordVPN. URL: https://nordvpn.com/uk/cybersecurity/glossary/web-client/ (date of access: 09.10.2024).

34. HYPR. What is a FIDO Relying Party (RP)? | Security Encyclopedia. Identity Security & Passwordless Authentication Solution | HYPR. URL: https://www.hypr.com/security-encyclopedia/relying-party-rp (date of access: 05.10.2024).

35. Is FIDO2 the Kingslayer of User Authentication? A Comparative Usability Study of FIDO2 Passwordless Authentication / S. Ghorbani Lyastani et al. 2020 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 18–21 May 2020. 2020. URL: https://doi.org/10.1109/sp40000.2020.00047 (date of access: 19.11.2024).

36. Biometrics, FIDO, and More: A Guide to Passwordless Authentication Methods. BIO-key Blog. URL: https://blog.bio-key.com/biometrics-fido-and-more-a-guide-to-passwordless-authentication-methods (дата звернення: 09.10.2024).

37. What is a relational database?. Oracle | Cloud Applications and Cloud Platform. URL: https://www.oracle.com/database/what-is-a-relational-database/ (date of access: 09.10.2024).

38. Does my browser support WebAuthn?. Does my browser support WebAuthn?. URL: https://webauthn.me/browser-support (date of access: 25.11.2024).

39. Rasmussen B. A Usability Study of FIDO2 Roaming Software Tokens as a Password Replacement. 2021. URL: https://scholarsarchive.byu.edu/etd/9227 (date of access: 19.11.2024).

40. What is Passwordless Authentication and How Does It Work?. LoginTC. URL: https://www.logintc.com/types-of-authentication/passwordless-authentication/ (date of access: 09.10.2024).

ДОДАТОК А

МОДЕЛЮВАННЯ СИСТЕМ З БЕЗПАРОЛЬНОЮ АВТЕНТИФІКАЦІЄЮ НА ОСНОВІ ПРОМИСЛОВИХ СТАНДАРТІВ

Тези на І Міжнародну науково-практичну конференцію
“Проблеми комп’ютерних наук, програмного моделювання та безпеки цифрових систем”,
13-16 червня 2024 року, м. Луцьк-Світязь

УКР.НТУУ”КПІ ім. Ігоря Сікорського” _ІАТЕ_ЦТЕ_ ТР-32мп

Аркуші 3

Київ — 2024

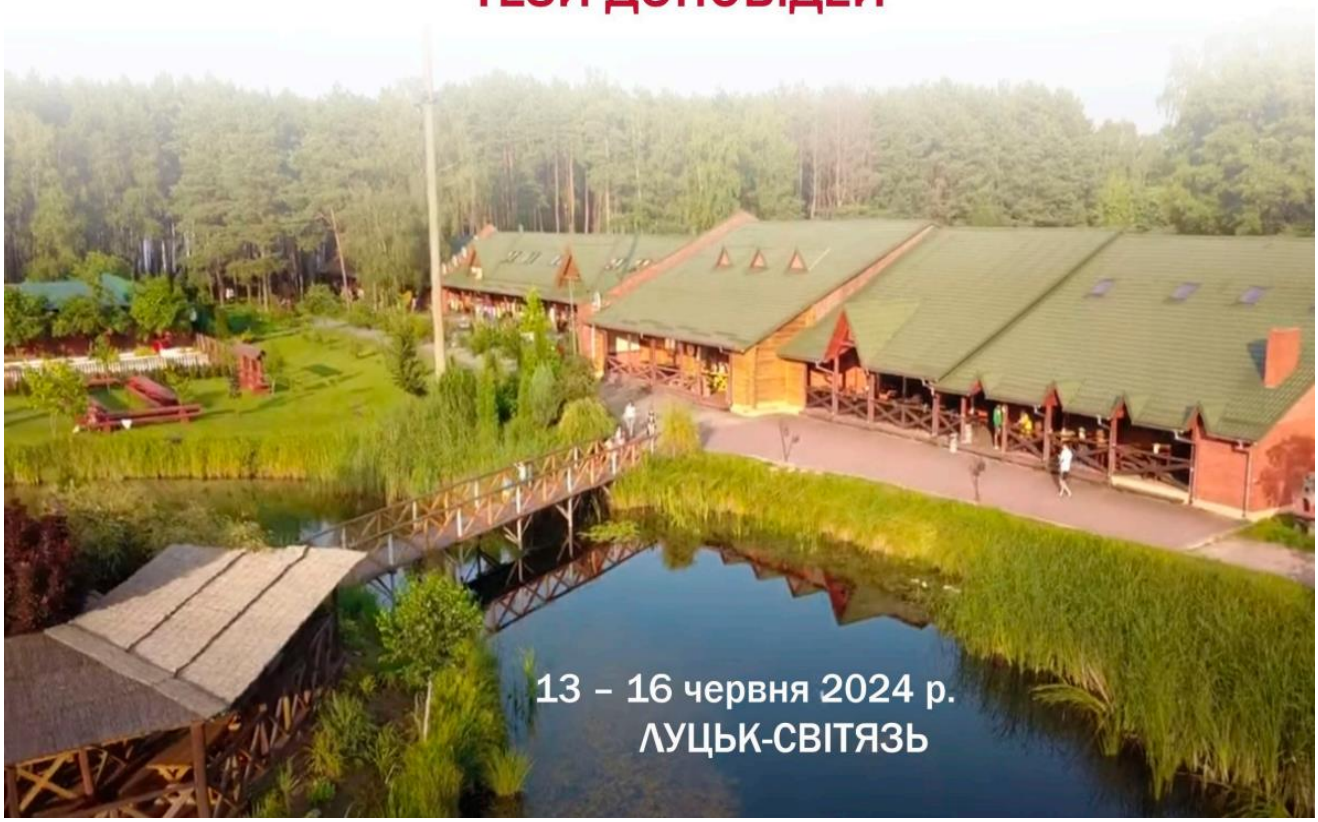
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ЛЕСІ УКРАЇНКИ

LESYA UKRAINKA VOLYN NATIONAL UNIVERSITY

I МІЖНАРОДНА НАУКОВО-ПРАКТИЧНА КОНФЕРЕНЦІЯ

ПРОБЛЕМИ КОМП'ЮТЕРНИХ НАУК, ПРОГРАМНОГО МОДЕЛЮВАННЯ ТА БЕЗПЕКИ ЦИФРОВИХ СИСТЕМ

ТЕЗИ ДОПОВІДЕЙ



УДК 004.921

МОДЕЛЮВАННЯ СИСТЕМ З БЕЗПАРОЛЬНОЮ АВТЕНТИФІКАЦІЄЮ НА ОСНОВІ ПРОМИСЛОВИХ СТАНДАРТІВ

MODELING SYSTEMS WITH PASSWORDLESS AUTHENTICATION BASED ON INDUSTRY STANDARDS

Юрій Тарнавський, Анна Дяк

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», просп. Берестейський, 37, Київ, 03056, Україна

Оскільки кіберзагрози продовжують зростати, обмеження та вразливості традиційних систем автентифікації на основі паролів стають все більш очевидними. Багатообіцяючою альтернативою стає безпарольна автентифікація, що пропонує підвищену безпеку та зручність для користувачів.

Традиційна автентифікація значною мірою покладається на паролі, які часто є слабкими, повторно використовуваними та вразливими до фішингу, атак грубої сили та зломів баз даних. На противагу цьому, безпарольна автентифікація усуває потребу в паролях, використовуючи безпечніші методи, такі як біометрія, апаратні токени (YubiKeys) та програмні додатки, які здійснюють автентифікацію за допомогою push-повідомлень або одноразових паролів, прив'язаних до певного часу (TOTP).

Було розроблено кілька галузевих стандартів, які керують впровадженням систем безпарольної автентифікації: FIDO2/WebAuthn, NIST SP 800-63 та OAuth2.0.

FIDO2/WebAuthn забезпечує безпарольну автентифікацію за допомогою криптографії з відкритим ключем. WebAuthn дозволяє веб-програмам використовувати автентифікатори, такі як біометричні дані або апаратні токени. Під час реєстрації автентифікатор створює та зберігає приватний ключ, надсилаючи відкритий ключ на сервер. Для автентифікації він підписує виклик сервера закритим ключем, безпечно перевіряючи користувача.

NIST SP 800-63 містить настанови щодо управління цифровою ідентичністю, зосереджуючись на оцінці ризиків, перевірці ідентичності, рівнях забезпечення автентичності (AAL) та федерації. Стандарт заохочує безпарольну автентифікацію та пропонує рекомендації для процесів автентифікації, адаптованих до різних рівнів ризику.

OAuth 2.0 - це фреймворк авторизації, який дозволяє стороннім додаткам отримувати обмежений доступ до ресурсів користувача без розкриття облікових даних. OpenID Connect базується на OAuth 2.0, забезпечуючи автентифікацію на додаток до авторизації за допомогою веб-токенів JSON (JWT) і підтримуючи безперебійний процес єдиного входу.

Моделювання систем з безпарольною автентифікацією на основі стандартів FIDO2/WebAuthn, NIST SP 800-63 та OAuth 2.0 значно підвищує безпеку і зручність для користувачів. Безпарольні методи усувають вразливості традиційних паролів. Використання цих стандартів дозволяє створювати надійні системи автентифікації, що відповідають сучасним вимогам кібербезпеки.

Бібліографія

1. Blokdyk G. Paswordless Authentication, Second Edition. 5STARCooks, 2021. 80-93 p.
2. Cybersecurity Essentials / C. J. Brooks et al. Hoboken. John Wiley & Sons, 2018. 784 p.