КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма «Управління проєктами»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА
на тему:

«Дослідження процесів управління проєктом створення SaaS-платформи для обміну електронними документами»

Студента 2-го курсу групи УПз-21

Олександра КУЗЬМЕНКА
(ім'я, прізвище)

(підпис студента)

Науковий керівник:

к.т.н.
(науковий ступінь, вчене звання)

Тетяна ЛАТИШЕВА
(ім'я, прізвище)

(дата) *(підпис)*

Попередній захист:

(Висновок: "До захисту в Екзаменаційній комісії")

Завідувач кафедри
технологій управління _____ Віктор МОРОЗОВ _____
(підпис) *(ім'я, прізвище)* *(дата)*

Київ – 2025

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій

Кафедра технологій управління
Освітній рівень Магістр
Спеціальність 122 Комп'ютерні науки
Освітньо-професійна програма Управління проєктами

ЗАТВЕРДЖУЮ
Завідувач кафедри
професор Віктор МОРОЗОВ

_____
«27» вересня 2025 року

З А В Д А Н Н Я
НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студента: Олександра КУЗЬМЕНКА
Група: УПз-21

1. Тема кваліфікаційної роботи: «Дослідження процесів управління проєктом створення SaaS-платформи для обміну електронними документами»
Затверджена протоколом № 15 від 16.06.2025 р.

2. Строк подання студентом готової роботи – «04» 12.2025 р.

3. Цільова установка та вихідні дані до роботи: дослідження методів управління IT-проєктами в сфері LegalTech; аналіз нормативно-правової бази (КЕП, eIDAS) та існуючих рішень на ринку; обґрунтування вибору методології розробки; формування організаційної структури команди та розподіл ролей; планування бюджету, розробка календарного графіку та створення прототипу SaaS-платформи «UniSign».

4. Зміст роботи: Обґрунтування актуальності та доцільності створення SaaS-платформи для транскордонного документообігу; аналіз ринку та нормативно-правової бази (КЕП, eIDAS); PEST та SWOT-аналіз середовища проєкту; розробка концептуальної моделі системи та формалізація задач управління ресурсами і часом; аналіз та пріоритизація вимог, моделювання сценаріїв взаємодії (Use Cases); проєктування архітектури, бази даних та програмна реалізація MVP; формування організаційної структури та матриці відповідальності (RACI); розробка ієрархічної структури робіт (WBS), календарного плану та бюджету; оцінка ризиків та економічної ефективності проєкту.

5. Перелік графічного матеріалу: титульний слайд, структурна модель проблем та аналіз середовища (PEST, SWOT), дерево цілей проєкту,

концептуальна схема та функціональна декомпозиція системи, діаграма варіантів використання (Use Case), математичні моделі управління проєктом, концептуальна, логічна та фізична моделі бази даних, архітектура та інтерфейси програмного забезпечення, організаційна структура команди та матриця відповідальності (RACI), структурна декомпозиція робіт (WBS), календарний план та бюджет проєкту, діаграма Ганта, реєстр ризиків та графік окупності, висновки.

6. Календарний план виконання роботи

№ з/п	Назва частин роботи	Виконання роботи
1	Вивчення літературних джерел з предмету дослідження	02.10.25-12.10.25
2	Збір і вивчення матеріалів	13.10.25-19.10.25
3	Складання розгорнутого плану кваліфікаційної роботи	20.10.25-23.10.25
4	Ознайомлення наукового керівника з розгорнутим планом кваліфікаційної роботи. Внесення змін	24.10.25-25.10.25
5	Підготовка розділу 1	26.10.25-03.11.25
6	Підготовка розділу 2	04.11.25-14.03.25
7	Підготовка розділу 3	15.11.25-23.11.25
8	Підготовка розділу 4	24.11.25-05.12.25
9	Оформлення кваліфікаційної роботи	25.11.25-03.12.25
10	Передача кваліфікаційної роботи науковому керівникові	04.12.25
11	Попередній захист кваліфікаційної роботи	09.12.25-13.12.25
12	Передача кваліфікаційної роботи рецензенту для рецензування	14.12.25

Дата видачі завдання «30» вересня 2025 р.
Керівник роботи к.т.н., Тетяна ЛАТИШЕВА
(посада, ім'я, прізвище)

_____
(підпис)

Завдання прийняв до виконання студент групи УПз-21
Олександр КУЗЬМЕНКО
(ім'я, прізвище)

_____
(підпис)

АНОТАЦІЯ

кваліфікаційної роботи магістра на тему

«Дослідження процесів управління проєктом створення SaaS-платформи для обміну електронними документами»

Студента: Кузьменка Олександра Сергійовича

Науковий керівник: Латишева Тетяна Володимирівна

Рік захисту — 2025

Мета кваліфікаційної роботи полягає у розробці комплексної системи управління проєктом створення SaaS-платформи «UniSign» для транскордонного електронного документообігу. Дослідження спрямоване на вирішення проблеми фрагментації стандартів цифрового підпису та забезпечення автоматизованого обміну юридично значущими документами між контрагентами з різних юрисдикцій (Україна, ЄС, США).

Ціль проєкту – проєктування архітектури та розробка технічного прототипу SaaS-платформи, що включає моделювання бізнес-процесів обміну документами, створення дизайну інтерфейсів та формування плану реалізації для забезпечення інтеграції стандартів КЕП та eIDAS.

Наукова цінність отриманих результатів полягає в удосконаленні теоретико-методичних підходів до управління IT-проєктами в умовах мультиюрисдикційного середовища. Вперше розроблено концептуальну модель управління життєвим циклом транскордонного документа, яка базується на динамічному визначенні стандарту підпису (КЕП, eIDAS-QES, ESIGN) залежно від юрисдикції контрагента, що дозволяє автоматизувати вибір криптографічного алгоритму. Удосконалено метод оцінки часових параметрів проєкту шляхом адаптації методу PERT до специфіки розробки MVP-версій SaaS-продуктів із високими регуляторними ризиками, що дозволяє врахувати невизначеність інтеграції з державними шлюзами.

Кваліфікаційна робота складається з анотації, вступу, основної частини, яка включає чотири розділи, висновків та переліку використаних джерел.

Перший розділ охоплює дослідження предметної області та обґрунтування доцільності проєкту. Проведено аналіз ринку SaaS-рішень та нормативно-правової бази України, ЄС і США. Визначено проблему технічної фрагментації стандартів електронного підпису. Розглянуто теоретичні підходи до управління проєктами (зокрема стандарти ISO 21500, PMBOK) та специфіку управління SaaS-продуктами. Здійснено стратегічний аналіз зовнішнього та внутрішнього середовища з використанням методів PEST та SWOT [21].

Другий розділ містить математичну постановку задачі та моделювання інформаційних процесів. Виконано функціональну декомпозицію системи, розроблено концептуальну модель та формалізовано задачі управління ресурсами, часом і надійністю. Проведено аналіз вимог до системи та змодельовано сценарії взаємодії користувачів.

Третій розділ присвячено розробці інформаційного та програмного забезпечення. Здійснено проєктування концептуальної та логічної моделей бази даних. Обґрунтовано архітектурний підхід та вибір технологічного стеку. Описано реалізацію основних програмних модулів, користувацьких інтерфейсів та алгоритмів взаємодії через API. Розглянуто питання реалізації інфраструктури, контейнеризації та розгортання системи.

Четвертий розділ охоплює планування та управління проєктом. Розроблено організаційну структуру команди та матрицю розподілу відповідальності (RACI). Створено ієрархічну структуру робіт (WBS) та сформовано беклог продукту згідно з методологією Scrum. Розроблено календарний план реалізації та діаграму Ганта для окремих спринтів. Здійснено планування ресурсів і бюджету, проведено оцінку економічної ефективності та розроблено стратегію управління ризиками.

Робота завершується висновками щодо результатів дослідження та оцінкою досягнення поставленої мети.

Ключові слова: *управління проєктами, SaaS, електронний документообіг, КЕП, eIDAS, архітектура системи, WBS, бюджетування, управління ризиками.*

ЗМІСТ

ВСТУП

В умовах глобалізації економіки та стрімкої цифровізації бізнес-процесів електронний документообіг (ЕДО) стає критично важливим інструментом забезпечення ефективності діяльності підприємств. Особливої гостроти набуває проблема транскордонної взаємодії, коли компанії змушені оперувати в різних правових полях одночасно.

Для українського малого та середнього бізнесу (МСБ), що активно виходить на зовнішні ринки, ключовим викликом є технічна та юридична несумісність стандартів електронного підпису. З одного боку, національне законодавство вимагає використання Кваліфікованого електронного підпису (КЕП), з іншого – євроінтеграційний курс України диктує необхідність відповідності регламенту eIDAS (ЄС). Крім того, співпраця з американськими партнерами вимагає дотримання норм ESIGN/UETA. Наразі підприємства змушені підтримувати декілька розрізнених систем ЕДО, що призводить до фрагментації даних, дублювання витрат та затягування процесів укладання угод.

Варто зазначити, що в контексті набуття Україною статусу кандидата в члени ЄС та курсу на інтеграцію до Єдиного цифрового ринку, пріоритетним вектором розвитку є забезпечення безшовної взаємодії саме з європейською інфраструктурою довірчих послуг. Напрямок інтеграції з ринком США, хоч і є стратегічно важливим для масштабування бізнесу, на даному етапі розглядається як опціональний розширений функціонал, реалізація якого є бажаною, але вторинною по відношенню до критичної необхідності сумісності з eIDAS.

У цьому контексті виникає нагальна потреба у створенні уніфікованої SaaS-платформи, яка б виступала єдиним шлюзом для юридично значущого документообігу, автоматизуючи вибір необхідного стандарту підпису та забезпечуючи відповідність вимогам безпеки. Розробка проєкту створення такої

системи, що включає обґрунтування архітектури, математичне моделювання процесів та детальне планування впровадження, зумовлює актуальність даної магістерської роботи.

Дана кваліфікаційна робота є логічним продовженням та поглибленням наукового дослідження, розпочатого автором у курсовій роботі. У дипломному проєкті використано, суттєво доопрацьовано та розширено результати попереднього аналізу предметної області, що забезпечує послідовність та цілісність наукового пошуку.

Метою роботи є провести комплексний аналіз та дослідження процесів для створення впровадження SaaS-платформи «UniSign», яка забезпечує автоматизацію транскордонного обміну документами шляхом інтеграції стандартів КЕП (Україна) та eIDAS (ЄС) в єдиному інтерфейсі, що дозволить зменшити операційні витрати користувачів та скоротити час укладання міжнародних угод.

Ціллю проєкту є безпосередня розробка проєкту створення цієї платформи, що включає проєктування архітектури, формування плану управління, розрахунок бюджету та створення технічного прототипу системи.
Для досягнення поставленої мети вирішено такі завдання**:**

1. Здійснити аналіз ринку систем електронного документообігу та нормативно-правової бази (Україна, ЄС, США) для визначення вимог до мультиюрисдикційної платформи.
2. Побудувати концептуальну модель системи та виконати математичну формалізацію задач управління бюджетом, часом та надійністю проєкту.
3. Провести аналіз вимог та змоделювати сценарії взаємодії користувачів (Use Cases) та інформаційні потоки системи.
4. Розробити інформаційне та програмне забезпечення проєкту: спроєктувати базу даних, обґрунтувати архітектурний підхід та реалізувати прототип системи.

5. Розробити план управління проєктом, включаючи організаційну структуру, структурну декомпозицію робіт (WBS), календарний план та бюджет.
6. Оцінити економічну ефективність проєкту та розробити стратегію управління ризиками.

Об’єктом дослідження є процеси розробки та впровадження SaaS-платформи для автоматизації транскордонного електронного документообігу.

Предметом дослідження є методи та моделі управління часовими та вартісними параметрами IT-проєкту, зокрема: метод оцінки тривалості робіт в умовах невизначеності (PERT), інструменти структурної декомпозиції (WBS) та матричні моделі розподілу відповідальності (RACI) при розробці SaaS-платформ.

Методи дослідження. У роботі використано комплекс загальнонаукових та спеціальних методів:

- системний аналіз — для дослідження предметної області, побудови дерева проблем та цілей;
- метод порівняльного аналізу — для дослідження існуючих рішень на ринку та вибору технологічного стеку;
- математичне моделювання — для формалізації задач управління ресурсами та оцінки надійності системи;
- метод PERT (Program Evaluation and Review Technique) — для оцінки тривалості виконання робіт в умовах невизначеності;
- методи структурного аналізу та проєктування (SADT, UML, DFD) — для моделювання бізнес-процесів, сценаріїв використання та архітектури системи;
- метод структурної декомпозиції робіт (WBS) — для планування змісту проєкту;
- методи інвестиційного аналізу (ROI, NPV) — для оцінки економічної ефективності проєкту.

Наукова новизна отриманих результатів полягає у наступному:

- вперше розроблено концептуальну модель управління проєктом створення гібридної платформи документообігу, яка, на відміну від існуючих, базується на динамічному визначенні стандарту підпису (КЕП/QES) залежно від юрисдикції контрагента, приорітизуючи інтеграцію з єдиним цифровим ринком ЄС;
- удосконалено метод планування IT-проєктів шляхом адаптації матричної організаційної структури та методу PERT для умов розробки продуктів із високими регуляторними ризиками (Compliance);

Практичне значення отриманих результатів полягає у розробці повного пакету проєктної документації та створенні робочого прототипу платформи «UniSign». Запропоновані рішення дозволяють:

- зменшити час підписання міжнародних контрактів з декількох днів до годин;
- забезпечити технічну готовність українського бізнесу до роботи в умовах цифрового ринку ЄС.

Результати роботи, зокрема архітектурні рішення, схеми баз даних та план управління, доведені до рівня практичного використання і можуть бути впроваджені в діяльність IT-компаній, що розробляють LegalTech-рішення.

Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. Робота виконана відповідно до методичних рекомендацій [5]. Перший розділ присвячено дослідженню предметної області, аналізу ринку SaaS-рішень та нормативно-правової бази. Другий розділ містить математичне моделювання процесів, формалізацію задач управління та аналіз вимог до системи. У третьому розділі описано розробку інформаційного та програмного забезпечення, зокрема проєктування архітектури та бази даних. Четвертий розділ розкриває питання планування та управління проєктом, включаючи розробку організаційної структури, бюджетування та стратегії управління ризиками.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ ПРОЕКТУ

1.1. Аналіз предметної області та тенденцій ринку

Огляд нормативно-правової бази та стандартів

Розробка власне програмного забезпечення у сфері юридично значимого документообігу вимагає суворого дотримання правових норм юрисдикцій, де планується використання системи. Оскільки платформа unisign позиціонується як транскордонне рішення, необхідно проаналізувати вимоги до легітимності електронних підписів в Україні, Європейському Союзі та Сполучених Штатах Америки.

З того, що я бачив, правове регулювання електронного документообігу в україні побудовано на трьох рівнях. Системний закон № 851-IV фактично прирівнює електронний документ до паперового і зобов'язує зберігати його разом з аудиторським слідом протягом усього життєвого циклу, що одразу ж перетворює його на журнал [1]. А хеш-ланцюжок – невід'ємною частиною архітектури SaaS. Як на мене, 852-IV визначає класи підписів і вимагає онлайн-перевірку сертифікатів, тому кожна операція з підписом повинна супроводжуватися автоматичною перевіркою OCSP/CRL [1]. Підзаконні акти (ДСТУ 4145-2002 [4], наказ ДССЗЗІ № 1398) конкретизують допустимі алгоритми та носії ключів, а роз'яснення центрального засвідчувального органу від грудня 2024 року дозволяє вносити європейські сертифікати QES до українського реєстру без переекзаменування, відкриваючи прямий сценарій Україна–ЄС [1].

На рівні ЄС eIDAS 910/2014 запроваджує ієрархію SES/AdES/QES та основну ідею взаємного визнання. Запропонована версія eidas 2.0 інтегрує європейський цифровий ідентифікаційний гаманець, що потенційно скоротить час реєстрації користувачів UniSign вдвічі. Практичну ефективність транскордонних підписів підтверджує DocuSign: компанії, що працюють у

більш ніж 3 юрисдикціях ЄС, заощаджують близько 22 годин на кожній транзакції завдяки відсутності процедур конвертації.

У США чинний закон ESIGN Act та UETA прирівнюють електронний підпис до власноручного за умови, що сторони дійшли згоди [25]. І підписаний файл можна зберегти. На мою думку, чесно кажучи, для україно-американських угод це зводиться до відображення чекбоксу згоди в UI та забезпечення зберігання PDF/A копії в захищеному сховищі. Таким чином, нормативний аналіз підтверджує, що платформа повинна підтримувати довгострокове зберігання, онлайн-перевірку сертифікатів та гнучкий вибір схеми підпису в залежності від контрагентів, що закладено в технічних вимогах UniSign.

Окрім законодавчих актів, при проектуванні SaaS-платформи слід враховувати міжнародні стандарти інформаційної безпеки, що є запорукою довіри корпоративних клієнтів. Зокрема, архітектура системи повинна відповідати принципам стандарту ISO/IEC 27001:2022 "Системи управління інформаційною безпекою" [22]. Цей стандарт вимагає буквально впровадження оцінки ризиків, контролю доступу та процесів криптографічного захисту даних [28].

Оскільки платформа буквально взаємодіє з користувачами з Європейського Союзу та України, дотримання GDPR та Закону України № 2297-VI стає важливою вимогою [3]. Це безпосередньо впливає на архітектурні рішення: система повинна бути побудована за правилом Privacy by Design, закладаючи механізми конфіденційності ще на етапі проектування. До того ж, на рівні бази даних необхідно технічно забезпечити "право на забуття" - функціонал для повного. І безповоротного видалення особистої інформації [22].

Таким чином, дійсно нормативна база диктує чітку основу для технічного завдання: функціональний рівень обов'язково повинен підтримувати стандарти CAdES і PAdES. Тоді як нефункціональні параметри системи жорстко прив'язані до вимог безпеки та безперервного аудиту операцій [28].

Аналіз ринку SaaS-рішень та доцільність проєкту.

Для обґрунтування інвестиційної привабливості. та ринкової ніші проекту, необхідно провести детальний аналіз поточного стану світового та локального ринків систем електронного документообігу. Ринок в цілому характеризується високою динамікою зростання, зумовленою діджиталізацією бізнес-процесів, але має яскраво виражену сегментацію на глобальних та локальних гравців [33].

Світовий ринок онлайн підписів демонструє стрімке зростання: за даними Fortune Business Insights, його обсяг підскочить з приблизно $10,8 млрд у 2025 році до $118,9 млрд у 2032 році [26], що відповідає середньорічним темпам зростання понад 41%. Інше джерело оцінює середній показник CAGR у 32-33% і зазначає, що з кожним роком все більше компаній переносять важливі транзакції в хмару [33]. Власне, на динаміку впливає перехід бізнесу на віддалену співпрацю, необхідність прискорення циклів продажів, а також бажання урядів скоротити бюрократичні витрати [26].

Конкурентний ландшафт вже сформувався: у глобальному сегменті DocuSign обробляє майже половину всіх підписів (частка ≈ 49%) [26]. Особисто до першої п'ятірки входять adobe sign, dropbox sign та різні нішеві гравці. З чого, при цьому, більшість з великої четвірки в першу чергу орієнтовані на європейські eIDAS-QES [24] та американські UETA/ESIGN підписи [20], залишаючи осторонь локальні схеми у Східній Європі.

В Україні беззаперечним лідером хмарного документообігу є сервіс "Вчасно": він позиціонує себе як найпоширеніший інструмент, яким користуються понад мільйон компаній. На другому місці за популярністю - бухгалтерська платформа M.E.Doc, яка традиційно залишається великим гравцем завдяки глибокій інтеграції з податковою звітністю. На мою думку, обидва рішення працюють виключно з національним кваліфікованим електронним підписом (КЕП) [4]. І практично не підтримують транскордонні сценарії.

Ситуація може змінитися вже найближчим часом: у 2024-2025 роках Міністерство цифрової трансформації запустило підтримку європейських форматів електронного підпису (CAdES, XAdES, PAdES, ASiC-E), що спростить інтеграцію українських довірчих послуг з екосистемою eIDAS. В цей же час державна податкова служба підтвердила можливість використання міжнародних сервісів (в тому числі docusign) у зовнішньоекономічних контрактах за умови дотримання вимог Закону України "Про електронні документи та електронний документообіг" [2].

Таким чином, з'являється нова нішева можливість на перетині внутрішнього та зовнішнього попиту. Компаніям, які працюють в основному одночасно в Україні та ЄС або США, потрібен універсальний SaaS-інструмент, який дозволяє обирати схему підпису в залежності від юрисдикції: українська КЕП для транзакцій Україна–Україна, eIDAS-QES для взаємодії з європейськими контрагентами [24], а також, за необхідності, графічний або ESIGN-підпис для американського ринку [20]. На мою думку, буквальне розширення національної інфраструктури довірчих послуг в напрямку eIDAS значно спрощує технічну реалізацію такого сервісу, а зростання глобального ринку гарантує достатній попит [33]. Це відкриває двері для стартапу, який може швидко запустити MVP і зайняти незаповнену нішу між локальними і глобальними гравцями.

Окрім загальної динаміки, важливо враховувати конкурентне середовище та вхідні бар'єри, які формують стратегію запуску нового продукту.

Світовий ринок електронного підпису залишається концентрованим: DocuSign контролює 49% доходів, Adobe Sign - 14%, Dropbox Sign - 6% [26]. У той же час, на більш ніж сорок менших провайдерів припадає 31% доходів. І їхня сукупна частка зросла на 4% у 2022-2024 роках, що свідчить про попит на нішеві інтеграційні рішення. Лідер сегменту, DocuSign, генерує $2,73 млрд річного обороту з 92% валової маржі, а п'ята частина його бюджету витрачається на R&D. А їх партнерська мережа налічує понад 4 тисячі інтеграцій. Дійсно потужний мережевий ефект підвищує бар'єр входу для

новачків, але водночас відкриває простір для сервісів, які поєднують локальні QEP-процеси з міжнародними робочими процесами DocuSign.

У повністю державному секторі проникнення систем електронного документообігу вже сягає 80%: у Польщі їх використовують 83% органів місцевого самоврядування. Але лише 42% інтегровані зі шлюзом eIDAS [24]. Схожа картина - близько 40-50% інтеграції - спостерігається в Румунії та Словаччині, тому існує велика незадоволена потреба в рішеннях "останньої милі", які автоматизують транскордонні підписи.

Ширший сегмент "Програмне забезпечення для обробки документів" також демонструє зміщення: Microsoft займає 36%, Adobe - 21%. А 15% залишається "вільною" нішею [26]. Прогноз CAGR для е-підпису до 2032 року становить 32%, що перевищує 23% зростання EDMS загалом, тому вузькоспеціалізована стратегія e-Signature-SaaS видається дійсно розумною. Щодо цінового діапазону, то DocuSign пропонує тариф для малого бізнесу $3,6 тис. на рік, для підприємств - $179 тис., а локальний гравець Вчасно стягує близько $600. Власне, своєрідне позиціонування UniSign у сегменті $1-2 тис. на рік з мульти юрисдикційною підтримкою дозволяє йому зайняти "середню нішу" - дешевше, ніж docusign. Але більше, ніж локальні гравці.

Таким чином, аналіз ринку показує, що в сегменті транскордонного документообігу існує стійкий незадоволений попит. Особисто я вважаю розвиток платформи UniSign доцільним, оскільки вона закриває технологічний розрив між дорогими глобальними сервісами, які ігнорують локальну специфіку, та локальними рішеннями, які не підтримують міжнародні стандарти eIDAS. Обрана стратегія "середньої ніші" (вартість $1-2 тис. на рік) дозволяє конкурувати за рахунок найкращого співвідношення ціна-функціональність для експортоорієнтованого бізнесу [11].

1.2. Стратегічний аналіз середовища проєкту

Після загального огляду ринкових тенденцій та нормативної бази, доцільно провести детальний стратегічний аналіз зовнішнього та внутрішнього середовища проєкту для забезпечення його стійкості. Для цього було використано методи PEST- та SWOT-аналізу.

Аналіз макросередовища (PEST-аналіз)

Для визначення можливостей та загроз на макрорівні було проведено своєрідний PEST-аналіз [8]. Це дійсно інструмент, який дозволяє структурувати зовнішні фактори, які не піддаються прямому управлінню з боку керівництва проекту [11]. Але мають суттєвий вплив на його життєздатність. Таким чином, результати аналізу політичних, економічних, соціальних. та технологічних факторів узагальнюються нижче.

З власне політико-правових чинників найважливішим фактором, що визначає архітектуру платформи unisign, є активна гармонізація українського законодавства з цифровими стандартами Європейського Союзу. Зокрема, імплементація положень Регламенту ЄС про електронні довірчі послуги (№ 910/2014) в національну правову систему через Закон України "Про електронні довірчі послуги" створює передумови для взаємного визнання кваліфікованих електронних підписів (КЕП). Для проекту це означає необхідність впровадження механізму перевірки сертифікатів, виданих як українськими Акредитованими центрами сертифікації ключів, так і європейськими провайдерами довірчих послуг. Такоє європейськими постачальниками довірчих послуг. Власне, ще одним викликом є дотримання GDPR (загального регламенту захисту даних) при роботі з контрагентами з ЄС. Це вимагає впровадження механізмів псевдонімізації персональних даних на рівні бази даних та технічної можливості реалізації "права на забуття" (повне видалення даних користувача на його вимогу), що слід враховувати при розробці схем резервного копіювання.

Економічні фактори Аналіз економічного середовища свідчить про стабільне зростання попиту на SaaS-рішення (CAGR > 30% у сегменті LegalTech), що зумовлено прагненням бізнесу модернізувати операційні витрати (OpEx). І відмовитися від капітальних інвестицій (CapEx) у власну ІТ-інфраструктуру. При цьому проект реалізується в умовах високої волатильності валютного курсу. Відверто кажучи, оскільки продукт розробляється в Україні (витрати на оплату праці номіновані в гривні або прив'язані до курсу), а витрати на інфраструктуру (хостинг, ліцензії) і потенційний дохід номіновані в доларах США, фінансова модель проекту повинна враховувати курсові ризики. Зростання вартості хмарних сервісів вимагає від системних архітекторів застосування підходів FinOps для підвищення ефективності використання ресурсів.

Соціокультурні фактори Трансформація бізнес-культури, спричинена масовим переходом до гібридного формату роботи, створила значний попит на мобільні інструменти документообігу. Власне, рівень цифрової грамотності персоналу [16], який раніше був бар'єром для впровадження складних систем, тепер є драйвером зростання. Актуальним залишається бар'єр психологічної недовіри до хмарного зберігання юридично значущих документів. Це вимагає від розробників реалізації прозорих механізмів аудиту дій (Audit Trail), які дозволяють користувачеві перевірити цілісність документа. А також історію змін у будь-який момент часу. Інтерфейс системи (UX/UI) повинен бути максимально простим, щоб виключити страх помилки.

Технологічний ландшафт характеризується швидким старінням криптографічних стандартів. Гадаю, перехід індустрії на еліптичні криві та підготовка до постквантової криптографії диктує необхідність створення модульної архітектури, здатної швидко замінити криптографічне ядро без зупинки системи. На мою думку, певним позитивним фактором є розвиток технологій контейнеризації (docker, kubernetes), що дозволяє уніфікувати середовища розробки та продуктивної експлуатації [32]. Хоча зростання інтенсивності кіберзагроз (зокрема, DDoS-атак та фішингу) вимагає інтеграції

інструментів безпеки (WAF, обмеження швидкості) вже на етапі розробки MVP, що збільшує початкову трудомісткість проекту.

Результати аналізу впливу цих факторів на проект наведені в таблиці 1.1.

Таблиця 1.1

Група факторів	**Фактори впливу**	**Вплив на проєкт**
Політико-правові	1.Гармонізація законодавства України з eIDAS (ЄС). 2.Вимоги GDPR щодо захисту персональних даних. 3.Державна підтримка цифровізації (Дія.Підпис).	Позитивний: Спрощує вихід на ринок ЄС. Негативний: Вимагає додаткових витрат на юридичний аудит та сервери в ЄС.
Економічні	1. Зростання ринку SaaS (CAGR > 30%). 2.Валютна нестабільність (курс USD/UAH). 3.Здорожчання хмарних послуг.	Позитивний: Високий попит на дешеві рішення. Ризик: Зростання собівартості інфраструктури (AWS оплачується у валюті).
Соціокультурні	1. Звичка бізнесу до віддаленої роботи. 2. Зростання цифрової грамотності персоналу. 3. Недовіра до хмарного зберігання документів.	Позитивний: Зниження порогу входу для користувачів. Ризик:Необхідність інвестувати Audit Trail
Технологічні	1. Розвиток стандартів криптографії. 2.Поширення контейнеризації (Docker/K8s). 3. Зростання кіберзагроз (DDoS, фішинг).	Позитивний: Можливість швидкого розгортання MVP. Ризик: Високі вимоги до архітектури безпеки (WAF, шифрування).

SWOT-аналіз проєкту створення платформи UniSign

На основі результатів PEST-аналізу та дослідження внутрішніх ресурсів стартапу, доцільно провести SWOT-аналіз. Власне, цей інструмент дозволяє систематизувати сильні та слабкі сторони проекту. Результати цього аналізу

ляжуть в основу вибору стратегії управління проектом, в тому числі ризиками. І управління стейкхолдерами.

Таблиця 1.2

Матриця SWOT-аналізу для проекту UniSign

Сильні сторони (Strengths)	**Слабкі сторони (Weaknesses)**
1. Унікальна торгова пропозиція (USP): Мультиюрисдикційність (підтримка КЕП України, eIDAS ЄС та ESIGN США) в одному інтерфейсі, що вирішує проблему фрагментації. 2. Сучасний технологічний стек: Використання FastAPI та асинхронної архітектури забезпечує високу продуктивність (RPS) при менших витратах на інфраструктуру порівняно з легасі-системами конкурентів. 3. Низька собівартість розробки: Використання PaaS-рішень та невеликої cross-functional команди дозволяє вкластися в бюджет 160 тис. USD, що значно нижче за бюджети R&D великих гравців. 4. Гнучкість розгортання: Архітектура "модульного моноліту" на старті дозволяє швидко адаптувати продукт під зміни вимог без складнощів мікросервісної оркестрації. 5. Відповідність нормативам: Вбудована підтримка вимог Закону України № 851-IV та GDPR "з коробки".	1. Обмежені фінансові ресурси: Фіксований бюджет на MVP створює ризики зупинки проєкту у випадку непередбачуваних витрат (наприклад, зростання цін на хмарні послуги). 2. Мала команда (Bus Factor): Команда з 6 осіб створює ризик залежності від ключових розробників. Хвороба або звільнення одного члена команди може суттєво загальмувати спринт. 3. Відсутність впізнаваності бренду: На етапі запуску довіра до UniSign буде значно нижчою, ніж до DocuSign або "Вчасно", що критично для сервісу, який працює з юридично значущими документами. 4. Залежність від третіх сторін: Робота платформи залежить від стабільності зовнішніх центрів сертифікації (ЦСК) та OCSP-серверів, на які команда проєкту не має прямого впливу.
Можливості (Opportunities)	**Загрози (Threats)**

Продовження табл. 1.2.

1. Євроінтеграція цифрового простору: Визнання українських довірчих послуг в ЄС та імплементація eIDAS 2.0 відкривають прямий доступ до європейського ринку SMB. 2. Зростання попиту на віддалену роботу: Глобальний тренд на цифровізацію документообігу забезпечує стабільне зростання ринку на рівні 32% CAGR. 3. Висока вартість конкурентів: Валютна прив'язка та високі тарифи глобальних гравців (DocuSign, Adobe) роблять цінову пропозицію UniSign привабливою для українського бізнесу. 4. Партнерські інтеграції: Можливість вбудовування UniSign у популярні CRM/ERP системи через відкритий API як додаткового каналу продажів.	1. Зміни у законодавстві: Раптові зміни в регуляторній політиці щодо КЕП або захисту даних можуть вимагати термінової переробки архітектури. 2. Кіберзагрози: Високий ризик цілеспрямованих атак на платформу, оскільки вона зберігає комерційні та юридичні дані. 3. Агресивна реакція конкурентів: Локальні гравці (наприклад, "Вчасно" чи "M.E.Doc") можуть швидко скопіювати функціонал мультиюрисдикційності, використовуючи свою наявну клієнтську базу. 4. Технологічні збої: Відмови на стороні хмарного провайдера (AWS/DigitalOcean) можуть призвести до порушення SLA та фінансових санкцій з боку клієнтів.

Аналіз сильних сторін: Ключовою перевагою проекту є його архітектурна та концептуальна гнучкість. На відміну від великих корпоративних систем, UniSign побудований з нуля з сучасною продуктивністю (fastapi). І з урахуванням вимог UX. Це забезпечує "безшовний" користувацький досвід, де перемикання між стандартами підпису (QES/QES) є непомітним. З точки зору управління проектами, компактна команда є сильною стороною. Я думаю, що саме це мінімізує комунікаційний шум та бюрократію, дозволяючи впроваджувати запити на зміни в один спринт, що неможливо у великих ієрархічних структурах.

Аналіз слабких сторін: Найважливішим вразливим місцем проекту є обмеженість ресурсів. Бюджет у 160 тисяч доларів США достатній для MVP,

але не залишає місця для маркетингових маневрів або довготривалих R&D у випадку архітектурних помилок. З управлінської точки зору, слабким місцем є відсутність дублювання компетенцій в команді. Це вимагає від менеджера проекту (PM) жорсткого контролю за розподілом знань. І кодової документації для мінімізації ризиків плинності кадрів. Ще одним слабким місцем, на мій погляд, є необхідність конкурувати на "довірі" - новій платформі складніше переконати юристів великих компаній зберігати документи у них.

Аналіз можливостей: Зовнішнє середовище наразі є сприятливим для проекту. Курс України на створення єдиного цифрового ринку з ЄС створює "блакитний океан" для посередницьких послуг, які можуть технічно забезпечити транскордонну сумісність підписів. Для проекту це означає можливість швидкого масштабування після успішного запуску MVP. Насправді, крім того, економічна ситуація змушує бізнес оптимізувати витрати, тому рішення, яке дозволяє відмовитися від трьох різних підписок на користь однієї (UniSign), має високий потенціал для проникнення на ринок.

Аналіз загроз: Домінування ризиків кібербезпеки є специфічним для цієї предметної області. Будь-які дані, по суті, витікають. Або компрометація ключа може призвести до миттєвого закриття проекту через репутаційні втрати. Це вимагає чесної інтеграції процесів безпеки (DevSecOps) безпосередньо в життєвий цикл розробки, що може сповільнити час виходу на ринок. Думаю, правова невизначеність також є значною загрозою: технічні стандарти для підписів змінюються. І проект повинен мати архітектуру, яка може адаптуватися до нових криптографічних бібліотек без повного перепроектування ядра.

Стратегічні ініціативи на основі SWOT-аналізу

На перетині визначених факторів ми сформулюємо основні стратегії управління проектами:

1. Стратегія SO (Сила + Можливості): Використовувати сучасний стек технологій (Python/FastAPI) для швидкого застосування нових європейських стандартів (eIDAS 2.0), випереджаючи неповоротких

конкурентів. Це фактично дозволить нам зайняти нішу транскордонного документообігу до того, як туди увійдуть великі гравці.

2. Стратегія WO (Weakness quite + Opportunity): Компенсувати недостатню впізнаваність бренду за рахунок агресивного ціноутворення для сегменту СМБ, який шукає шляхи скорочення витрат. Власне, використовувати партнерські APIs для швидкого охоплення клієнтів існуючих CRM-систем, не витрачаючи обмежений бюджет на прямий маркетинг.

1. Стратегія ST (Strength + Threats): Протидіяти кіберзагрозам за допомогою архітектури "безпеки за задумом". Використання перевірених криптографічних бібліотек та ізоляція ізоляція даних клієнтів дозволить мінімізувати ризики зламів. До того ж, гнучкість коду дозволить нам швидко реагувати на зміни в законодавстві.

2. Стратегія WT (Слабкість + Загрози): Мінімізувати ризик "малої команди" та технологічних збоїв, налаштувати CI/CD та впроваджувати стандарти автоматизованого тестування. Це зменшить залежність від людського фактору. І зменшить ймовірність виробничих помилок, які можуть бути фатальними для репутації стартапу.

Таким чином, аналіз підтверджує життєздатність проекту, але вказує на необхідність зміщення фокусу управління з "чистої розробки" на забезпечення безпеки та дотримання стандартів (Compliance) в умовах обмежених ресурсів.

1.3. Побудова дерева проблем та дерева цілей проєкту

Сучасні компанії, що працюють у мультиюрисдикційному середовищі, змушені підтримувати декілька розрізнених систем електронного документообігу (ЕДО). Зокрема, для взаємодії з українськими контрагентами необхідний Кваліфікований електронний підпис (КЕП), для співпраці з партнерами з ЄС — підпис, сумісний з регламентом eIDAS (рівень QES), а для роботи з ринком США — рішення, що відповідають актам UETA/ESIGN. Така фрагментація призводить до значних операційних труднощів: документи розпорошені між різними сервісами, API систем часто несумісні, контроль

версій та доступу ускладнений, а витрати на ліцензійні платежі зростають кратно. Ситуація ускладнюється тим, що кожен провайдер послуг підпадає під окремі регуляторні режими (Закон України № 851-IV, eIDAS в Європі, GDPR щодо захисту даних, HIPAA для специфічних контрактів у США тощо).

Для систематизації цих викликів застосовано метод побудови структурної моделі проблем, що є стандартною практикою фази ситуаційного аналізу згідно з логіко-структурним підходом (Logical Framework Approach — LFA) [6, 7]. Цей метод передбачає візуалізацію негативної ситуації для подальшої її трансформації у структурну модель цілей.

Щоб структурувати ситуацію та визначити кореневі причини неефективності, у центрі дерева сформульовано ключову проблему: необхідність підтримувати декілька ЕДО-систем для різних контрагентів. Нижче наведено схему, яка демонструє причинно-наслідкові зв’язки та обґрунтовує вимоги до майбутнього ІТ-проєкту.

Рис. 1.1. Структурна модель проблем ІТ-проєкту

Варто зазначити, що імплементація нових європейських форматів електронного підпису (CAdES, ASiC) в українській інфраструктурі довірчих

послуг частково пом'якшує проблему технічної несумісності. Проте бізнесу все ще бракує єдиної SaaS-платформи, здатної автоматизувати вибір необхідної схеми підпису для сценаріїв Україна↔Україна, Україна–ЄС, Україна–США та вести централізований журнал аудиту подій. Запропонований проєкт спрямований саме на вирішення цієї комплексної проблеми інтеграції.

З огляду на виявлені проблеми, існує об'єктивна потреба у створенні уніфікованого інструменту, здатного усунути бар'єри сумісності між різними стандартами електронного підпису. На основі побудованої моделі проблем та для їх вирішення розроблено структурну модель цілей проєкту, яка дозволяє трансформувати виявлені негативні стани у позитивні результати.

Для формалізації очікуваних результатів проєкту застосовано метод побудови дерева цілей. Цей інструмент дозволяє декомпозувати головну мету створення та впровадження SaaS-платформи «UniSign» на підпорядковані тактичні та операційні цілі, досягнення яких гарантує успіх проєкту.

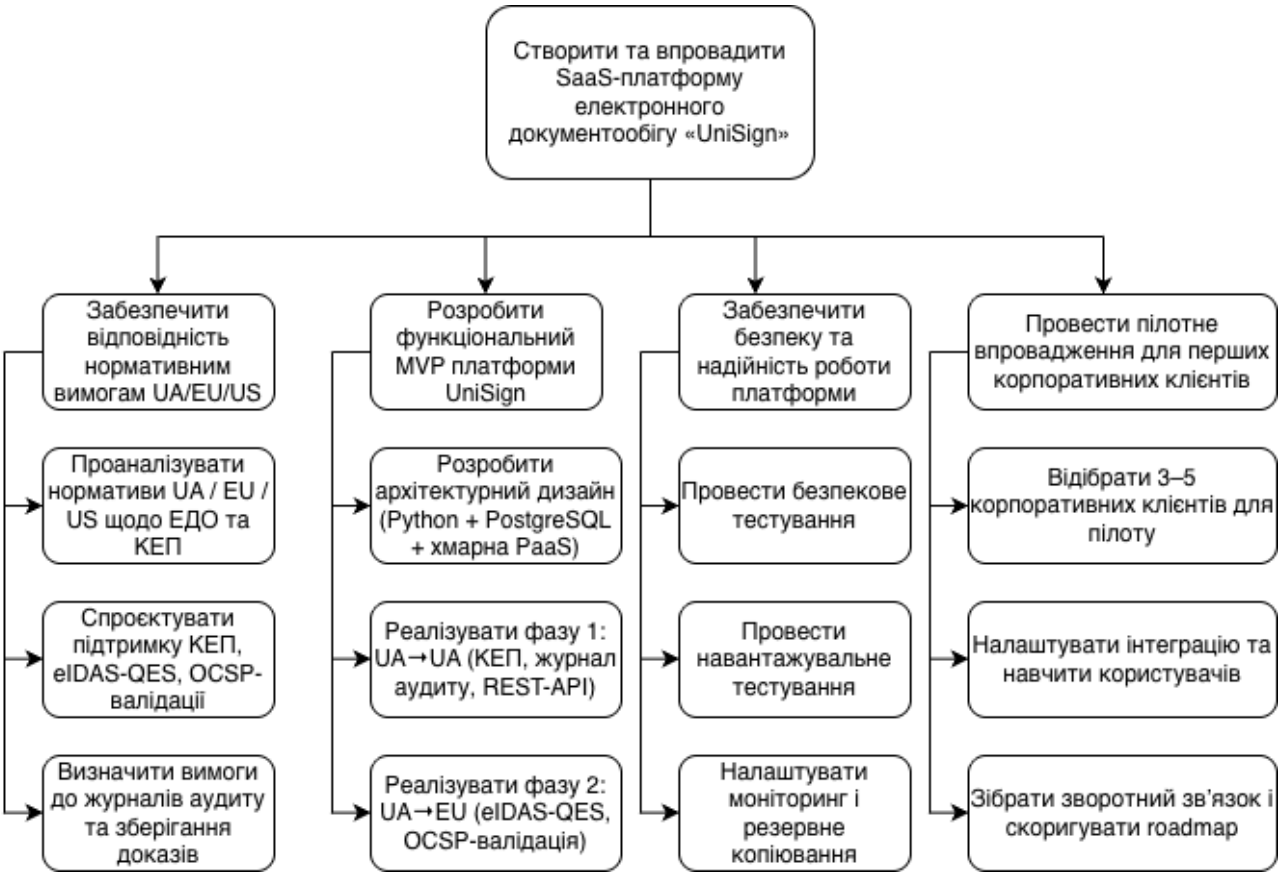


Рис. 1.2. Структурна модель цілей

Кожна гілка дерева деталізує стратегічні наміри до рівня конкретних дій, таких як аналіз нормативів, проєктування архітектури, проведення навантажувального тестування та налаштування інтеграцій для перших клієнтів. Такий підхід дозволяє чітко визначити критерії успішності для кожного етапу робіт, забезпечуючи узгодженість між бізнес-вимогами та технічними рішеннями.

1.4. Формулювання технічного завдання на розробку у вигляді паспорту проєкту

На основі проведеного аналізу проблем та сформованого дерева цілей було розроблено Паспорт проєкту. Цей документ виступає первинним технічним завданням (ТЗ) на розробку, фіксуючи ключові параметри, обмеження та критерії успішності майбутньої системи ще до початку етапу проєктування. У ньому закріплено домовленості між командою виконавців та стейкхолдерами (інвесторами) щодо строків, бюджету та очікуваного функціоналу MVP.

Основні параметри проєкту «UniSign» наведено в Таблиці 1.3.

Таблиця 1.3

Паспорт IT-проєкту

Назва проєкту	**SaaS-платформа «UniSign»**
Замовник / фінансування	Стартап-засновники, власний капітал
Керівник проєкту	Кузьменко О. С.
Команда	6 осіб: 2 backend, 1 frontend, 1 DevOps, 1 QA, 1 PM
Період виконання	01 вересня 2025 – 28 лютого 2026

Продовження табл. 1.3.

Бюджет MVP	≈ 160 000 USD (R&D 128k; PaaS 20k; юридичні 12k)
Цільова аудиторія	UA / CEE SMB із міжнародними контрактами
Ключові функції	КЕП; eIDAS-QES; ESIGN; REST-API; журнал аудиту
Очікувані вигоди	−50 % витрат на ліцензії; цикл угоди ≈ 2 год
Ключові ризики	Складнощі інтеграції з українськими ЦСК та eIDAS
Показники успіху	SLA ≥ 99 %; ≥ 3 корп. клієнти; ≥ 10k doc / 3 міс

Формалізація ключових параметрів у вигляді паспорту дозволяє мінімізувати ризики неконтрольованого розширення змісту та забезпечує єдине бачення кінцевого результату для всіх зацікавлених сторін. Затверджений паспорт слугує точкою відліку для подальшого детального планування, вибору методології управління та розробки архітектури системи, що буде розглянуто в наступних розділах роботи.

1.5. Формування наукової новизни та інноваційності IT-проєкту

Розвиток платформи UniSign спрямований не тільки на вирішення прикладних бізнес-завдань. Але й на створення нової науково-методичної бази для управління проектами у сфері LegalTech. Наукова новизна отриманих результатів полягає в удосконаленні методів і моделей управління it-проектами, що реалізуються в умовах жорстких нормативних обмежень та мультиюрисдикційному середовищі [9].

Наукова новизна результатів дослідження конкретизується в наступних положеннях:

1. Вперше розроблено концептуальну модель управління життєвим циклом транскордонного електронного документа, яка, на відміну від існуючих, базується на динамічному визначенні необхідного стандарту підпису (QES, QES або ESIGN) в залежності від юрисдикції контрагента. Це, на мою думку, дозволяє автоматизувати процес вибору криптографічного алгоритму та забезпечити юридичну силу документа без участі оператора.
2. Удосконалено метод оцінки часових параметрів IT-проекту шляхом адаптації методу PERT (Program Evaluation. And Review Technique) до специфіки розробки MVP-версій SaaS-продуктів. Запропонований чесний метод враховує високу невизначеність технологічних ризиків на етапі інтеграції з державними шлюзами (ЦСК/eIDAS), що дозволяє підвищити точність планування релізів на 15-20% [17].
3. Набуло подальшого розвитку використання методу аналізу ієрархій (MAI) для обґрунтування вибору технологічного стеку в проектах з високими вимогами до інформаційної безпеки. Побудована модель враховує не тільки технічні метрики (продуктивність, масштабованість). Але й специфічні критерії відповідності локальним стандартам криптографічного захисту інформації (КСЗІ).

Інноваційність проекту полягає у створенні унікального архітектурного рішення, що забезпечує безшовну інтеграцію різнорідних стандартів електронного підпису в єдиному інтерфейсі.

Ключові аспекти інновацій:

- Технологічне нововведення: Реалізація механізму "відокремленої агрегації підписів", який дозволяє зберігати та перевіряти кілька різних типів цифрових підписів на одному документі без порушення його цілісності.
- Інноваційність продукту: Реалізація буквально моделі Compliance-as-a-Service, де платформа бере на себе відповідальність за

перевірку дійсності сертифікатів контрагентів у режимі реального часу за допомогою протоколу OCSP, знижуючи юридичні ризики для клієнтів.

- Інновації в управлінні: застосування гібридної моделі управління (поєднання вимог до документації PMBOK з гнучкістю Scrum), адаптованої для розподілених команд, що працюють в галузі RegTech.

Практична цінність роботи підтверджується створенням робочого прототипу системи, який демонструє можливість скорочення часу обробки міжнародних контрактів з 3-5 днів до декількох годин. А також розробленим пакетом проектної документації (паспорт, WBS, план ризиків), який може бути використаний як шаблон для подібних стартапів.

Стандартизація власне управлінських процесів за ISO 21500 та PMBOK

Розглядаючи теоретичні основи менеджменту, неможливо оминути увагою модель процесів, що регламентується міжнародним стандартом ISO 21500:2012 "Керівництво з управління проектами" [21]. На мою думку, цей стандарт уніфікує понятійний апарат. І визначає п'ять ключових груп процесів, які є актуальними для будь-якого ІТ-проєкту, в тому числі і для створення SaaS-платформи:

1. Ініціація: Буквальне визначення зацікавлених сторін, затвердження статуту проекту. І призначення керівника проекту. На цьому етапі для стартап-проекту дуже важливо встановити початкові вимоги. І бюджетні обмеження.
2. Планування: Дійсно найбільш трудомістка група процесів, яка включає структуру розбиття робіт (WBS) [39], розробку розкладу, оцінку ресурсів. І планування якості. Особисто для мене, в контексті гнучких методологій, планування не є одноразовою діяльністю, а відбувається ітеративно перед кожним спринтом [38].

3. Виконання: Безпосередня координація в буквальному сенсі людських і матеріальних ресурсів для створення продукту. Чесно кажучи, в it-проектах ця фаза тісно пов'язана з розробкою технічних практик.
4. Моніторинг і контроль: Відстеження прогресу, порівняння плану з реальністю та управління змінами. Чесно кажучи, для SaaS-проектів важлива швидкість роботи команди І метрики якості коду відіграють тут ключову роль.
5. Закриття: Формальне завершення етапу або проекту, передача результатів клієнту. І аналіз отриманих уроків.

На відміну від ISO, PMBOK (7-е видання) пропонує більш гнучкий підхід, зосереджуючись на восьми сферах діяльності [19]. Від чого, власне, і залежить шлях розвитку. І домен життєвого циклу заслуговує на особливу увагу в контексті розвитку платформи unisign. Я думаю, що це підкреслює необхідність адаптувати техніку в залежності від типу продукту. Власне, оскільки SaaS-платформа вимагає частих оновлень і високої надійності, рекомендується застосовувати модель доставки інкрементної цінності, коли базовий функціонал (MVP) стає доступним користувачам у найкоротші терміни, а додаткові модулі (інтеграції, AI-аналітика) додаються в наступних релізах [10].

Специфіка управління SaaS-проєктами: Product vs Project Mindset

Управління створенням хмарних рішень (SaaS) має головну відмінність від класичної проектної діяльності - це перехід від "проектного мислення" до "продуктового". Насправді, якщо проект має чіткий початок, то життєвий цикл SaaS-продукту є безперервним. Відверто кажучи, це вимагає від керівника проекту врахування наступних специфічних аспектів:

- Технічний борг повністю під контролем менеджменту: У гонитві за швидким виходом на ринок команди часто обирають неоптимальні

технічні рішення. Власне, завдання управління проектом полягає в тому, щоб збалансувати швидкість розробки нових функцій з рефакторингом коду. Ігнорування цього аспекту призводить до експоненціального зростання вартості внесення змін у майбутньому (з мого досвіду).

- Безперервна інтеграція та доставка (CI/CD): Управління релізами перестає бути адміністративною процедурою і стає частиною інженерної культури. Делівері менеджер не повинен планувати дату релізу. Замість цього він повинен створити автоматизовані процеси, які дозволять розгортати оновлення в різний час доби, не зупиняючи роботу сервісу.
- Чесно зосередитись на метриках продукту: Успіх IT-проекту вимірюється не лише дотриманням трикутника "час-бюджет-контент", але й бізнес-показниками: вартість залучення клієнтів (CAC), життєва цінність клієнта (LTV) та рівень відтоку. Ці показники мають бути інтегровані в систему прийняття управлінських рішень.

Інструментальне забезпечення управління IT-проєктом

Ефективне застосування методологій неможливе без відповідних програмних інструментів. Щоб дійсно реалізувати гібридну модель управління в проекті UniSign, доцільно використовувати інтегровану екосистему інструментів:

- Системи відстеження завдань: Наприклад, Jira Software, дозволяє візуалізувати робочий процес на Скрам або Канбан дошках, управляти беклогом, відстежувати залежності між завданнями, автоматично генерувати звіти (діаграми, контрольні діаграми) [27]. Це забезпечує прозорість роботи розподіленої команди.
- Бази знань: Наприклад, Confluence. Використовується для зберігання технічної документації, вимог до продукту, протоколів зустрічей та описів архітектури. Інтеграція на кшталт Jira. А Confluence дозволяє створити єдиний інформаційний простір проекту, забезпечуючи простежуваність вимог від ідеї до реалізації.

- Інструменти для комунікації: Корпоративні месенджери (Slack, Microsoft Teams) стають центральним центром оперативної взаємодії, інтегруючись з системами моніторингу та CI/CD для миттєвого сповіщення про стан збірки або про інциденти на сервері [31].

Отже, теоретико-методологічна база управління IT-проєктом є складним синтезом класичних стандартів, гнучких фреймворків та інженерних практик. Для проєкту UniSign обрана стратегія базується на гібридній моделі: жорстке бюджетування та управління ризиками за стандартами PMBOK поєднується з ітеративною розробкою за Scrum (детальний опис та обґрунтування застосування цієї методології наведено у підрозділі 4.1), що підкріплюється сучасним інструментарієм автоматизації процесів [19].

1.6. Висновок до першого розділу

У першому розділі здійснено комплексний аналіз предметної області та формалізовано процедуру ініціації проєкту створення SaaS-платформи «UniSign». Дослідження ринку та нормативної бази (КЕП, eIDAS, ESIGN) підтвердило критичну проблему фрагментації стандартів підпису, що стримує експортну діяльність українського бізнесу . За результатами стратегічного аналізу середовища (PEST, SWOT) обґрунтовано економічну доцільність реалізації проєкту та обрано конкурентну стратегію. Використання логіко-структурного підходу дозволило побудувати дерево проблем і трансформувати його в дерево цілей, визначивши ключовим завданням автоматизацію вибору схеми підпису. Фінальні параметри, бюджет MVP та технічні обмеження системи були затверджені у розробленому Паспорті проєкту, який слугує базовим технічним завданням для подальшого проєктування.

РОЗДІЛ 2. РОЗРОБКА МАТЕМАТИЧНОЇ МОДЕЛІ

Повна розробка сучасної SaaS-платформи для транскордонного електронного документообігу - це складна інженерна задача. І технічне завдання, яке вимагає не тільки підбору сучасного технологічного стеку, а й глибокого попереднього моделювання. У першому розділі, було визначено, що ключовою проблемою в предметній області є фрагментарність процесів підписання документів через відмінності в законодавчих базах України, ЄС та Сполучених Штатів Америки. На мою думку, для вирішення цієї проблеми недостатньо просто створити веб-інтерфейс, необхідно побудувати надійну математичну та інформаційну модель, яка гарантуватиме цілісність даних, юридичну значимість транзакцій та стійкість системи при високих навантаженнях.

Мета цього розділу - формалізувати завдання створення платформи UniSign. Для цього ми перейдемо від бізнес-вимог до чітких математичних абстракцій. А також до інформаційних моделей. Буде побудовано концептуальну модель бізнес-процесів "як є" та "як буде" розроблено математичний апарат опису станів електронного документу, спроектовано структуру інформаційного забезпечення, яка стане основою для програмної реалізації. Такий метод дійсно дозволить мінімізувати архітектурні ризики на етапі проектування та забезпечує наукове обґрунтування технічних рішень.

2.1. Концептуальне моделювання бізнес-процесів

Побудова концептуальної моделі - це перший крок у проектуванні складної інформаційної системи. Вона дозволяє абстрагуватися від технічних деталей реалізації (таких як мови програмування чи сервери) і зосередитися на логіці інформаційних потоків та взаємодії об'єктів. Для платформи UniSign важливо змоделювати процес перетворення неструктурованого файлу в юридично великий документ [18].

Концептуальна схема, що відображає функціональні модулі платформи та їх взаємодію із зовнішнім середовищем, наведена в підрозділі 2.1.4 (Рис. 2.1).

2.1.1 Аналіз поточного стану процесів («AS-IS»)

Сьогодні процес підписання міжнародного контракту в сегменті МСБ характеризується високим рівнем ентропії. І ручним управлінням. Типовий сценарій виглядає так: менеджер створює документ, підписує його за допомогою локальної КЕП (наприклад, через Дія, Вчасно або M.E.Doc), потім надсилає файл електронною поштою партнеру в ЄС. Партнер, не маючи можливості перевірити український підпис у своїй звичній системі (наприклад, DocuSign), змушений або шукати спеціалізовані сервіси перевірки. Або запитувати паперову версію.

Така організація процесу має низку суттєвих недоліків, які можна формалізувати як вразливості системи:

1. Розрив ланцюга довіри: Відсутність єдиного середовища, де зберігаються і документ, і метадані підпису, унеможливлює автоматизований аудит.
2. Ризики для безпеки: передача конфіденційних файлів через незахищені канали (електронна пошта, месенджери) створює загрозу перехоплення даних – атаки "людина посередині" (Man-in-the-Middle).
3. Тимчасові затримки: Необхідність конвертації форматів та ручна перевірка підписів збільшує час закриття угоди в середньому на 2-3 дні.

2.1.2 Концептуальна модель пропонованого рішення («TO-BE»)

Щоб частково усунути виявлені недоліки, пропонується впровадити платформу UniSign, яка виступає як єдина довірена третя сторона між учасниками транзакції. Концептуально цю систему можна представити як "чорний ящик", який отримує на вхід проект документа та ідентифікатори підписувачів, а на виході генерує криптографічний контейнер (PAdES/CAdES), що містить сам документ, накладені підписи та мітки часу.

Концептуальна модель виділяє три ключові класи суб'єктів (акторів), взаємодія яких визначає функціональний контур системи:

1. Ініціатор: Користувач, який повністю володіє документом та ініціює процес підписання. Їхня ключова потреба - це контроль над процесом та даними. В моделі системи ініціатор відповідає за завантаження файлу, визначення маршруту підписання (послідовний або паралельний) та вибір необхідного рівня підпису (КЕП, ЕЦП або простий електронний підпис) для кожного контрагента.
2. Контрагент (отримувач): Зовнішній користувач, який отримує доступ до документа через тимчасове захищене посилання. Особисто для мене основною вимогою до цього актора є мінімізація бар'єрів. Гарна система повинна дозволяти йому переглянути документ. І підписати його без необхідності повної реєстрації або встановлення складного програмного забезпечення, використовуючи хмарні ключі або BankID.
3. Компанія, яка фактично є адміністратором (Tenant Admin): особа, відповідальна за управління доступом в організації. Тобто, на концептуальному рівні, вони керують політиками безпеки і мають доступ до розширених журналів аудиту. А також до білінгової інформації.

2.1.3 Інформаційні потоки в системі

Концептуальна схема обробки даних у системі UniSign базується на принципі конвеєра. Особисто вхідний інформаційний об'єкт (файл) проходить ряд станів, кожен з яких збагачується додатковими метаданими.

- На етапі Ingestion (Прийом) система генерує унікальний хеш-відбиток документа, який стає його постійним ідентифікатором. Особисто для мене це унеможливлює заміну файлу на наступних етапах.
- Етап Обробки (Processing) передбачає взаємодію із зовнішніми довірчими службами (ЦСК, OCSP) для перевірки дійсності сертифікатів користувачів у режимі реального часу.

- На етапі Completion (Завершення) формується фінальний архів, який включає не тільки підписаний файл, але й аудиторський слід, захищений від модифікацій журнал, який фіксує хронологію всіх дій над документом (хто відкривав або підписував файл, коли, з якої IP-адреси).

Такий концептуальний дизайн дозволяє вирішити проблему мультиюрисдикційності: система абстрагує користувача від складності криптографічних операцій, надаючи уніфікований інтерфейс для роботи з різними стандартами підпису.

2.1.4 Функціональна декомпозиція системи

Щоб перевірити принципову цілісність концептуальної моделі та забезпечити можливість подальшого масштабування, необхідно виконати функціональну декомпозицію платформи UniSign. Система не розглядається як монолітний об'єкт, а проєктується як набір взаємопов'язаних логічних підсистем, кожна з яких відповідає за певний, чітко відокремлений аспект обробки інформації. Такий підхід дозволяє реалізувати архітектурні принципи слабкої зв'язаності (Low Coupling) та високої згуртованості (High Cohesion), що є критично важливим для забезпечення надійності SaaS-рішень.

Важливо зазначити, що платформа UniSign не функціонує ізольовано. Як буде показано на схемі, вона є інтегрованим елементом складної ієрархічної структури. На верхньому рівні знаходиться надсистема, яка диктує правила гри: це глобальна цифрова економіка, європейська екосистема eIDAS, національна інфраструктура електронних довірчих послуг України (ЦЗО/КНЕДП) та міжнародні стандарти безпеки (ISO, GDPR). Саме ці фактори визначають функціональні вимоги до платформи.

Запропонована декомпозиція візуалізує межі системи, вхідні та вихідні інформаційні потоки, а також групування внутрішніх сервісів навколо ключових бізнес-процесів: управління ідентичністю, криптографічні операції, документообіг та аналітика. Графічне представлення функціональних модулів у контексті їхнього оточення наведено на рисунку нижче.

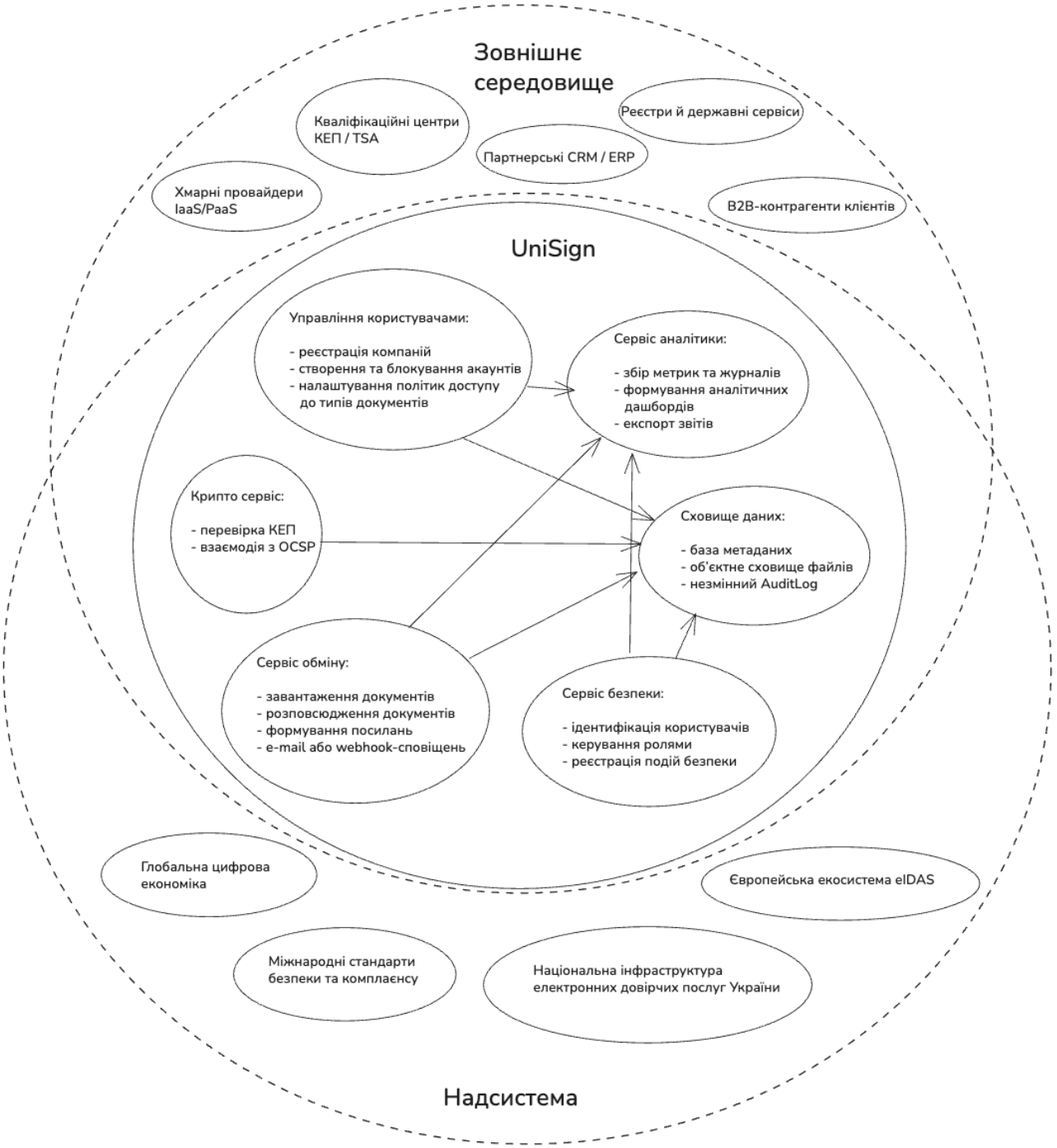


Рис. 2.1. Концептуальна схема функціональних модулів та оточення системи UniSign

Підсистема керування ідентичністю та доступом

Цей модуль відповідає за початкову ідентифікацію користувачів та розмежування прав доступу. Враховуючи транскордонний характер платформи, підсистема повинна підтримувати гібридну модель автентифікації:

- Для резидентів України: інтеграція з BankID, Дія.Підпис та більшістю локальних ЦСК.
- Для резидентів ЄС: фактична підтримка засобів ідентифікації, сумісних з eIDAS.
- Для американських користувачів: стандартна двофакторна автентифікація (2FA) через електронну пошту/SMS. На концептуальному рівні цей модуль діє (особисто для мене) як "фільтр" який дозволяє тільки перевіреним суб'єктам отримувати доступ до внутрішніх процесів, мінімізуючи ризик несанкціонованого доступу до комерційної таємниці.

Підсистема документообігу

Це ядро платформи, що керує станами документів. На відміну від простих файлових сховищ, цей модуль реалізує логіку кінцевого автомата. Він відповідає за:

- Перевірка форматів вхідних файлів (перевірка на відповідність стандартам PDF/A для довгострокового зберігання).
- Керуйте маршрутизацією: визначте порядок підписання (послідовний чи паралельний).
- Контроль версій: якщо основний документ відхилено та створюється нова версія, система повинна зберігати зв'язок між версіями, формуючи деревоподібну структуру історії документа (що досить корисно).

Криптографічний шлюз

Ця підсистема є ключовим нововведенням проекту. Вона абстрагується від складнощів роботи з різними криптографічними бібліотеками. Тобто, на концептуальному рівні, шлюз виступає в ролі "адаптера":

- Отримує запит на підпис.
- Визначає необхідний стандарт (CAdES-X Long для України, PAdES-B-LTA для ЄС).
- Генерує хеш документа (SHA-256 або ГОСТ 34.311)

- Взаємодіє в основному з клієнтським середовищем (браузером) для накладення приватного ключа без передачі самого ключа на сервер. Це повністю забезпечує дотримання принципу "zero trust".

2.1.5 Взаємодія із зовнішнім середовищем

Концептуальна модель була б неповною без опису інформаційних потоків між системою UniSign та зовнішніми сервісами, від яких залежить її функціонування.

Взаємодія з Центрами сертифікації

Система не зберігає списки відкликаних сертифікатів (CRL) локально. Це створює ризик використання застарілих даних. Замість цього реалізована модель, що працює в режимі реального часу: при кожному підписі або спроби перевірки підпису система надсилає OCSP-запит до відповідного кваліфікованого постачальника електронних довірчих послуг (КНЕДП).

- Вхідний потік: Статус сертифіката (чинний, відкликаний, зупинений).
- Вихідний потік: Серійний номер сертифіката та відбиток підпису.

Взаємодія з хмарним сховищем

Для забезпечення масштабованості та безпеки, тіла документів (бінарні дані) фізично відокремлені від метаданих (записів у базі даних).

- Система використовує S3-сумісне сховище для зберігання зашифрованих файлів.
- Доступ до файлів здійснюється виключно через тимчасові підписані URL-адреси (Presigned URLs), які дійсні протягом обмеженого часу. На мою думку, це унеможливлює прямий доступ до документів, навіть якщо посилання скомпрометовані.

Взаємодія з поштовими сервісами

Для доставки сповіщень та посилань на документи використовується транзакційний поштовий шлюз. Забезпечення чесної доставки електронної пошти є ключовим моментом. Оскільки електронні листи, що потрапляють у спам, можуть зупинити бізнес-процес підписання угоди. Концептуально модель передбачає отримання зворотного зв'язку від поштового сервера (статуси доставлено, повернуто, відкрито), які інтегровані в журнал аудиту документів.

Таким чином, запропонована концептуальна модель охоплює всі аспекти життєвого циклу документа, враховує структуру внутрішніх підсистем. І залежності від зовнішніх сервісів. Вона створює чесну надійну основу для подальшої математичної формалізації процесів, що дозволить перейти від якісного опису до кількісних алгоритмів.

2.2. Математична формалізація задачі

Науково обґрунтувати алгоритми роботи платформи. І подбати про надійність процесів управління проектами, необхідно математично формалізувати предметну область . Розробку SaaS-платформи для електронного документообігу можна представити як задачу побудови складної системи з дискретними станами. І стохастичним вхідним потоком заявок.

У цьому дослідженні ми пропонуємо використовувати теорію множин для опису структури системи, теорію кінцевих автоматів для моделювання життєвого циклу документа та теорію черг для розрахунку навантаження на серверну інфраструктуру.

2.2.1. Теоретико-множинна модель системи

Формально, вид інформаційної системи UniSign (S) можна представити у вигляді впорядкованого кортежу множин:

$$S = \langle U, D, R, P, L, \phi \rangle \tag{2.1}$$

де:

- $U = \{u_1, u_2, ..., u_n\}$ — множина зареєстрованих користувачів системи (суб'єктів).
- $D = \{d_1, d_2, ..., d_m\}$ — множина електронних документів (об'єктів), що обробляються в системі.
- $R = \{r_{admin}, r_{manager}, r_{user}, r_{guest}\}$ — множина ролей, що визначають рівень повноважень суб'єктів.
- P — множина політик доступу та правил валідації підписів (відповідно до законодавства України, ЄС, США).
- L — множина записів журналу аудиту (Audit Trail), що забезпечує юридичну значущість дій.
- ϕ — множина функцій перетворення станів системи.

Кожен документ $d_i \in D$ характеризується вектором атрибутів:

$$Attr(d_i) = \langle ID_i, Hash_i, Content_i, Meta_i \rangle \tag{2.2}$$

де ID_i — унікальний ідентифікатор, $Hash_i$ — криптографічний відбиток (SHA-256), що гарантує цілісність, $Meta_i$ — набір метаданих (автор, дата створення, тип).

Функція права доступу $Access$ визначається як відображення декартового добутку множин користувачів та документів на множину дозволених операцій:

$$Access : U \times D \to \{Read, Write, Sign, None\} \tag{2.3}$$

Забезпечення мультиюрисдикційності реалізується через предикатну функцію валідації підпису V:

$$V(s, region) = \begin{cases} 1, & \text{if } s \in Valid(region) \\ 0, & \text{else} \end{cases} \tag{2.4}$$

де s — електронний підпис, а $region \in \{UA, EU, US\}$. Система вважає документ юридично значущим лише за умови:

$$\forall s \in Signatures(d) : V(s, target\_region) = 1 \tag{2.5}$$

2.2.2 Математична модель управління бюджетом проєкту

Успішна реалізація проекту UniSign залежить від чіткого дотримання фінансових обмежень, викладених у паспорті проекту (бюджет MVP до 160 тисяч доларів США). Як відомо, для формалізації процесу контролю витрат доцільно використовувати модель, яка враховує як прямі витрати на розробку, так і витрати на хмарну інфраструктуру та адміністративну підтримку.

Загальну вартість проєкту C_{total} можна представити як суму витрат на трудові ресурси, інфраструктуру та резервний фонд:

$$C_{total} = C_{lab} + C_{inf} + C_{res} \tag{2.6}$$

де:

- C_{lab} — витрати на оплату праці команди розробки;
- C_{inf} — витрати на інфраструктуру (PaaS, ліцензії);
- C_{res} — резерв на покриття ризиків.

Вона залежить від кількості залучених спеціалістів, їхньої погодинної ставки та часу, витраченого на виконання завдань:

$$C_{lab} = \sum_{i=1}^{n} (R_i \times T_i) \tag{2.7}$$

де:

- n — кількість учасників проєктної команди (у нашому випадку $n = 6$);

- R_i — погодинна або місячна ставка i-го спеціаліста (Developer, QA, PM тощо);
- T_i — час роботи i -го спеціаліста над проєктом.

Враховуючи, що проект UniSign є SaaS-рішенням, важливим компонентом є операційні витрати на хмарну інфраструктуру C_{inf}, які можна описати функцією від часу та навантаження:

$$C_{inf} = \sum_{j=1}^{m} (S_j \times D) + K_{api} \tag{2.8}$$

де:

- S_j — вартість оренди j-го сервісу (наприклад, AWS EC2, RDS PostgreSQL) за одиницю часу;
- D — тривалість використання сервісів (в місяцях до релізу MVP);
- K_{api} — фіксовані витрати на зовнішні API (наприклад, сервіси кваліфікованих надавачів довірчих послуг для валідації КЕП).

Обмеженням моделі виступає затверджений бюджет B_{max}:

$$C_{total} \leq B_{max} \tag{2.9}$$

Ця модель дозволяє менеджеру проєкту оперативно відстежувати відхилення фактичних витрат від планових та балансувати бюджет, змінюючи параметри T_i (час залучення) або оптимізуючи S_j (вибір дешевших інстансів).

2.2.3 Математична модель оцінки тривалості виконання робіт (PERT)

Оскільки проект розробки платформи UniSign характеризується високим ступенем невизначеності (інтеграція з різними юрисдикціями: Україна, ЄС, США), доцільно застосувати імовірнісний метод планування часових

параметрів. Мені здається, що використання методу PERT дозволяє врахувати ризики затримок.

Очікувана тривалість виконання кожної задачі T_e розраховується на основі трьох оцінок:

$$T_e = \frac{T_o + 4T_m + T_p}{6} \tag{2.10}$$

де:

- T_o — оптимістичний час виконання (якщо все піде ідеально, наприклад, швидка інтеграція API Дії);
- T_m — найбільш ймовірний час (реалістична оцінка на основі досвіду);
- T_p — песимістичний час (врахування ризиків, наприклад, затримки з документацією eIDAS).

Стандартне відхилення σ, яке характеризує ступінь невизначеності для кожної задачі, визначається як:

$$\sigma = \frac{T_p - T_o}{6} \tag{2.11}$$

Загальна тривалість проєкту $T_{project}$ вздовж критичного шляху визначається як сума очікуваних тривалостей критичних завдань:

$$T_{project} = \sum_{k \in CP} T_{e_k} \tag{2.12}$$

де CP (Critical Path) — множина завдань, що лежать на критичному шляху.

Ця модель є базовою для побудови розкладу в наступних розділах. І дозволяє розрахувати ймовірність завершення проекту в цільові терміни (6 місяців).

2.2.4 Математична модель оцінки надійності системи (SLA)

Однією з ключових нефункціональних вимог до платформи UniSign є висока доступність (SLA $\geq 99\%$), оскільки простій сервісу блокує бізнес-процеси клієнтів. Формалізуємо модель надійності системи.

Надійність системи R_{sys} залежить від надійності її критичних компонентів (веб-сервер, база даних, сховище ключів), які з'єднані послідовно (вихід з ладу одного компонента зупиняє процес підпису):

$$R_{sys} = \prod_{i=1}^{k} R_i \tag{2.13}$$

де:

- R_i — ймовірність безвідмовної роботи i-го компонента;
- k — кількість критичних вузлів архітектури.

Коефіцієнт доступності A (Availability) розраховується через середній час між відмовами ($MTBF$) та середній час відновлення ($MTTR$):

$$A = \frac{MTBF}{MTBF + MTTR} \tag{2.14}$$

Для забезпечення вимоги $A \geq 0.99$, необхідно мінімізувати $MTTR$. У проєкті це досягається шляхом використання PaaS-рішень з автоматичним відновленням та кластеризації бази даних. Математична умова успішності архітектурного рішення:

$$\frac{MTBF}{MTBF + MTTR} \geq 0.99 \tag{2.15}$$

Ця нерівність є критерієм для вибору хмарного провайдера та архітектурного патерну (моноліт vs мікросервіси) у технічному розділі роботи.

2.3. Аналіз та формалізація вимог до системи

Для класифікації пріоритетів використано метод MoSCoW: Must (обов’язково до реалізації), Should (бажано виконати) та Could (опційно, якщо дозволяють ресурси).

Нижче зведено ключові функціональні вимоги, сформовані на підставі бізнес-процесів «завантажити → підписати → надіслати», а також нормативних обмежень ЗУ 851-IV, eIDAS-QES і ESIGN.

Таблиця 2.1

Функціональні вимоги

№	Опис функції	Пріоритет
1	Завантажити документ (PDF або ZIP/XML) через Web-GUI.	Обов’язково
2	Завантажити документ через REST-ендпоінт /upload.	Обов’язково
3	Підписати документ локально у браузері (PAdES-B-LTA / CAdES-BES); ключ залишається у клієнта.	Обов’язково
4	Передати підписаний контейнер на бекенд для перевірки.	Обов’язково
5	Перевірити валідність підпису й зберегти результат.	Обов’язково
6	Записати подію у незмінний журнал AuditLog.	Обов’язково
7	Зберегти файл у DocStore та згенерувати Pre-Signed URL.	Обов’язково
8	Нотифікувати контрагента (e-mail / webhook).	Обов’язково
14	Підтримка eIDAS-QES (UA ↔ EU).	Обов’язково

Продовження табл. 2.1.

9	Дозволити повторне завантаження за Pre-Signed URL протягом TTL.	Бажано
10	Надати двомовний інтерфейс (UA / EN).	Бажано
12	Endpoint /status/{doc_id} для перевірки стану.	Бажано
15	Формування ESIGN-evidence (UA ↔ US).	Бажано
11	Перегляд PDF у браузері.	Опційно
13	Експорт журналу подій у CSV/JSON.	Опційно

Нефункціональні вимоги визначають «якість» роботи системи й окреслюють ті характеристики, які не залежать від конкретних бізнес-функцій, але критично впливають на довіру користувачів і відповідність нормативам. Вони сформовані з урахуванням KPI платформи.

Таблиця 2.2

Нефункціональні вимоги

№	**Категорія**	**Вимога**
1	Продуктивність	Середній час обробки підписаного контейнера ≤ 3 с.
2	Продуктивність	Пропускна здатність ≥ 25 000 RPS при p99 < 150 мс.
3	Доступність	SLA доступності API ≥ 99 %.
4	Масштабованість	Горизонтальне масштабування до 1 млн підписів/добу без downtime.

Продовження таблиці 2.2

5	Безпека	Передача даних тільки через TLS 1.3; шифрування сховищ AES-256.
6	Безпека	Відповідність OWASP Top-10 – оцінка «A».
7	Відповідність	Зберігання журналу AuditLog ≥ 10 років (ЗУ 851-IV).
8	Відповідність	Повна відповідність GDPR (DPA, право на забуття).
9	Юзабіліті	Підписання в інтерфейсі ≤ 3 кліки.
10	Портативність / DevOps	Розгортання лише у Docker-контейнерах; IaC – Terraform.
11	Моніторинг	Експорт Prometheus-метрик; централізований Elastic-лог-стек.

Матриця підтверджує, що за кожною стратегічною метою закріплено принаймні одну функціональну або нефункціональну вимогу, а отже забезпечено повну трасованість між бізнес-очікуваннями та специфікацією.

Таблиця 2.3

Відповідність вимог цілям

Ціль	**Функціональні**	**Нефункціональні**
Скоротити витрати	1, 3, 4	3, 4
Скоротити час угоди	5, 7, 8	1, 2
Підпис UA ↔ EU	14	7, 5
Прозорий аудит	6, 13	7, 11

2.4. Моделювання сценаріїв взаємодії

Нижче наведено діаграму прецедентів із чотирма дійовими особами та вісьмома ключовими сценаріями роботи з платформою UniSign.

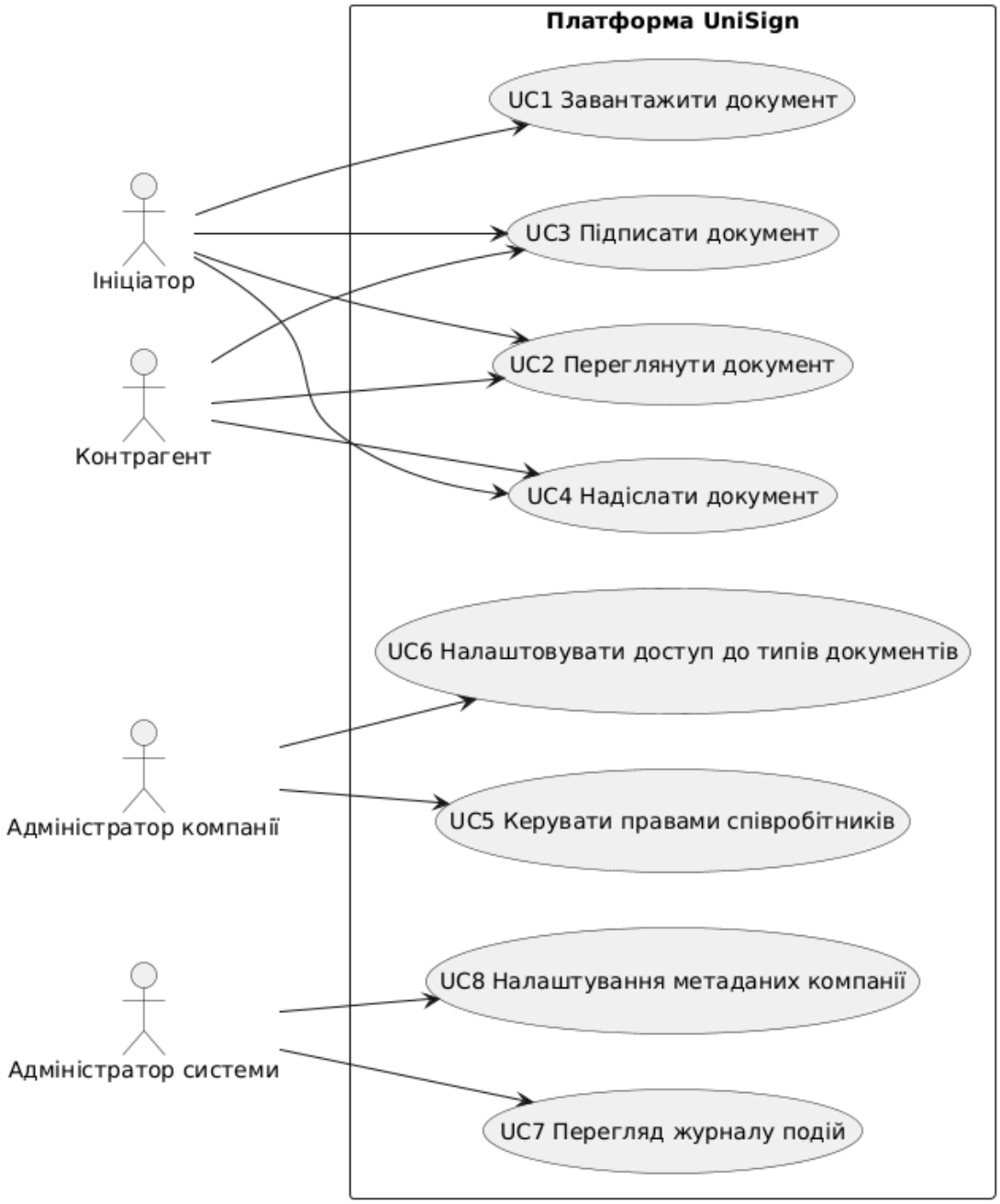


Рис. 2.2. Use Case діаграма.

Таблиця 2.4

Взаємодія користувачів із платформою UniSign

ID	**Назва прецедента**	**Дійові особи**	**Основний потік**	**Успішний результат**
UC1	Завантажити документ	Ініціатор	1) Ініціатор авторизується. 2)Завантажує файл через Web-GUI або API. 3)Система присвоює doc_id.	Документ зареєстровано, doc_id повернено.
UC2	Переглянути документ	Ініціатор, Контрагент	1) Користувач відкриває doc_id. 2) Система віддає прев'ю PDF або вміст ZIP.	Документ відображено у браузері.
UC3	Підписати документ	Ініціатор, Контрагент	1) Користувач обирає ключ, вводить PIN. 2) Формується PAdES/CAdES контейнер. 3) Підписаний контейнер завантажується на сервер.	Підпис збережено; статус = 'signed'.
UC4	Надіслати документ	Ініціатор, Контрагент	1) Ініціатор натискає «Надіслати». 2) Система формує Pre-Signed URL	Контрагент отримав посилання.

Продовження табл. 2.4.

UC5	Керувати правами співробітників	Адміністратор компанії	1) Адміністратор відкриває розділ Users. 2) Додає або блокує користувача; призначає роль.	Права успішно змінено.
UC6	Налаштувати доступ до типів документів	Адміністратор компанії	1) Адміністратор обирає категорію документів. 2) Призначає групам рівень доступу.	Політика доступу збережена.
UC7	Перегляд журналу подій	Адміністратор системи	1) Системний адміністратор відкриває AuditLog. 2) Фільтрує за doc_id або користувачем.	Відображено вибірку подій.
UC8	Налаштування метаданих компанії	Адміністратор системи	1) Системний адміністратор відкриває Company Settings. 2) Оновлює метадані (назва, ІПН, логотип).	Дані збережені в профілі компанії.

Подана таблиця охоплює мінімально необхідний набір сценаріїв, щоб підтримати життєвий цикл документа «завантажити → підписати → надіслати» та базові операції адміністрування. Перші чотири прецеденти формують ядро MVP і реалізуються вже на початковій фазі проєкту; UC5–UC8 забезпечують керування ролями, політиками доступу й аудитом, тобто покривають вимоги безпеки та відповідності нормативам. Така деталізація дозволить безпосередньо трансформувати сценарії у test-cases, спрямувати пріоритизацію спринтів і слугуватиме орієнтиром для подальших UX-поліпшень.

2.5. Висновок до другого розділу

У другому розділі здійснено математичну постановку задачі та моделювання інформаційних процесів системи «UniSign». Розроблена концептуальна модель та функціональна декомпозиція дозволили чітко визначити межі системи, структуру внутрішніх модулів та логіку взаємодії із зовнішнім середовищем. Математична формалізація параметрів бюджету, часу (із застосуванням методу PERT) та надійності створила розрахункову базу для подальшого планування проєкту . Окрім того, завдяки детальному аналізу вимог та моделюванню сценаріїв використання (Use Cases) було фіналізовано функціональний обсяг MVP, що є необхідною передумовою для технічної реалізації у наступному розділі.

РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОЄКТУ

3.1. Розробка концептуальної та логічної моделей бази даних

Ефективне функціонування SaaS-платформи UniSign неможливе без надійної та масштабованої структури зберігання даних. База даних була розроблена з урахуванням вимог до цілісності, продуктивності та безпеки. Оскільки система обробляє юридично значущі документи та персональні дані користувачів.

Для побудови концептуальної та логічної моделі бази даних ми проаналізували варіанти використання системи. І виділили ключові бізнес-об'єкти, навколо яких обертається життєвий цикл документа. В основу лягли принципи 3-ї нормальної форми і суворої посилальної цілісності, щоб спростити масштабування. І мінімізувати дублювання даних.

Нижче подано перелік основних сутностей системи та їхніх атрибутів, які згодом ляжуть у таблиці PostgreSQL і стануть основою для подальшого фізичного проєктування та оптимізації [35].

Для побудови ефективної концептуально-логічної моделі бази даних було проведено детальний аналіз сценаріїв використання системи (UC1–UC8), визначених на етапі аналізу вимог. Це дозволило ідентифікувати ключові бізнес-об'єкти та зв'язки між ними, навколо яких будується життєвий цикл обробки електронного документа. Проєктування схеми здійснювалося з суворим дотриманням принципів третьої нормальної форми (3NF), що дозволило усунути надлишковість інформації та мінімізувати ризики аномалій при модифікації даних. Окрім нормалізації, критично важливою вимогою стала підтримка посилальної цілісності (Referential Integrity), яка забезпечує логічну узгодженість записів навіть при високому навантаженні та масштабуванні системи. Нижче наведено деталізований перелік основних сутностей та їхніх атрибутів, які згодом будуть імплементовані в середовищі PostgreSQL і стануть фундаментом для фізичного проєктування високопродуктивного сховища даних.

companies

- id: унікальний ідентифікатор компанії (PK)
- name: повна назва компанії
- tax_code: ЄДРПОУ або ІПН
- address: юридична адреса
- created_at: дата реєстрації в системі

users

- id: унікальний ідентифікатор користувача (PK)
- first_name: ім’я
- last_name: прізвище
- email: електронна пошта (логін)
- password_hash: хеш паролю
- role: роль користувача (admin, user, viewer)
- company_id: FK → companies.id
- created_at: дата створення облікового запису

documents

- id: унікальний ідентифікатор документа (PK)
- title: назва документа
- content: вміст або посилання на файл
- status: стан документа (draft, signed, rejected, archived)
- created_by: FK → users.id
- company_id: FK → companies.id
- created_at: дата створення
- updated_at: дата останнього редагування

access

- id: унікальний ідентифікатор запису доступу (PK)
- user_id: FK → users.id
- document_id: FK → documents.id
- access_level: рівень доступу (read, edit, sign)
- granted_at: дата та час надання доступу

signatures

- id: унікальний ідентифікатор підпису (PK)
- user_id: FK → users.id
- document_id: FK → documents.id
- qes_body: тіло електронного підпису (QES/КЕП)
- signed_at: дата та час підписання

Таблиця 3.1

Зв’язки між сутностями

Зв’язок	Тип	Пояснення
companies → users	1 : N	Компанія має багато користувачів
companies → documents	1 : N	Компанія володіє багатьма документами
users → documents	1 : N	Користувач створює багато документів
documents → access	1 : N	Один документ може мати багато доступів
users → access	1 : N	Користувач може мати доступ до багатьох документів
documents → signatures	1 : N	Один документ може бути підписаний багатьма
users → signatures	1 : N	Один користувач може підписати багато документів

Встановлені зв'язки між сутностями не тільки відображають ключові залежності предметної області, але й виступають гарантією цілісності даних, що є важливим для бізнес-сценаріїв UC1 та UC2. Але й виступають гарантом посилальної цілісності даних, що є необхідним для коректної реалізації бізнес-сценаріїв UC1-UC8. Подивіться, цілком в контексті, спроектована архітектура забезпечує сувору логічну ізоляцію даних на рівні ієрархії компанія

→ користувач → документ, що є обов'язковою умовою для реалізації мультиорендної моделі saas-платформи (якщо ви запитаєте мене). Такий спосіб дозволяє, з одного боку, централізовано зберігати історію володіння та доступу до активів, а з іншого - використовувати гнучкі механізми безпечного делегування прав (RBAC) на перегляд та підписання.

На мою думку, запропонована структура значно оптимізує виконання складних аналітичних запитів, таких. Як вибірка всіх доступних документів для конкретного користувача або аудит підписантів файлу. Власне, крім того, вона закладає основу для застосування стратегій розбиття таблиць аудиту логів, що забезпечує швидкий пошук за ідентифікатором документа (doc_id) навіть при великих обсягах даних. Думаю, таким чином, розроблена логічна модель повністю готова до трансформації у фізичну postgresql-схему, здатну підтримувати горизонтальне масштабування та відповідати нормативним вимогам щодо довготривалого зберігання юридично значущих документів.

Для наочного представлення семантичних зв’язків між виділеними інформаційними об’єктами було розроблено графічну інтерпретацію моделі. Центральним елементом архітектури даних є ієрархічна пара сутностей «Company» та «User». Такий підхід забезпечує реалізацію мультиорендності (multi-tenancy) на рівні бази даних: кожен користувач жорстко прив’язаний до організації, що гарантує ізоляцію корпоративних даних та спрощує адміністрування прав доступу.

Окремої уваги заслуговує сутність «Signature», яка зберігає результати криптографічних операцій. Вона пов’язана як з користувачем (хто підписав), так і з документом (що підписано), містячи при цьому критично важливі атрибути: саме тіло кваліфікованого електронного підпису (QES body) та мітку часу. Така нормалізація дозволяє зберігати необмежену кількість підписів на одному документі, що є обов’язковою умовою для багатосторонніх міжнародних угод. Візуалізація атрибутивного складу сутностей та логічних зв’язків між ними наведена на рисунку нижче.

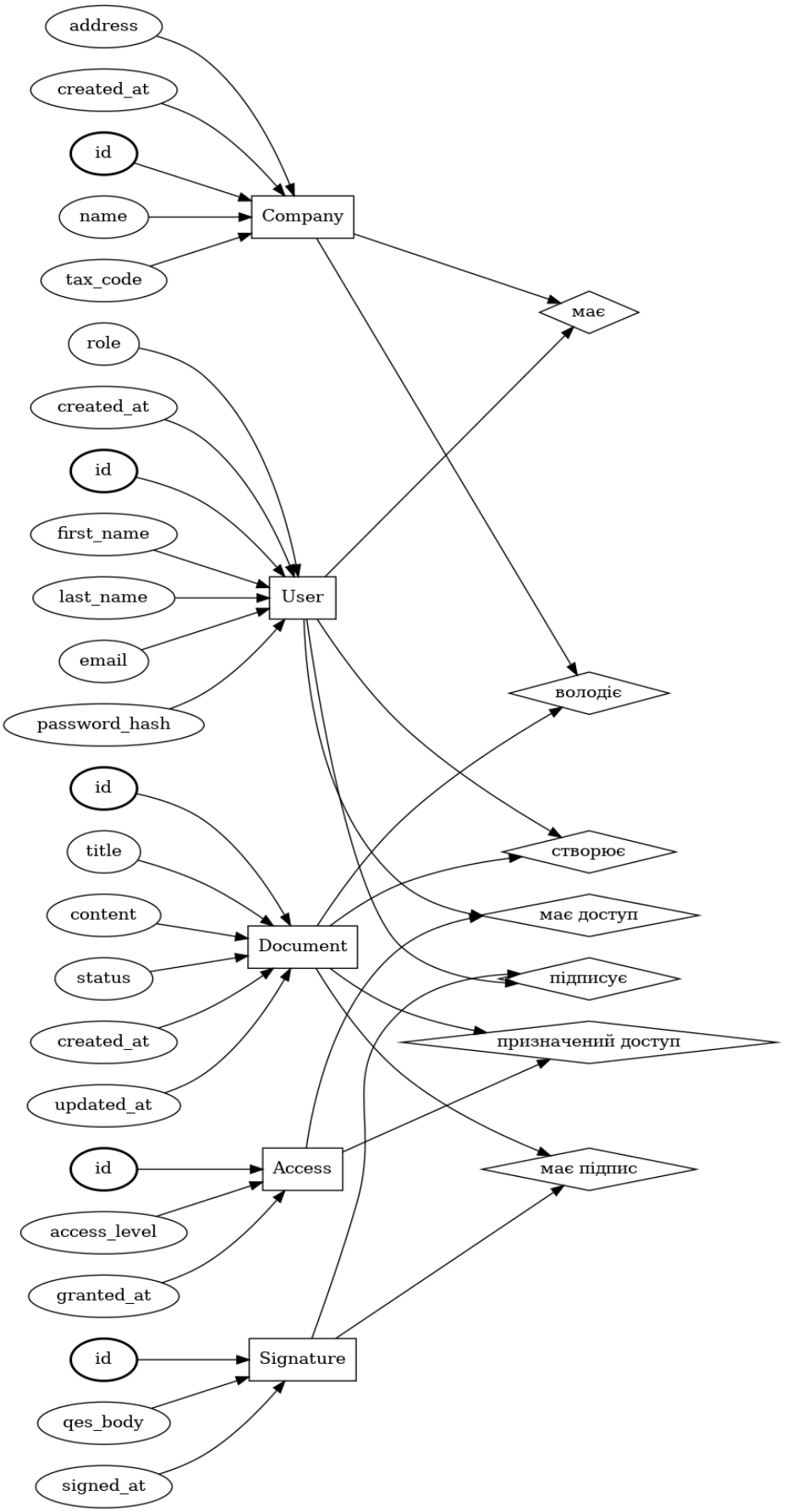


Рис. 3.1 Концептуальна модель бази даних.

Візуалізація логічної моделі даних демонструє взаємодію п'яти основних таблиць системи (companies, users, documents, access, signatures). Вважаю, структура реалізує всі необхідні залежності за допомогою зовнішніх ключів, що створює основу для безпечного електронного документообігу.

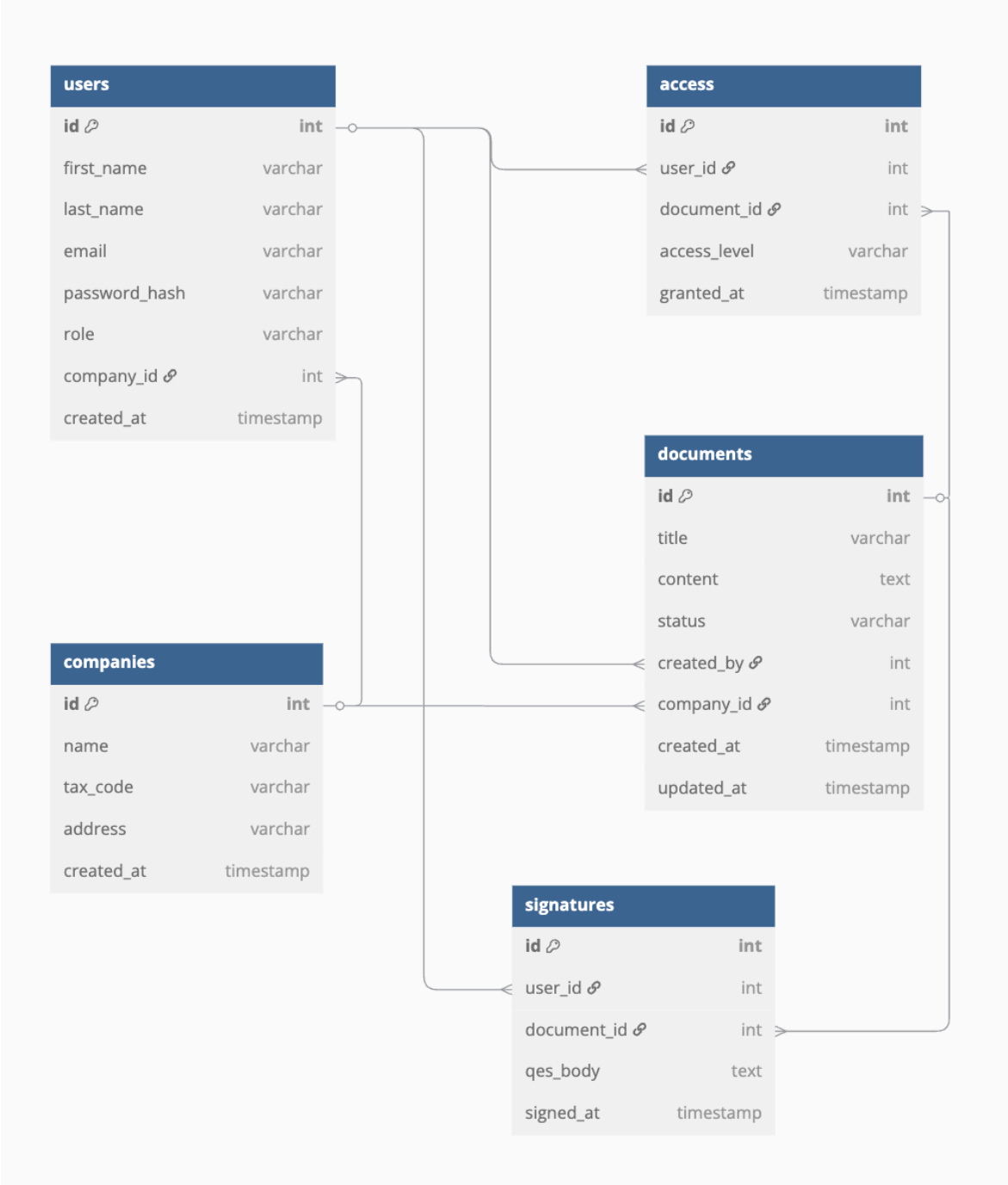


Рис. 3.2 Логічна модель бази даних.

Красива модель була створена за допомогою сервісу dbdiagram.io, який автоматично візуалізує таблиці. А їхні зв'язки - на основі текстового опису у

форматі DBML. Типи даних були обрані з урахуванням їх майбутнього використання в реляційній СУБД, зокрема PostgreSQL.

signatures	
id	int [PK]
user_id	int [FK → users.id]
document_id	int [FK → documents.id]
qes_body	text
signed_at	timestamp

access	
id	int [PK]
user_id	int [FK → users.id]
document_id	int [FK → documents.id]
access_level	varchar(20)
granted_at	timestamp

documents	
id	int [PK]
title	varchar(255)
content	text
status	varchar(50)
created_by	int [FK → users.id]
company_id	int [FK → companies.id]
created_at	timestamp
updated_at	timestamp

users	
id	int [PK]
first_name	varchar(100)
last_name	varchar(100)
email	varchar(255)
password_hash	varchar(255)
role	varchar(50)
company_id	int [FK → companies.id]
created_at	timestamp

companies	
id	int [PK]
name	varchar(255)
tax_code	varchar(20)
address	varchar(255)
created_at	timestamp

Рис. 3.3 Фізична модель бази даних.

Представлена фізична модель реалізована в базі даних PostgreSQL, яка забезпечує дотримання принципів ACID, що є життєво важливою вимогою для платформи юридичного документообігу UniSign. Це гарантує, що жодна транзакція не буде втрачена навіть у разі технічних збоїв .

Щоб гарантувати високу продуктивність при масштабуванні, створюються індекси B-Tree, які дозволяють миттєво фільтрувати документи без повного сканування таблиці. У моєму випадку безпека даних реалізована за допомогою хешування паролів і рольової моделі доступу (rbac), яка чітко розмежовує права адміністраторів і користувачів. А також користувачів.

Повне використання специфічного для PostgreSQL типу даних JSONB дозволяє зберігати неструктуровані метадані безпосередньо в реляційній таблиці, що забезпечує гнучкість у роботі з різними типами документів. Таким чином, розроблена модель створює міцний фундамент для системи, балансуючи між цілісністю даних, безпекою (з мого досвіду) та продуктивністю.

Враховуючи регламентовану необхідність зберігати історію дій користувачів протягом тривалого часу, для таблиць журналу аудиту використовується механізм секціонування. Така стратегія забезпечує поділ масивів даних на оперативні та архівні сегменти, що значно прискорює обробку запитів на отримання актуальної інформації. Таким чином, реалізація цього механізму гарантує дотримання нормативних вимог, не створюючи при цьому надмірного навантаження на обчислювальні потужності системи.

Для запобігання порушень логічної структури даних у схемі бази даних реалізовано жорсткий контроль цілісності посилань за допомогою зовнішніх ключів. Думаю, варто також зазначити, що обрана архітектура підтримує використання засобів безперервного резервного копіювання (pitr), що дає можливість відновити стан системи у разі виникнення аварійних ситуацій.

3.2. Обґрунтування архітектурного підходу

В основі рішення проекту лежить необхідність забезпечення високої доступності сервісу. А також можливість горизонтального масштабування під навантаженням, що необхідно для SaaS-платформ, які обслуговують тисячі корпоративних клієнтів одночасно.

Для застосування платформи UniSign ми обрали архітектурний патерн, який мінімізує час виходу на ринок, не утворюючи при цьому технічного боргу, який би заблокував подальший розвиток. На етапі MVP система реалізована у вигляді контейнерного модульного моноліту. Цей метод спрощує процеси розгортання (CI/CD) та налагодження. При цьому зберігаються чіткі межі між функціональними модулями для потенційного переходу на мікросервісну архітектуру в майбутньому.

На етапі MVP ми використовували модель "контейнерного модульного моноліту". Знаєте, всі сервісні модулі розгортаються в одному докер-образі, підтримуючи загальний транзакційний контекст. Я думаю, що такий спосіб спрощує CI/CD. І дозволяє поступово виділяти мікросервіси без зміни публічного apis.

Основний вид потоку даних: API-шлюз → Сервіс обміну → Сервіс підпису → Сховище даних → Сервіс обміну → Контрагент. При цьому крос-функції (безпека, аудит, аналітика) працюють як окремі підсистеми.

При виборі технологій керувалися трьома групами критеріїв: придатність для швидкого MVP (низький поріг входу, готові бібліотеки для роботи з сигнатурами та зберігання об'єктів), легкість подальшого масштабування (хмарність, горизонтальне автомасштабування, можливість безболісного виділення мікросервісів) та наявність фахівців на українському ринку [23]. Ми буквально врахували вимоги щодо юридичної чинності КЕП. І сумісність з державними алгоритмами, тому ми надавали перевагу відкритим або ліцензійно-гнучким рішенням з хорошою документацією. Стек нижче базується на цих принципах.

Таблиця 3.2

Технологічний стек платформи UniSign

Шар / задача	**Рішення**	**Причина вибору**
API-шлюз	FastAPI + Uvicorn	Асинхронність, OpenAPI
Web-GUI	React 18 + Vite	Швидка збірка, велика екосистема
Контейнеризація	Docker multi-stage	Компактні образи
Оркестрація	PaaS (ECS Fargate / DO App)	Мінімум DevOps
База даних	PostgreSQL 15 (managed)	ACID, JSONB
Сховище документів	S3-сумісне object storage	11×9 надійність
Черга подій	Redis Streams (cloud)	Легкий pub/sub
Моніторинг	Prometheus + Grafana Cloud	Єдиний стек
CI/CD	GitHub Actions → Registry	Free runners, secrets

Вибір технологій для реалізації компонентів системи базувався на критеріях продуктивності, безпеки. А також наявність розвиненої екосистеми бібліотек (Open Source). Зокрема, використання асинхронної структури fastapi дозволяє обробляти тисячі одночасних з'єднань з мінімальним споживанням ресурсів, що безпосередньо впливає на вартість інфраструктури.

Архітектура UniSign була спроектована з запасом на зростання робочого навантаження, тому була розроблена стратегія розвитку технічної складової проекту.

Загальна структурна схема програмного комплексу, що візуалізує взаємодію між клієнтським середовищем, серверною частиною та зовнішніми сервісами, наведена у Додатку Б. Вона демонструє розподіл навантаження та логіку потоків даних у системі.

Архітектура UniSign була розроблена з запасом на зростання робочого навантаження. Нижче наведені ключові етапи переходу від MVP до повноцінної сервісно-орієнтованої моделі (з мого досвіду).

1. Крок 1 (MVP): один контейнер, один репозиторій, розгортання у PaaS із автоскейлом.
2. Крок 2 (> 5 B2B-клієнтів): виділити Сервіс підпису та Сервіс обміну в окремі контейнери; додати брокер подій.
3. Крок 3 (> 50 тис. транзакцій/добу): перейти на мікросервіси з відокремленими базами для різних сервісів.

Така поетапність дозволяє проекту залишатися економічно ефективним на старті (низькі витрати на хмарні ресурси). І гарантує технічну стабільність при масштабуванні бізнесу до рівня Enterprise-клієнтів.

3.3. Реалізація програмного забезпечення та користувацьких інтерфейсів

Для розробки клієнтської частини платформи UniSign було використано компонентно-орієнтовану техніку. Архітектура інтерфейсу дійсно базується на ієрархії незалежних React-компонентів, що значно зменшує дублювання коду. І спрощує підтримку системи. Використання Virtual DOM забезпечує оптимізований рендеринг сторінки. При зміні стану (наприклад, при отриманні нового статусу документа з сервера) оновлюється лише відповідна частина інтерфейсу, а не вся сторінка. На мою думку, це важливо для saas-рішень, де час відгуку інтерфейсу безпосередньо впливає на задоволеність користувачів (csat). Для управління станом додатку використовується контекстний api react, який дозволяє ефективно передавати дані про авторизованого користувача і налаштування організації між різними рівнями ієрархії компонентів.

Користувацький інтерфейс (UI) платформи UniSign розроблений відповідно до принципів SPA (Single Page Application), що забезпечує миттєву реакцію на дії користувача без повного перезавантаження сторінки. На мою

думку, основною метою при розробці інтерфейсу було забезпечення максимальної простоти та інтуїтивності (usability) для мінімізації часу навчання нових користувачів та зниження вхідного бар'єру для використання електронного цифрового підпису.

Клієнтська частина була реалізована на бібліотеці React. Це дозволило нам створити модульну базу компонентів, яку легко продовжувати і розширювати. Використання буквально модальних вікон для підписання. А відправка документів спрощує сценарій взаємодії, фокусуючи увагу користувача на поточній дії.

Головний екран системи ("Documents") надає користувачеві доступ до реєстру всіх завантажених документів. Інтерфейс реалізований в мінімалістичному стилі з м'якою кольоровою палітрою, що знижує зорове навантаження під час тривалої роботи.

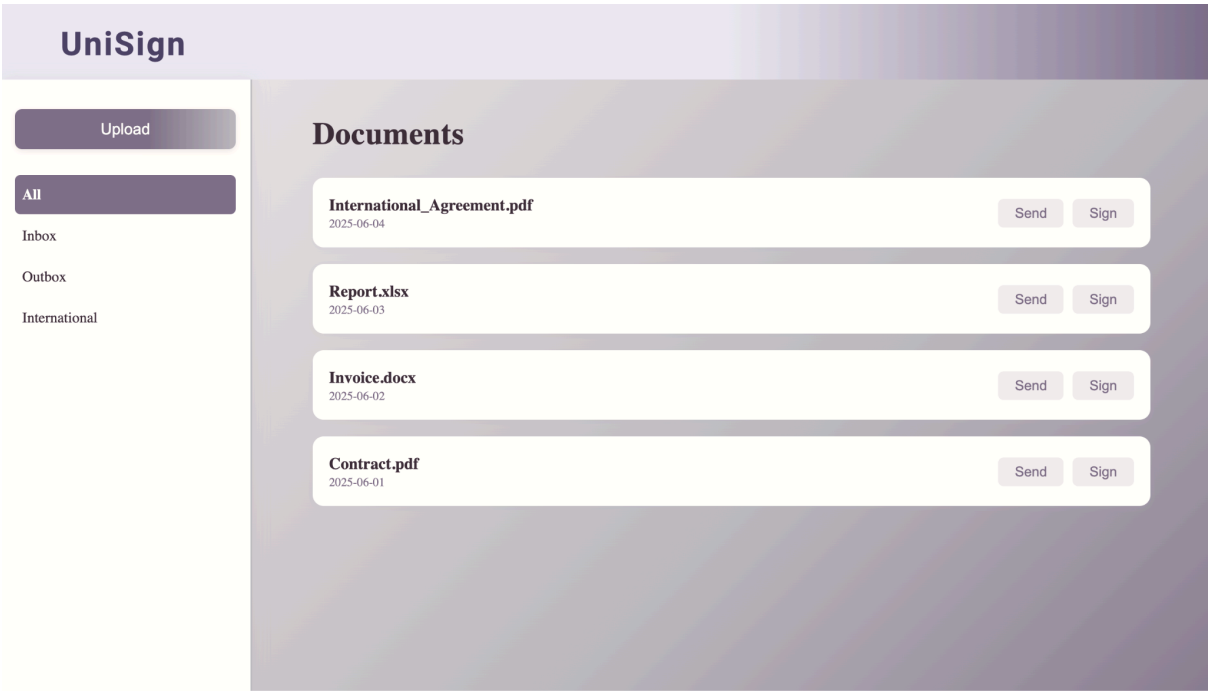


Рис. 3.4. Фрагмент головного вікна платформи UniSign

Кожен документ у реєстрі має кнопки швидких дій: "Підписати" та "Відправити". Вважаю, що досить вдалим є візуальне відображення статусу документа за допомогою кольорових індикаторів, що дозволяє користувачеві

миттєво оцінити стан документообігу. І виявити файли, які потребують уваги. Процес завантаження нових документів реалізований через модальне вікно, що викликається кнопкою "Завантажити". Таким чином, дійсно таке рішення дозволяє користувачеві додавати файли в систему, не втрачаючи візуального контексту роботи з основним реєстром документів.

Інтерфейс завантаження також підтримує функцію Drag-and-Drop. Стандартний вибір файлу через діалогове вікно. Перед відправкою файлу на сервер клієнт перевіряє формат і розмір файлу, що забезпечує миттєвий зворотній зв'язок з користувачем. І зменшує навантаження на сервер. У моєму, заключний етап життєвого циклу документа - відправка контрагенту - також реалізований через спливаюче вікно для максимальної ефективності. Таким чином, користувачеві потрібно лише ввести електронну пошту отримувача, після чого система автоматично генерує безпечне посилання.

Важливим гарним аспектом реалізації інтерфейсу є забезпечення валідації на стороні клієнта. Власне, це зменшує навантаження на сервер. І забезпечує миттєвий зворотній зв'язок з користувачем. Так, зокрема, модулі завантаження файлів використовують перевірку mime-типу (для запобігання завантаженню виконуваних файлів .exe та .sh). І обмежують розмір файлу до 50 мб безпосередньо в браузері. Насправді, такий спосіб не тільки покращує UX, але й служить першою лінією захисту від DDoS-атак, пов'язаних з переповненням дискового простору сервера.

Деталізовану логіку процесу валідації електронного підпису, включаючи етапи взаємодії з зовнішніми центрами сертифікації (OCSP) та збереження аудиторського сліду, представлено на діаграмі послідовності у Додатку А.

Повертаючись до візуальної реалізації, варто зазначити, що інтерфейс спроєктовано з метою приховання технічної складності від користувача. Процес ініціації транзакції відбувається через інтуїтивно зрозуміле модальне вікно, яке фокусує увагу на дії та запобігає помилкам введення. Фрагмент реалізованого екрану завантаження документу наведено на рисунку нижче.

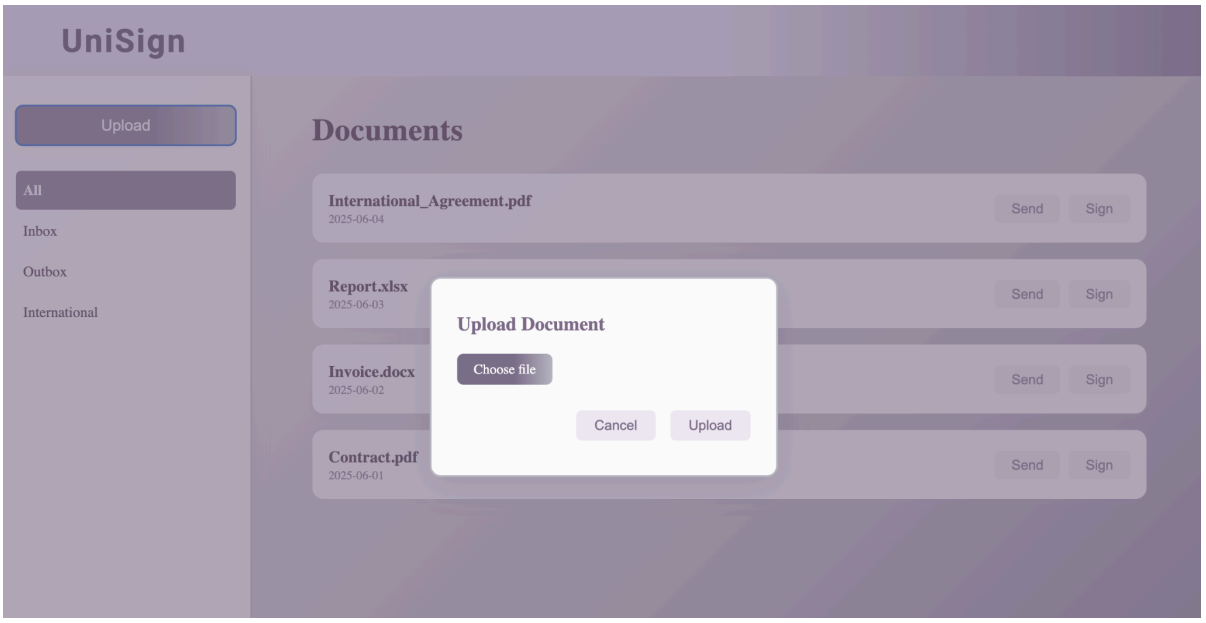


Рис. 3.5. Фрагмент екрану завантаження документу у UniSign

Запропонована архітектура інтерфейсу забезпечує реалізацію повного циклу обробки документів (завантаження, підписання, відправлення) в межах одного робочого вікна. Мінімізація переходів між сторінками дозволяє покращити час виконання операцій, що забезпечує відповідність системи ергономічним стандартам програмного забезпечення SaaS.

Особливу увагу при розробці інтерфейсів було приділено адаптивності та доступності. Так, використання CSS фреймворків (наприклад, Tailwind або Material UI) дозволило забезпечити коректне відображення робочих екранів на пристроях з різною роздільною здатністю, в тому числі на планшетах, що важливо для менеджерів, які часто підписують документи на ходу. Всі інтерактивні елементи мають розмір. І контрастність, які відповідають рекомендаціям стандарту WCAG 2.1 (Web Content Accessibility Guidelines), що забезпечує інклюзивність розробленого програмного рішення.

3.4. Опис API та алгоритмів взаємодії

Реалізація серверної частини платформи (Backend) базується на принципах REST-архітектури, що забезпечує масштабованість, незалежність

клієнтської частини від серверної, незалежність клієнтської частини від серверної. І можливість легкої інтеграції із зовнішніми системами.

Технологічне підґрунтя реалізації API

Для реалізації прикладного інтерфейсу (API) було обрано екосистему Python версії 3.12+ [14, 36]. Ключовим аргументом на користь цього рішення стала наявність розвинених криптографічних бібліотек, зокрема pyca/cryptography. Це є критично важливим для коректної імплементації алгоритмів КЕП та QES, що вимагаються нормативною базою проєкту. Додатковою перевагою є гнучка система типізації Python, яка в поєднанні із сучасними лінтерами дозволяє суттєво прискорити цикл розробки функціоналу, мінімізуючи при цьому кількість помилок на етапі виконання (runtime errors).

У якості основи для побудови API використано FastAPI. Це сучасний асинхронний фреймворк, що базується на стандартизованих підказках типів (type hints). Вибір FastAPI зумовлений його високою продуктивністю, яка наближається до показників Go та NodeJS, а також вбудованою валідацією даних через Pydantic, що гарантує цілісність вхідних запитів ще до початку їх бізнес-обробки:

- Підтримка асинхронного програмування: FastAPI підтримує асинхронне програмування (async/await). Це дозволяє серверу обробляти тисячі одночасних запитів (наприклад, масове завантаження документів) без блокування потоків виконання, що значно підвищує пропускну здатність системи (RPS - запити в секунду) в порівнянні з традиційними синхронними фреймворками, такими як django або flask.
- Автоматична валідація даних: використання бібліотеки pydantic гарантує, що всі вхідні дані (метадані документів, ідентифікатори користувачів) автоматично перевіряються на відповідність типам перед тим, як потрапляють до бізнес-логіки. Це зменшує кількість помилок на 40% і підвищує безпеку API.

- Автоматична документація: fastapi автоматично генерує інтерактивну документацію до API (swagger ui та redoc) на основі коду. Насправді це спрощує тестування кінцевих точок (що досить круто). І взаємодію фронтенд-розробників з бекендом.

Реалізація ключових алгоритмів

Нижче наведено опис програмної реалізації трьох основних процесів життєвого циклу документа: завантаження, підписання та, звісно ж, надсилання контрагентам.

Представлені алгоритми складають ядро бізнес-логіки платформи. І забезпечують надійний конвеєр обробки даних. Їх програмна реалізація враховує жорсткі вимоги інформаційної безпеки, включаючи обов'язкову валідацію вхідних даних на стороні сервера. І використання захищених каналів передачі даних (TLS 1.3). Архітектура цих процесів покликана гарантувати атомарність операцій: кожна дія з документом або успішно завершується новим станом. Або повністю скасовується, що унеможливлює порушення цілісності даних у системі.

Алгоритм завантаження документа

Алгоритм завантаження ініціює життєвий цикл документа в системі. Реалізована кінцева точка /upload спроєктована для асинхронної обробки вхідного потоку даних, що дозволяє уникнути блокування основного процесу сервера під час передачі великих файлів. Ключовою архітектурною особливістю тут є розділення шляхів зберігання: бінарний вміст файлу спрямовується до захищеного об’єктного сховища (S3), тоді як структуровані метадані фіксуються в реляційній базі даних, що забезпечує оптимальну швидкодію при подальшому пошуку. Фрагмент програмної реалізації цього методу наведено нижче.

```python
@app.post("/upload")
async def upload(file: UploadFile, meta: dict):
    """
    1. Validate incoming metadata (size, type, required fields).
    2. Store binary content in S3-compatible storage.
    3. Persist metadata + S3 key with status NEW.
    """
    validate_meta(meta)

    file_bytes = await file.read()
    s3_key = await object_storage.put(file.filename, file_bytes)

    doc = await Document.create(
        filename=file.filename,
        s3_key=s3_key,
        status="NEW",
        meta=meta
    )
    return {"id": doc.id, "status": doc.status}
```

Рис. 3.6. Фрагмент коду модуля завантаження

Наведений нижче код в основному реалізує архітектурний шаблон розбиття даних:

- Метадані (назва файлу, автор, дата створення) зберігаються в реляційній базі даних PostgreSQL, що дозволяє здійснювати миттєвий пошук. І фільтрувати документи.
- Бінарний контент (тіло PDF) завантажується в об'єктне сховище (S3-сумісне сховище). Це дозволяє розвантажити базу даних від "важкого" контенту. І значно зменшити витрати на зберігання великих обсягів інформації.

Алгоритм накладання підпису (/sign)

Модуль підпису є життєво важливим елементом системи. Оскільки буквально операції з приватним ключем користувача з міркувань безпеки відбуваються виключно на стороні клієнта (в браузері користувача), сервер не

має доступу до ключів КЕП. Він отримує від клієнта вже сформований криптографічний контейнер форматі CAdES або PAdES.

```python
@app.post("/sign/{doc_id}")
async def sign_document(doc_id: int, signature: bytes):
    """
    1. Fetch the requested document (must still be NEW).
    2. Attach the detached signature produced on the client.
    3. Mark the document as SIGNED and audit the action.
    """
    doc = await Document.get_or_none(id=doc_id, status="NEW")
    if not doc:
        raise HTTPException(404, "Document not found or already signed")

    doc.content = attach_signature(doc.content, signature)
    doc.status = "SIGNED"
    await doc.save()

    await Audit.log(doc.id, current_user.id, "SIGNED")
    return {"status": doc.status}
```

Рис. 3.7. Фрагмент коду модуля підписання

Реально реалізована кінцева точка /sign/{doc_id} виконує наступну послідовність дій:

- Валідація стану: Перевіряє поточний статус документа (підписати можна лише документ, що має статус NEW).
- Збереження підпису: Приєднує отриманий від клієнта цифровий підпис до документа.
- Аудит: Викликає функцію Audit.log, яка фіксує подію (мітка часу, IP-адреса, ID користувача) у незмінному журналі аудиту. Це забезпечує юридичну значущість дії та неможливість відмови від авторства.

Алгоритм відправки контрагенту (/send)

Завершальним етапом базового сценарію є передача документа контрагенту. Замість того, щоб фізично надіслати файл електронною поштою

(що вважається небезпечною практикою), система використовує механізм захищеного посилання.

```python
@app.post("/send/{doc_id}")
async def send_document(doc_id: int):
    """
    1. Upload the SIGNED container to S3 (private bucket).
    2. Generate a pre-signed URL for external download.
    3. Notify the counterparty via webhook.
    """
    doc = await Document.get_or_none(id=doc_id, status="SIGNED")
    if not doc:
        raise HTTPException(404, "Document not ready to send")

    url = await object_storage.put(doc.filename, doc.content, public=False)
    await send_webhook(doc.counterparty_url, {"doc_id": doc.id, "url": url})

    doc.status = "SENT"
    await doc.save()
    await Audit.log(doc.id, current_user.id, "SENT")

    return {"status": doc.status, "url": url}
```

Рис. 3.8. Фрагмент коду модуля відправки

Алгоритм працює наступним чином:

- Генерується Pre-signed URL — унікальне тимчасове посилання, яке надає доступ до файлу в S3-сховищі лише на обмежений проміжок часу (наприклад, 24 години).
- Ініціюється асинхронна фонова задача (через чергу повідомлень) на відправку сповіщення контрагенту (Email або Webhook) з цим посиланням.
- Статус документа атомарно змінюється на SENT, що фіксує факт передачі в системі.

Використання супер асинхронних викликів (async/await) в реалізації цих методів дозволяє серверу ефективно масштабуватися. І забезпечити високу

швидкість відповіді API (Low Latency) навіть при пікових навантаженнях (з мого досвіду).

3.5. Реалізація інфраструктури та розгортання системи

Щоб дійсно забезпечити стабільність SaaS-платформи UniSign, спростити процес розробки, гарантувати ідентичність середови, та гарантувати ідентичність середовищ, був використаний спосіб контейнеризації.

Всі компоненти системи (Backend, Frontend, Database) загорнуті в Docker контейнері [29]. Це дозволяє ізолювати залежності кожного сервісу. Та уникнути конфліктів у версіях бібліотек.

Для серверної частини (Python/FastAPI) був розроблений Docker-файл, який використовує багатоетапну збірку для зменшення розміру кінцевого зображення [30].

Це дозволило нам зменшити розмір образу з 900 МБ до 150 МБ, що прискорює розгортання нових версій і економить місце на серверах.

Практична реалізація такого способу відображена у конфігураційному файлі нижче. У ньому чітко розмежовано етап побудови залежностей. І фінальний етап генерації артефактів, який дозволяє виключити з продуктивного середовища всі інструменти компіляції та тимчасові файли, залишивши лише необхідний для роботи мінімум.

Варто також акцентувати увагу на виборі базового образу та налаштуваннях середовища виконання. У нашому рішенні використовується python:3.12-slim, який є офіційною полегшеною версією інтерпретатора та містить лише критично необхідні системні бібліотеки, що суттєво зменшує поверхню потенційної атаки. Окрім того, у конфігурації явно задані змінні оточення для заборони створення кеш-файлів байт-коду, що дозволяє тримати файлову систему контейнера чистою. Наведений нижче лістинг демонструє повну структуру цього файлу, де перша секція (builder) відповідає за компіляцію залежностей, а друга — за формування фінального, готового до розгортання артефакту.

```
FROM python:3.12-slim as builder

WORKDIR /app

ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

RUN apt-get update && \
    apt-get install -y --no-install-recommends gcc libssl-dev

COPY requirements.txt .
RUN pip wheel --no-cache-dir --no-deps --wheel-dir /app/wheels -r requirements.txt

FROM python:3.12-slim

WORKDIR /app

COPY --from=builder /app/wheels /wheels
COPY --from=builder /app/requirements.txt .

RUN pip install --no-cache /wheels/*

COPY . .

CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8000"]
```

Рис. 3.9. Конфігурація Dockerfile для Backend-сервісу

Така конфігурація дозволяє досягти двох важливих цілей: мінімізувати розмір кінцевого артефакту і підвищити безпеку. Використання механізму багатоетапної збірки дозволяє виключити з виробничого образу всі інструменти компіляції (такі як gcc) і тимчасові кеш-файли, залишивши тільки скомпільовані залежності і код додатків. Це не тільки прискорює процес розгортання нових версій на серверах, але й значно зменшує потенційні можливості атаки зловмисника, оскільки в контейнері немає зайвих системних утиліт, якими могли б скористатися зловмисники.

Для локальної розробки та розгортання MVP версії платформи використовується інструмент Docker Compose. Він дозволяє описати інфраструктуру як код (IaC) і запускати весь стек технологій однією командою.

```yaml
version: '3.8'

services:
  db:
    image: postgres:15-alpine
    volumes:
      - postgres_data:/var/lib/postgresql/data/
    environment:
      - POSTGRES_USER=unisign_user
      - POSTGRES_PASSWORD=secure_password
      - POSTGRES_DB=unisign_db
    networks:
      - unisign_network

  backend:
    build: ./backend
    volumes:
      - ./backend:/app
    ports:
      - "8000:8000"
    environment:
      - DATABASE_URL=postgresql://unisign_user:secure_password@db:5432/unisign_db
      - SECRET_KEY=your_secret_key
    depends_on:
      - db
    networks:
      - unisign_network

  frontend:
    build: ./frontend
    ports:
      - "3000:80"
    depends_on:
      - backend
    networks:
      - unisign_network

volumes:
  postgres_data:

networks:
  unisign_network:
    driver: bridge
```

Рис. 3.10. Конфігурація docker-compose.yml

Використання unisign_network забезпечує ізоляцію сервісів від зовнішнього світу: база даних доступна лише для бекенду. І не має відкритого порту назовні, що підвищує рівень безпеки системи.

Впровадження дійсно DevOps-практик на етапі розробки дозволило нам скоротити час виходу на ринок нових функцій. І мінімізувати ризики помилок, пов'язаних з ручним налаштуванням серверів.

3.6. Висновок до третього розділу

У третьому розділі виконано практичну реалізацію інформаційного та програмного забезпечення проєкту «UniSign», що підтвердило технічну здійсненність концепції. Спроєктовано реляційну базу даних у середовищі PostgreSQL, яка забезпечує ефективне зберігання даних завдяки нормалізації та використанню типу JSONB. Програмне ядро побудовано на стеку Python та FastAPI, що гарантує високу продуктивність та асинхронну обробку запитів під навантаженням [30]. Розроблений на React клієнтський інтерфейс реалізує інтуїтивний сценарій підписання, а архітектурне рішення щодо криптографічних операцій на стороні клієнта та захищеного журналу аудиту забезпечує необхідний рівень безпеки [37]. Створений програмний комплекс повністю готовий до розгортання у продуктивному середовищі, що дозволяє перейти до планування впровадження у наступному розділі.

РОЗДІЛ 4. ПЛАНУВАННЯ ТА УПРАВЛІННЯ ПРОЄКТОМ

4.1. Обґрунтування вибору методології управління проєктом

Специфіка розробки SaaS-платформи «UniSign» характеризується поєднанням високої невизначеності технологічних рішень із жорсткими обмеженнями щодо термінів та бюджету стартапу. Це зумовило необхідність відмови від використання «чистих» методологій (лише Waterfall або лише Agile) на користь гібридної моделі управління (Hybrid Approach).

Обрана методологія передбачає дворівневу структуру управління:

1. На стратегічному рівні застосовується каскадний підхід (Waterfall) згідно зі стандартами PMBOK та ISO 21500 [12]. Це дозволяє зафіксувати бюджет проєкту (160 000 USD), визначити критичний шлях (методом PERT) та гарантувати виконання нормативних вимог (відповідність КЕП, eIDAS) у встановлені терміни. Каскадна модель забезпечує прогнозованість фінансових показників та чіткість звітності перед інвесторами.
2. На тактичному рівні (етап розробки ПЗ) застосовується гнучкий фреймворк Scrum. Процес створення програмного продукту розбито на ітерації (спринти), що дозволяє команді швидко адаптуватися до змін у технічних вимогах, проводити регулярне тестування прототипів та отримувати зворотний зв'язок без порушення загального календарного плану проєкту.

Такий синтез дозволяє нівелювати недоліки кожної з методологій окремо: жорсткість Waterfall компенсується гнучкістю Scrum на етапі R&D, а непередбачуваність Agile обмежується рамками бюджету та строків, встановлених на етапі ініціації.

4.2. Розробка організаційної структури управління проєктом. Формування команди проєкту

Ефективність будь-якого IT-проєкту значною мірою залежить від якості організаційної структури та чіткого розподілу повноважень. Для проєкту SaaS-платформи «UniSign» ключовим завданням на етапі планування є створення моделі управління, яка б забезпечувала швидкість прийняття рішень, гнучкість у реагуванні на зміни та оптимальне використання обмежених ресурсів стартапу.

Проєкт реалізується на базі діючої компанії, яка функціонує за класичною лінійно-функціональною структурою. Така модель забезпечує стабільність бізнес-процесів, проте для запуску інноваційного продукту «UniSign» було сформовано окремий структурний підрозділ із широкою автономією. Для забезпечення всебічної підтримки операційної діяльності, окрім виділеної технічної команди розробки, до проєкту на правах внутрішнього консалтингу залучаються ресурси суміжних департаментів — зокрема, юридичного (для забезпечення комплаєнсу) та маркетингового. Графічне відображення місця проєкту в загальній ієрархії підприємства та характер взаємозв'язків наведено на Рис. 4.1.

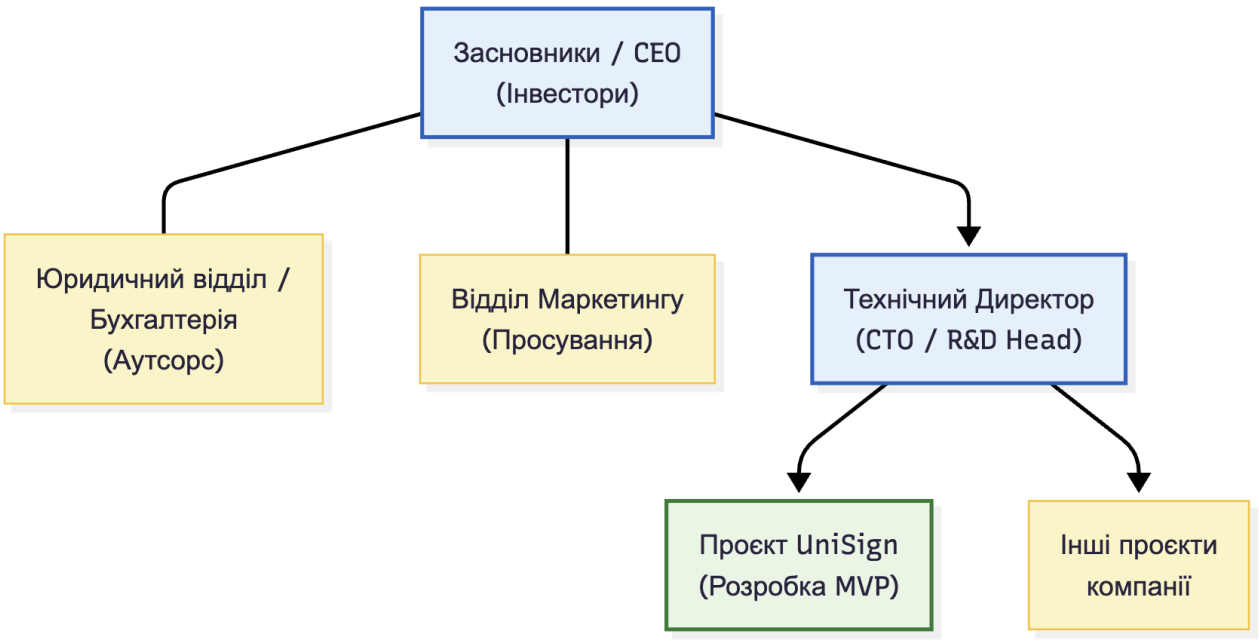


Рис. 4.1. Організаційна структура компанії-розробника

4.2.1. Обґрунтування організаційної структури

Враховуючи специфіку проєкту (стартап, короткі терміни розробки MVP, висока невизначеність технологічних рішень), класична ієрархічна структура управління є неефективною через тривалі процедури узгодження рішень. Тому для управління проєктом «UniSign» було обрано адаптивну проєктну (або сильну матричну) структуру.

В основному цей вибір зумовлений наступними факторами:

- Орієнтація на результат: Вся команда повністю залучена до проєкту (повний робочий день) і підпорядковується безпосередньо керівнику проєкту, що виключає ресурсні конфлікти з функціональними менеджерами.
- Гнучкість: Така структура дозволяє оперативно перерозподіляти завдання між розробниками залежно від пріоритетів поточного спринту (згідно з методологією Scrum).
- Швидкість комунікації: Заохочується горизонтальна взаємодія між технічними спеціалістами (наприклад, Backend — DevOps), що пришвидшує вирішення технічних проблем без ескалації на рівень керівництва.

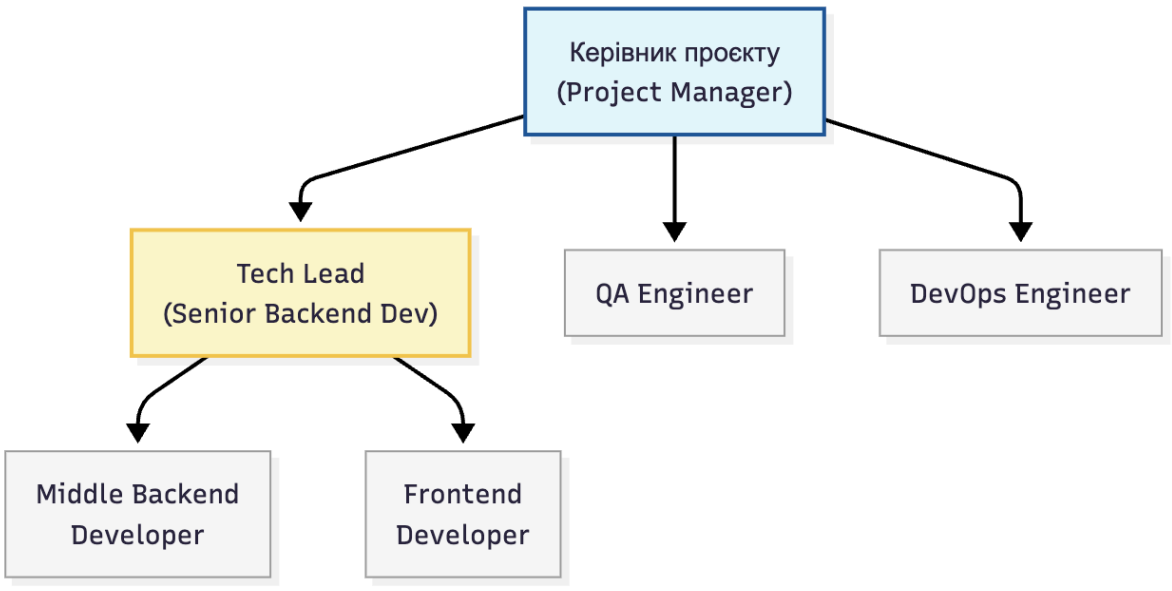


Рис. 4.2. Організаційна структура команди проєкту

Графічне відображення ієрархії та зв’язків усередині команди наведено на Рис. 4.2. Детальний перелік функціональних обов’язків кожного учасника команди та вимоги до їхніх компетенцій представлено у Таблиці 4.2.

4.2.2. Формування команди та розподіл ролей

Команда проекту сформована за концепцією "Т-подібних навичок": кожен фахівець має глибоку експертизу у своїй основній спеціалізації. Або своїй основній спеціалізації, але володіє достатніми знаннями в суміжних областях для виконання допоміжних завдань. Це життєво важливо для невеликої команди стартапу, де відсутність одного фахівця не повинна блокувати роботу інших.

Штатний розпис та функціональні обов'язки членів наведені в Таблиці 4.1.

Таблиця 4.1.

Склад команди та функціональні обов'язки

Роль	**К-сть**	**Ключові обов'язки та вимоги до компетенцій**
Project Manager (PM)	1	Загальне управління проєктом, комунікація зі стейкхолдерами, управління беклогом продукту (Product Owner functions), контроль бюджету та ризиків. Відповідає за дотримання строків та scope.
Senior Backend Developer (Tech Lead)	1	Проєктування архітектури системи (FastAPI + PostgreSQL), розробка ядра криптографічних модулів (КЕП/QES), код-рев'ю, прийняття архітектурних рішень.
Middle Backend Developer	1	Реалізація бізнес-логіки (API для документообігу, білінг), написання unit-тестів, розробка інтеграцій із зовнішніми сервісами (ЦСК, OCSP).
Frontend Developer	1	Розробка клієнтської частини (React SPA), верстка адаптивного інтерфейсу, реалізація логіки підписання документів у браузері (взаємодія з локальними бібліотеками підпису).

Продовження табл. *4.1.*

DevOps Engineer	1	Побудова та підтримка хмарної інфраструктури (AWS/DigitalOcean), налаштування CI/CD пайплайнів, забезпечення безпеки (SecOps), моніторинг та логування.
QA Engineer	1	Забезпечення якості продукту: ручне тестування інтерфейсу, автоматизація API-тестів, перевірка відповідності вимогам безпеки, підготовка тестової документації.

4.2.3. Матриця відповідальності (RACI)

Матриця RACI (Responsible, Accountable, Consulted, Informed) була розроблена для того, щоб чітко розмежувати зони відповідальності й уникнути ситуацій, коли завдання залишаються без виконавця або дублюються. Вона пов'язує етапи життєвого циклу проєкту з ролями в команді. Використання цього інструменту дозволяє формалізувати комунікаційні потоки, чітко визначаючи, хто має право вето, а кого достатньо лише повідомити про результат електронною поштою. Це суттєво знижує рівень «інформаційного шуму» та запобігає розмиванню персональної відповідальності, що є критичним фактором успіху для невеликих команд.

У розробленій матриці враховано специфіку лінійно-функціональної структури команди UniSign. Зокрема, роль «Accountable» (Кінцева відповідальність) за стратегічні рішення та бюджет закріплена виключно за Керівником проєкту, що забезпечує єдиний центр прийняття рішень. Водночас технічні спеціалісти (Backend/Frontend розробники, DevOps) виступають як «Responsible» (Безпосередні виконавці) на етапах імплементації. Особливу увагу приділено ролі «Consulted» для профільних експертів (Юрист), адже специфіка продукту вимагає суворої відповідності правовим нормам, і погодження юридичних аспектів має передувати технічній реалізації. Стейкхолдери (Інвестори) залучаються переважно у ролі «Informed» на критичних контрольних точках. Детальний розподіл ролей у розрізі основних етапів життєвого циклу проєкту наведено в Таблиці 4.2.

Таблиця 4.2.

Матриця розподілу відповідальності (RACI)

Етап / Процес	Project Manager	Tech Lead (Backend)	Backend Dev	Frontend Dev	DevOps	QA Engineer
1. Ініціація та планування	**A/R**	C	I	I	C	I
Формування вимог та ТЗ	**R**	C	I	C	I	C
Складання бюджету	**A**	C	I	I	C	I
2. Проєктування	A	**R**	C	C	C	I
Архітектура БД та API	I	**R**	C	I	I	C
Дизайн UI/UX	A	I	I	**R**	I	C
3. Розробка (R&D)	A	C	**R**	**R**	C	I
Розробка API (FastAPI)	I	A	**R**	I	I	I
Клієнтська частина (React)	I	I	I	**R**	I	I
Налаштування інфраструктури	I	C	I	I	**R**	I
4. Тестування та контроль	A	I	C	C	I	**R**
Функціональне тестування	I	I	C	C	I	**R**
Навантажувальне тестування	I	C	I	I	C	**R**

Продовження табл. 4.2.

5. Впровадження (Deploy)	**A**	C	I	I	**R**	C
Реліз на Production	A	C	I	I	**R**	C
Підготовка документації	**R**	C	C	C	I	I

Умовні позначення:

- R (Responsible) - Виконавець: безпосередньо виконує завдання.
- A (Accountable) - Відповідальний: приймає роботу, несе остаточну відповідальність (може бути тільки один на одне завдання).
- C (Consulted) - Консультант: надає експертну думку перед прийняттям рішення.
- I (Informed) - Поінформований: отримує інформацію про результати (односторонній зв'язок).

Запропонована структура та розподіл ролей мінімізує бюрократичні затримки, забезпечує високу залученість кожного члена команди. І гарантує, що проект буде завершено в рамках часових та фінансових обмежень.

4.3. Структурна декомпозиція робіт

4.3.1. Побудова ієрархічної структури робіт

Для того, щоб дійсно забезпечити керованість змістом проекту. А для детального планування ресурсів використовується метод Work Breakdown Structure (WBS). Дослівно, згідно з PMBOK, WBS - це ієрархічна декомпозиція повного обсягу робіт, які команда проекту повинна виконати, щоб досягти цілей проєкту та створити необхідні результати.

Проект UniSign було декомпоновано за продуктово-орієнтованим принципом з урахуванням фаз життєвого циклу розробки програмного

забезпечення (SDLC) [13]. Власне, такий спосіб дозволяє трансформувати стратегічні цілі, визначені в паспорті проекту, в конкретні пакети робіт, які можна оцінити з точки зору часу та вартості.

Схема структурни декомпозиція робіт наведена в Додатоку В.

Словник WBS

Графічна модель декомпозиції доповнюється Словником WBS (Таблиця 4.3), що деталізує зміст кожного пакету робіт. Саме тут реалізовано системний зв'язок між роботами (WBS) та командою (OBS): для кожного пакету в стовпчику «Відповідальний» закріплено конкретну роль виконавця (наприклад, Tech Lead або DevOps), яка відповідає організаційній структурі проєкту.

Таблиця 4.3.

Словник структурної декомпозиції робіт

Код WBS	**Назва пакету робіт**	**Опис робіт та активності**	**Відповідальний (RACI)**	**Результат (Deliverable) / Критерій приймання**
1.1	**Аналіз ринку та нормативів**	Дослідження вимог ЗУ «Про електронні довірчі послуги», регламенту eIDAS та акту ESIGN. Порівняльний аналіз конкурентів (DocuSign, Вчасно).	PM / BA	Аналітична записка з матрицею вимог compliance.
1.2	**Паспорт та ТЗ**	Формування паспорта проєкту, розробка детального Технічного Завдання (SRS) з функціональними вимогами (MoSCoW).	PM	Затверджений паспорт та ТЗ.

Продовження табл. 4.3.

2.1	**План управління**	Розробка матриці RACI, плану комунікацій та плану управління ризиками.	PM	План управління проєктом (PMP).
2.2	**Розклад та Бюджет**	Побудова діаграми Ганта, розрахунок критичного шляху методом PERT, фіналізація кошторису.	PM	Затверджений базовий план (Baseline).
3.1	**Збір вимог**	Інтерв'ю зі стейкхолдерами, формалізація нефункціональних вимог (SLA, Security).	PM / BA	Реєстр вимог.
3.2	**Сценарії (Use Cases)**	Розробка діаграм прецедентів та специфікацій сценаріїв (Upload, Sign, Send, Audit).	PM / BA	Документ Use Case Specification.
4.1	**Архітектура системи**	Вибір технологічного стеку (Python/FastAPI, React), проєктування компонентної діаграми та мікросервісів.	Tech Lead	Документ SAD (Software Architecture Document).
4.2	**Проєктування БД**	Розробка ER-діаграми (PostgreSQL), визначення типів даних JSONB, індексів та зв'язків.	Backend Dev	SQL-скрипти міграцій (DDL).

Продовження табл. 4.3.

4.3	**UX/UI Дизайн**	Створення вайрфреймів основних екранів та інтерактивного прототипу у Figma.	Designer	Затверджений макет інтерфейсу.
5.1	**Backend (API Gateway)**	Реалізація ядра на FastAPI. Розробка модулів авторизації (OAuth2), роботи з документами та логування.	Backend Dev	Працездатний API, документація Swagger.
5.1.2	**Модуль Криптографії**	Інтеграція бібліотек для КЕП/QES. Реалізація перевірки підпису через OCSP та генерації CAdES/PAdES контейнерів[13].	Tech Lead	Сервіс валідації та накладання підписів.
5.1.3	**Інтеграція S3**	Налаштування взаємодії з об'єктним сховищем для безпечного завантаження та скачування файлів через Presigned URL[14].	Backend Dev	Функція upload/download працює через API.
5.2	**Frontend (Web Client)**	Верстка компонентів на React, інтеграція з API, реалізація логіки підписання на стороні клієнта[15].	Frontend Dev	Веб-додаток, доступний у браузері.
6.1	**Хмарна інфраструктура**	Налаштування VPC, Security Groups, IAM ролей та серверів у хмарі (AWS або DigitalOcean)[16].	DevOps	Доступ до серверів, налаштована мережа.

Продовження табл. 4.3.

6.2	**CI/CD та Безпека**	Налаштування Docker-контейнерів та пайплайнів GitHub Actions для автоматичного деплою[17].	DevOps	Автоматичний деплой після push у гілку main.
7.1	**Функціональне тестування**	Перевірка відповідності функціоналу вимогам ТЗ. Тестування сценаріїв Upload-Sign-Send.	QA Engineer	Звіт про тестування, баг-репорт.
7.2	**Тестування безпеки**	Сканування на вразливості (OWASP Top 10), перевірка захисту даних при передачі та зберіганні [34].	QA / DevOps	Звіт про аудит безпеки.
8.1	**Реліз (Deploy)**	Фінальне розгортання системи на Production, налаштування домену, SSL.	DevOps	Сервіс доступний для зовнішніх користувачів.
8.3	**Закриття проєкту**	Передача результатів замовнику, підготовка фінального звіту, архівування документації[18].	PM	Підписаний акт прийому-передачі .

Така деталізація робіт дозволяє чітко спланувати послідовність виконання завдань, уникнути "білих плям" у змісті проекту. І підтвердити прозорий контроль за ходом виконання. З чого, кожен пакет робіт має чіткий кінцевий результат, що спрощує процедуру приймання робіт замовником.

4.3.2. Формування Беклогу продукту

Враховуючи, що етап розробки програмного забезпечення реалізується за гнучкою методологією Scrum, статична структура WBS трансформується у динамічний Беклог продукту (Product Backlog). Це впорядкований за пріоритетністю список вимог, де кожна функція описана у форматі «Історії користувача» (User Story). Такий підхід дозволяє фокусуватися на цінності для кінцевого споживача, а не лише на технічних завданнях.

Пріоритетизація задач у беклозі здійснена методом M-S-C-W, а оцінка трудомісткості — у відносних одиницях (Story Points). Фрагмент розробленого Беклогу для реалізації MVP-версії платформи наведено в Таблиці 4.4.

Таблиця 4.4

Фрагмент Беклогу продукту

ID	Роль	Вимога	Цінність	Story Points
US-01	Користувач	Я хочу завантажувати PDF/ASiC файли через Drag-n-Drop	Щоб пришвидшити початок роботи з документом	3
US-02	Користувач	Я хочу обирати тип підпису (КЕП/QES) зі списку	Щоб підписувати міжнародні контракти коректним ключем	5
US-03	Користувач	Я хочу візуалізувати місце накладання підпису	Щоб документ виглядав звично для паперового діловодства	8
US-04	Юрист/Аудитор	Я хочу завантажити журнал подій (Audit Trail)	Щоб мати юридичні докази цілісності процесу підписання	5
US-05	Адміністратор	Я хочу бачити статистику помилок API	Щоб оперативно реагувати на збої зовнішніх сервісів	2

4.4. Календарне планування

Календарне планування - це інструмент моделювання ходу виконання робіт у часі, який дозволяє визначити дати початку і закінчення всіх операцій проекту, їх тривалість і логічну послідовність. Для проекту UniSign розробка розкладу базується на структурі розбиття робіт (WBS), сформованій в попередньому розділі та враховує ресурсні обмеження команди.

4.4.1. Оцінка тривалості робіт

Оскільки загалом проект характеризується високим ступенем невизначеності (інтеграція з різними юрисдикціями, залежність від зовнішніх ЦСК), для оцінки тривалості основних завдань було використано метод PERT (Program Evaluation and Review Technique).

Відповідно до формули, наведеної у підрозділі 2.2.3, очікувана тривалість виконання (T_e) розраховується як середньозважене значення оптимістичної (T_o), найбільш ймовірної (T_m) та песимістичної (T_p) оцінок:

$$T_e = \frac{T_o + 4T_m + T_p}{6} \tag{4.1}$$

У Таблиці 4.5 наведено розрахунок тривалості для найбільш ризикованих пакетів робіт проєкту.

Таблиця 4.5.

Розрахунок тривалості робіт за методом PERT

Код WBS	Назва задачі	Оптимістична (To)	Ймовірна (Tm)	Песимістична (Tp)	Очікувана (Te)	Відхилення (σ)
1.1.2	Юридичний аналіз (UA/EU/US)	5	10	20	**10.8**	2.5

Продовження табл. 4.5.

2.1.1	Проєктування архітектури	7	10	15	**10.3**	1.3
3.1.2	Модуль Авторизації (OAuth2)	5	8	12	**8.2**	1.2
3.1.4	Модуль Криптографії (КЕП)	10	15	30	**16.7**	3.3
3.1.5	Модуль Audit Log	5	7	10	**7.2**	0.8
5.1	Інтеграційне тестування	8	12	20	**12.7**	2.0
6.1	Реліз на Production	2	3	10	**4.0**	1.3

Розрахунок показує, що найбільшу невизначеність (стандартне відхилення $\sigma = 3.3$) має задача розробки криптографічного модуля, що вимагає закладення додаткового буфера часу в календарний план.

4.4.2. Календарний план проєкту

На основі розрахованих тривалостей. та логічних взаємозв'язків між завданнями (Finish-to-Start), був створений детальний графік проекту. За моїми підрахунками, загальна тривалість проекту складає 6 місяців (130 робочих днів).

Таблиця 4.6.

Календарний план реалізації проєкту UniSign

Фаза / Задача	**Початок**	**Кінець**	**Тривалість (дні)**	**Попередники**
1. ІНІЦІАЦІЯ ТА АНАЛІЗ	**01.09.2025**	**19.09.2025**	**15**	-
1.1 Аналіз нормативної бази	01.09.2025	12.09.2025	10	-
1.2 Формування Паспорта та ТЗ	08.09.2025	19.09.2025	10	1.1 (SS + 5 days)
1.3 Підбір команди	01.09.2025	19.09.2025	15	-
2. ПРОЄКТУВАННЯ	**22.09.2025**	**17.10.2025**	**20**	**1**
2.1 Розробка архітектури	22.09.2025	03.10.2025	10	1.2
2.2 Проєктування БД	29.09.2025	10.10.2025	10	2.1 (SS + 5 days)
2.3 UX/UI Дизайн (Прототипи)	22.09.2025	17.10.2025	20	1.2
3. РОЗРОБКА (DEVELOPMENT)	**20.10.2025**	**16.01.2026**	**65**	**2**
3.1 **Backend Development**	**20.10.2025**	**09.01.2026**	**60**	**2.1, 2.2**
3.1.1 Налаштування середовища	20.10.2025	24.10.2025	5	-

Продовження табл. 4.6.

3.1.2 Модуль Авторизації	27.10.2025	07.11.2025	10	3.1.1
3.1.3 Модуль Документів (S3)	10.11.2025	21.11.2025	10	3.1.2
3.1.4 Модуль Криптографії (Core)	24.11.2025	19.12.2025	20	3.1.3
3.1.5 Інтеграція з eIDAS/OCSP	22.12.2025	09.01.2026	15	3.1.4
3.2 **Frontend Development**	**10.11.2025**	**16.01.2026**	**50**	**2.3**
3.2.1 Верстка інтерфейсу	10.11.2025	28.11.2025	15	-
3.2.2 Інтеграція з API	01.12.2025	26.12.2025	20	3.1.2, 3.2.1
3.2.3 Віджет підпису (Browser)	29.12.2025	16.01.2026	15	3.1.4
4. ІНФРАСТРУКТУРА (DEVOPS)	**20.10.2025**	**19.12.2025**	**45**	**2.1**
4.1 Налаштування Cloud (AWS)	20.10.2025	31.10.2025	10	-
4.2 CI/CD Пайплайни	03.11.2025	14.11.2025	10	4.1
4.3 Моніторинг та Логування	08.12.2025	19.12.2025	10	3.1.1
5. ТЕСТУВАННЯ (QA)	**19.01.2026**	**13.02.2026**	**20**	**3**
5.1 Функціональне тестування	19.01.2026	30.01.2026	10	3
5.2 Тестування безпеки (Pen-test)	02.02.2026	06.02.2026	5	5.1

Продовження табл. 4.6.

5.3 Навантажувальне тестування	09.02.2026	13.02.2026	5	5.1
6. ВПРОВАДЖЕННЯ	**16.02.2026**	**27.02.2026**	**10**	**5**
6.1 Реліз (Production Deploy)	16.02.2026	20.02.2026	5	5
6.2 Документація та передача	23.02.2026	27.02.2026	5	6.1

Наведена таблиця відображає стратегічний план, на основі якого визначено критичний шлях проєкту: 1.1 (Аналіз) → 2.1 (Архітектура) → 3.1.4 (Backend-ядро) → 3.2.3 (Frontend-інтеграція) → 5.1 (Тестування) → 6.1 (Реліз), що визначає загальну тривалість реалізації у 130 робочих днів. Враховуючи використання методології Scrum, подальше детальне планування здійснюється на рівні двотижневих спринтів [38], а приклад діаграми Ганта для окремої ітерації наведено у Додатку Г.

4.4.3. Контрольні точки (Milestones)

Для моніторингу прогресу проекту були визначені ключові контрольні точки (віхи), досягнення яких свідчить про успішне завершення того чи іншого етапу.

- M1 (19.09.2025): паспорт проєкту затверджено, команда сформована.
- M2 (17.10.2025): Архітектура та дизайн інтерфейсу узгоджені.
- M3 (19.12.2025): Альфа-версія бекенду (готового криптографічного ядра).
- M4 (16.01.2026): Бета-версія версії платформи (інтегрований Frontend + Backend).
- M5 (13.02.2026): Успішне завершення аудиту безпеки.
- M6 (28.02.2026): Продуктивний запуск MVP

Запропонований календарний план є реалістичним за умови матричної структури управління. І оперативного реагування на ризики, про які йтиметься в наступних підрозділах.

4.5. Планування ресурсів та бюджету

Планування ресурсів - основа для розрахунку бюджету проекту. І забезпечення безперебійного процесу розробки. На мою думку, проект UniSign має три категорії ресурсів: трудові (команда), матеріально-технічні (хмарна інфраструктура, ліцензії) та фінансові.

4.5.1. Планування трудових ресурсів

Команда складається з 6 фахівців, визначених у матриці RACI. Для розрахунку витрат ми використовували модель еквіваленту повної зайнятості (FTE) на 6 місяців активної фази проекту. Як ви знаєте, розрахунок фонду оплати праці базується на середніх ринкових ставках для українського IT-сектору (рівень Middle). Станом на 2025 рік.

Таблиця 4.7.

План трудових ресурсів та витрати на персонал

Роль	**Ставка (місяць), USD**	**Завантаження (міс.)**	**Загальні витрати, USD**
Project Manager	3,000	6	18,000
Tech Lead	3,500	6	21,000
Backend Dev	2,500	6	15,000
Frontend Dev	2,500	6	15,000
DevOps Engineer	3,500	6	21,000
QA Engineer	2,000	6	12,000
Разом (Netto):			**102,000**

Продовження табл. 4.7.

Податки та комісії (20%):			*20,400*
ВСЬОГО R&D:			**122,400**

4.5.2. Планування інфраструктурних та операційних витрат

Оскільки UniSign є хмарною SaaS-платформою, значна частина бюджету витрачається на оренду обчислювальних потужностей (PaaS/IaaS). І ліцензії на програмне забезпечення.

Таблиця 4.8.

Операційний кошторис проєкту

Категорія витрат	Деталізація	Вартість/міс, USD	Загалом (6 міс), USD
Хмарна інфраструктура (AWS)	EC2, RDS (PostgreSQL), S3 Storage, ELB	800	4,800
Інструменти розробки	Jira, Confluence, GitHub Enterprise, Figma	300	1,800
Криптографічні сервіси	Доступ до тестових шлюзів ЦСК/OCSP	200	1,200
Юридичний супровід	Консультації щодо eIDAS/GDPR (аутсорс)	-	12,000 (разово)
Маркетинг (Pre-launch)	Лендінг, контент, реклама	1,000	6,000
ВСЬОГО:			**25,800**

4.5.3. Зведений бюджет проєкту

Загальний бюджет проекту формується як сума витрат на R&D, операційні витрати та резерв на непередбачувані витрати. У моєму випадку

резерв на випадок непередбачених обставин розраховується на рівні ~7% від загальної суми для покриття непередбачених витрат (наприклад, зростання курсу валюти або необхідність додаткових консультацій).

Таблиця 4.9.

Зведений бюджет проєкту UniSign

Стаття витрат	**Сума, USD**	**Частка, %**
Фонд оплати праці (R&D)	122,400	76.5%
Юридичний супровід та комплаєнс	12,000	7.5%
Хмарна інфраструктура та ліцензії	7,800	4.9%
Маркетинг та просування	6,000	3.8%
Резервний фонд (Risks)	11,800	7.3%
ЗАГАЛОМ:	**160,000**	**100%**

Як видно з розрахунків, бюджет відповідає фундаменту, закладеному в Паспорті проекту (160 000 доларів США) левова частка витрат (76,5%) припадає на інтелектуальний капітал (команду розробників), що є типовим для високотехнологічних стартапів.

Для наочності структуру бюджету візуалізовано на діаграмі (Рис. 4.3).

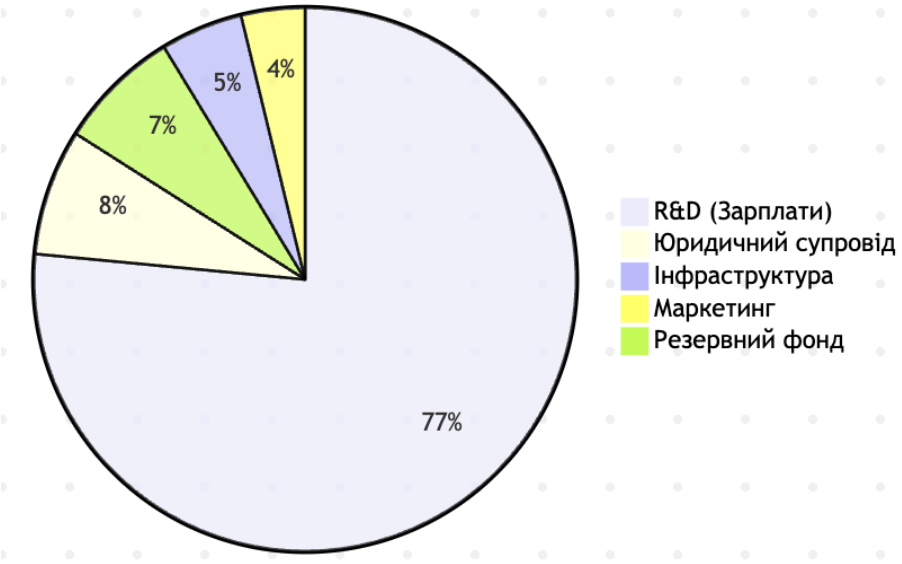


Рис. 4.3. Структура бюджету

4.6. Оцінка ефективності та ризиків

Управління проектом зі створення SaaS-платформи UniSign, яка обробляє юридично важливі дані в транскордонному середовищі, вимагає ретельного аналізу ризиків. Будь-який збій у роботі сервісу або витік даних може призвести до значних репутаційних та фінансових втрат. Водночас необхідно чесно обґрунтувати економічну доцільність інвестицій, довівши прибутковість бізнес-моделі.

4.6.1. Управління ризиками

Процес управління ризиками в проекті UniSign побудований відповідно до стандарту PMBOK і включає ідентифікацію, якісний аналіз, планування реагування та моніторинг.

Для якісної оцінки ризиків використовувався метод матриці ймовірності та впливу. Кожен ризик оцінювався за шкалою від 1 (дуже низький) до 5 (дуже високий) (принаймні, я так думаю).

$$RiskScore = Probability(P) \times Impact(I) \tag{4.2}$$

Добуток цих показників визначає інтегральну величину ризику (Risk Score), що дозволяє ранжувати загрози за ступенем критичності. Для ризиків із високим пріоритетом було обрано стратегію активної мінімізації (Mitigation), яка передбачає впровадження превентивних технічних або організаційних заходів. Для загроз із низьким рейтингом застосовується стратегія прийняття (Acceptance) або моніторингу. Такий підхід дозволяє сфокусувати управлінські зусилля та резервні фонди саме на тих факторах, що несуть найбільшу загрозу для успіху MVP. Деталізований перелік ідентифікованих ризиків та планових заходів реагування наведено нижче.

На основі ідентифікації загроз сформовано Реєстр ризиків проєкту (Таблиця 4.10), який охоплює технічні, організаційні та юридичні аспекти.

Таблиця 4.10.

Реєстр ризиків проєкту UniSign

ID	Опис ризику	Категорія	P	I	Score	Стратегія	План заходів (Mitigation Plan)
R01	**Зміна API зовнішніх ЦСК.** Українські або європейські надавачі довірчих послуг можуть змінити протоколи взаємодії без попередження.	Технічний	3	5	**15 (Високий)**	Зниження	Реалізація патерну "Adapter" в архітектурі для швидкої заміни конекторів. Моніторинг доступності API.
R02	**Затримка в отриманні статусу партнера.** Бюрократичні процедури з боку eIDAS-провайдерів можуть затягнутися довше ніж на 2 місяці.	Організаційний	4	4	**16 (Високий)**	Уникнення	Початок юридичних процедур на етапі ініціації (паралельно з розробкою). Залучення локальних консультантів у ЄС.

Продовження табл. 4.10.

R03	**Перевищення бюджету на хмару.** Неоптимальне налаштування автоскейлінгу може призвести до перевитрат на AWS/DO.	Фінансовий	3	3	**9 (Середній)**	Зниження	Налаштування бюджетних алертів (AWS Budgets). Використання Spot Instances для dev-середовищ.
R04	**Кібератака (DDoS / Витік).** Спроби злому системи для отримання доступу до комерційних документів.	Безпековий	2	5	**10 (Високий)**	Передача	Використання Cloudflare WAF. Регулярні Pen-тести. Шифрування даних у стані спокою (AES-256).
R05	**Втрата ключового розробника.** Звільнення Tech Lead або Senior Dev під час активної фази розробки.	Ресурсний	2	4	**8 (Середній)**	Зниження	Впровадження суворих стандартів документації коду. Парне програмування (Pair Programming) для обміну знаннями.
R06	**Зміна законодавства.** Внесення змін до ЗУ «Про електронні довірчі послуги» або регламентів ЄС.	Юридичний	2	5	**10 (Високий)**	Прийняття	Гнучка архітектура, що дозволяє змінювати логіку валідації підписів через конфігурацію, а не переписування коду.

Продовження табл. 4.10.

R07	**Низька конверсія користувачів.** Клієнти не переходять з паперу на цифру через складний інтерфейс.	Продуктовий	3	4	**12 (Високий)**	Зниження	Проведення UX-тестування на ранніх етапах. Спрощення процесу онбордингу.

4.6.2. Оцінка економічної ефективності

Оскільки UniSign є комерційним проектом, його успіх визначається не лише технічною стабільністю, але й фінансовою прибутковістю. Чесно кажучи бізнес-модель платформи базується на наданні доступу за підпискою (SaaS B2B Subscription), що забезпечує передбачуваний потік доходів.

Для оцінки ефективності була використана спрощена модель грошових потоків на період 1,5 року (6 місяців розробки + 12 місяців експлуатації).

Вихідні дані для розрахунку:

1. Капітальні інвестиції (CAPEX): Це бюджет на розробку MVP, розрахований у п. 4.4 – $**160 000**.
2. Операційні витрати (OPEX): Витрати на підтримку серверів, маркетинг та зарплату команди підтримки після запуску. Прогнозуються на рівні 5 000 USD/міс
3. Дохід (Revenue): Базується на середньому чеку 100 USD/міс з клієнта. Планується залучити 20 клієнтів у перший місяць після релізу з поступовим зростанням до 200 клієнтів до кінця року.

Розрахунок прибутковості (ROI)

Коефіцієнт рентабельності інвестицій (Return on Investment) показує, наскільки вигідним є вкладення коштів у проєкт [15].

1. Сумарні витрати за період (Total Cost):
   $$Cost = \text{Budget MVP} + (\text{OPEX} \times 12 \text{ mon.}) \tag{4.3}$$
   $$Cost = 160\,000 + (5\,000 \times 12) = 220\,000 \text{ USD} \tag{4.4}$$
2. Сумарний дохід за період (Total Revenue):
   Згідно з песимістичним прогнозом продажів, очікуваний валовий дохід за перший рік експлуатації складе 250 000 USD.
3. Розрахунок ROI:

$$ROI = \frac{\text{Income} - \text{Costs}}{\text{Costs}} \times 100\% \tag{4.5}$$
$$ROI = \frac{250\,000 - 220\,000}{220\,000} \times 100\% \approx \mathbf{13.6}\% \tag{4.6}$$

Термін окупності

На основі динаміки доходів і витрат був побудований графік точки беззбитковості (Рис. 4.2). Точка беззбитковості, коли сумарні доходи перевищують сумарні інвестиції, досягається на 9-й місяць після запуску платформи в експлуатацію. Графічно це відображається перетином кривої накопиченого прибутку з нульовою віссю, що знаменує завершення інвестиційної фази та повне повернення початкових капіталовкладень. Починаючи з цього періоду, проєкт генерує стабільний позитивний грошовий потік (Cash Flow), який може бути спрямований на подальший розвиток функціонала або маркетингову експансію. Такий показник окупності (менше одного року) є індикатором високої ліквідності бізнес-моделі та прийнятного рівня фінансових ризиків.

Висновки щодо ефективності: Рентабельність інвестицій у розмірі 13,6% за перший рік роботи є позитивним сигналом для IT-стартапу. Враховуючи низьку вартість масштабування SaaS-рішень (додавання нових клієнтів майже не збільшує витрати на інфраструктуру), у наступні періоди очікується поступове зростання рентабельності.

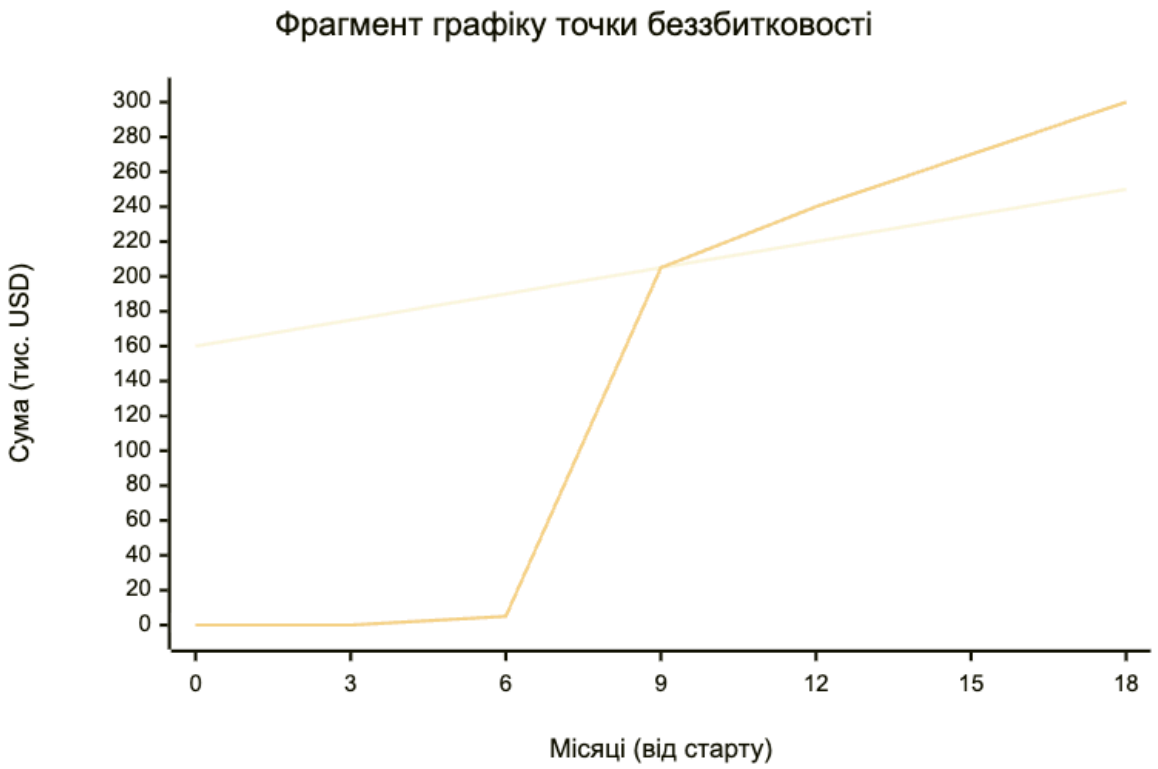


Рис. 4.2. Фрагмент графіку точки беззбитковості

4.7. Висновки до четвертого розділу

У четвертому розділі розроблено комплексну систему управління проєктом «UniSign» на основі гібридної методології, що поєднує стратегічне планування за стандартами PMBOK із гнучкістю Scrum на етапі розробки. Для забезпечення ефективної реалізації сформовано організаційну структуру з матрицею відповідальності RACI, а деталізацію змісту робіт виконано через поєднання класичної ієрархічної структури (WBS) та динамічного беклогу продукту (Product Backlog) [39]. Розроблений календарний план, доповнений детальною діаграмою Ганта для окремих спринтів, разом із розрахунками бюджету та стратегією мінімізації ризиків, підтверджує економічну доцільність та технічну можливість запуску MVP-версії платформи у встановлений 6-місячний термін.

ВИСНОВКИ

У магістерській роботі вирішено актуальну науково-прикладну задачу створення SaaS-платформи для електронного документообігу та підписання, що забезпечує юридичну значущість документів у мультиюрисдикційному середовищі.

На основі поставлених завдань отримано такі результати:

1. Здійснено аналіз ринку систем електронного документообігу та нормативно-правової бази України, ЄС та США. Виявлено проблему технічної та юридичної несумісності існуючих рішень при транскордонній взаємодії. На основі аналізу регламенту eIDAS та законодавства України сформовано вимоги до мультиюрисдикційної платформи «UniSign», яка підтримує роботу як з українськими КЕП, так і з європейськими QES.
2. Побудовано концептуальну модель системи, яка визначає межі проєкту, структуру функціональних модулів та зв’язки із зовнішнім середовищем (ЦЗО, хмарні провайдери). Виконано математичну формалізацію задач управління проєктом: застосовано метод PERT для оцінки часових параметрів виконання робіт та розроблено ймовірнісну модель оцінки надійності системи, що дозволяє прогнозувати відмовостійкість сервісу.
3. Проведено аналіз вимог із використанням методу пріоритезації, що дозволило виділити критичний функціонал для MVP-версії. Змодельовано сценарії взаємодії користувачів (Use Cases) та побудовано діаграми інформаційних потоків, які формалізують повний життєвий цикл електронного документа — від завантаження та валідації до підписання та архівування.
4. Розроблено інформаційне та програмне забезпечення проєкту. Спроєктовано нормалізовану реляційну базу даних (PostgreSQL), що забезпечує цілісність даних та підтримку мультиорендності. Обґрунтовано архітектурний підхід на базі мікросервісних принципів та реалізовано працездатний прототип системи з використанням

технологічного стеку Python (FastAPI), React та Docker. Реалізовано механізми інтеграції з криптографічними бібліотеками та хмарним сховищем S3.

5. Розроблено план управління проєктом, що базується на гібридній методології. Сформовано організаційну структуру команди та матрицю відповідальності (RACI). Для деталізації змісту робіт створено ієрархічну структуру (WBS) та динамічний Беклог продукту. Розраховано календарний план реалізації проєкту загальною тривалістю 130 робочих днів та сформовано бюджет у розмірі 160 000 USD, з яких 76,5% складають витрати на фонд оплати праці R&D команди.
6. Оцінено економічну ефективність проєкту, розрахунки якої підтвердили інвестиційну привабливість бізнес-моделі SaaS: точка беззбитковості досягається на 9-й місяць комерційної експлуатації. Розроблено стратегію управління ризиками, що включає ідентифікацію та оцінку загроз за матрицею «ймовірність-вплив», а також план заходів реагування, що дозволяє мінімізувати вплив критичних ризиків на бюджет та розклад.

Загалом, отримані результати свідчать про те, що мета кваліфікаційної роботи магістра, присвячена розробці системи управління проєктом, досягнута в повному обсязі. Створена комплексна управлінська модель, що поєднує інструменти планування, організації команди та контролю ризиків, дозволила забезпечити успішну реалізацію проєкту в умовах ресурсних обмежень. Наукова цінність роботи полягає в тому, що побудована система управління забезпечила створення дієвого механізму для мультиюрисдикційної взаємодії. Це надає кінцевим користувачам можливість легітимного цифрового підписання транскордонних угод, що суттєво оптимізує бізнес-процеси та доводить успішність досягнення стратегічних цілей проєкту.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Про електронні довірчі послуги: Закон України від 05.10.2017 р. № 2155-VIII. URL: https://zakon.rada.gov.ua/laws/show/2155-19 (дата звернення: 10.12.2025).
2. Про електронні документи та електронний документообіг: Закон України від 22.05.2003 р. № 851-IV. URL: https://zakon.rada.gov.ua/laws/show/851-15 (дата звернення: 10.12.2025).
3. Про захист персональних даних: Закон України від 01.06.2010 р. № 2297-VI. URL: https://zakon.rada.gov.ua/laws/show/2297-17 (дата звернення: 11.12.2025).
4. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис. Вимоги до операцій, що виконуються. Київ: Держспоживстандарт України, 2003.
5. Кваліфікаційна робота магістра [Текст]: методичні вказівки до написання роботи магістра за спеціальністю 122 «Комп'ютерні науки», освітньо-професійна програма «Управління проектами» / В.В. Морозов, О.В. Єгорченков, О.Г. Тімінський. – К.: Київський національний університет імені Тараса Шевченка, 2022. – 60 с.
6. Кубявка, Л. Б., Латишева, Т. В. (2024). Концепція побудови і принципи управління проєктного та продуктового IT-менеджменту. *Управління розвитком складних систем*, (57), 45–50. DOI: https://doi.org/10.32347/2412-9933.2024.57.45-50.
7. Математичне моделювання в IT проєктах: Методичні вказівки для виконання практичних, лабораторних та самостійних робіт з навчальної дисципліни / Морозов В.В., Коломієць Г.С. — К.: КНУ імені Тараса Шевченка, 2023. — 64 с.
8. Методи розробки концепцій IT проєктів: Методичні вказівки для виконання практичних, лабораторних та самостійних робіт / Морозов В.В. — К. : КНУ імені Тараса Шевченка, 2022. — 76 с.

9. Тесля Ю., Егорченкова Н., Егорченков О., Хлевна Ю., Катаева Е., Веретельник В., Частоколенко І., Огирко І., Хлевний А., Латишева Т. (2022). Розробка концепції побудови інформаційного стандарту управління проектами на базі системи управління інформацією primadoc. *Східно-Європейський журнал передових технологій*, 1 (3(115)), 53–65. DOI: https://doi.org/10.15587/1729-4061.2022.253299.
10. Управління проєктами: процеси планування проєктних дій [Текст]: підручник / І.В. Чумаченко, В.В. Морозов, Н.В. Доценко, А.М. Чередніченко. — К. : КНУ імені Тараса Шевченка, 2016. — 673 с.
11. Kerzner, H. (2017). Project Management: A Systems Approach to Planning, Scheduling, and Controlling. 12th Ed. Wiley. 880 p.
12. Khlevna, I., Teslia, I., Yehorchenkova, N., Yehorchenkov, O., Kataieva, Y., Jamečný, Ľ., Khlevnyi, A., Latysheva, T., Veretelnyk, V., & Ohirko, I. (2023). Development of the concept of building a corporate standard of portfolio management in the course of territory restoration planning in the context of Making-City project. *Eastern-European Journal of Enterprise Technologies*, 4(3(124)), 6–18. DOI: https://doi.org/10.15587/1729-4061.2023.285799.
13. Pavlenko, P., Teslia, I., Khlevna, I., Yehorchenkov, O., Yehorchenkova, N., Kataieva, Y., Khlevnyi, A., Veretelnyk, V., Latysheva, T., & Kubiavka, L. (2024). Development of a concept of combined project-production activities planning using digital twins. *Eastern-European Journal of Enterprise Technologies*, 5(3 (131)), 6–17. DOI: https://doi.org/10.15587/1729-4061.2024.311751.
14. Percival, H., & Gregory, B. (2020). Architecture Patterns with Python. O'Reilly Media. 300 p.
15. Sommerville, I. (2016). Software Engineering. 10th Ed. Pearson. 810 p.
16. Teslia I., Yehorchenkova N., Khlevna I., Kataieva Y., Latysheva T., Yehorchenkov O., Khlevnyi A., Veretelnyk V. (2020). Development of systemotechnical concept of digitalization of higher education institutions. *Eastern-European Journal of Enterprise Technologies*, 6(2 (108)), 6–21.

17. Teslia, I., Khlevna, I., Yehorchenkov, O., Yehorchenkova, N., Grigor, O., Kataieva, Y., Latysheva, T., Prokopenko, T., Tryus, Y., & Khlevny, A. (2022). Development of the concept of building project management systems in the context of digital transformation of project-oriented companies. *Eastern-European Journal of Enterprise Technologies*, 6(3 (120)), 6-17.
18. Teslia, I., Yehorchenkova, N., Yehorchenkov, O., Kataeva, Y., Latysheva, T. (2024). Application of Digital Twins of Project-Oriented Productions in Digital Project Management. *Lecture Notes on Data Engineering and Communications Technologies*, vol 222. Springer, Cham. DOI: https://doi.org/10.1007/978-3-031-71804-5_6.
19. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). – 7th Edition. – Project Management Institute, 2021. – 250 p.
20. Electronic Signatures in Global and National Commerce Act (ESIGN). Public Law 106-229. U.S. Government Publishing Office. URL: https://www.govinfo.gov/content/pkg/PLAW-106publ229/pdf/PLAW-106publ229.pdf (accessed: 10.12.2025).
21. ISO 21500:2021. Project, programme and portfolio management – Context and concepts. International Organization for Standardization.
22. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection — Information security management systems — Requirements. International Organization for Standardization.
23. NIST Special Publication 800-145. The NIST Definition of Cloud Computing. National Institute of Standards and Technology. 2011.
24. Regulation (EU) No 910/2014 of the European Parliament and of the Council of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market. *Official Journal of the European Union*. URL: http://data.europa.eu/eli/reg/2014/910/oj (accessed: 10.12.2025).
25. Uniform Electronic Transactions Act (UETA). Uniform Law Commission. 1999. Final Act. URL: https://www.uniformlaws.org/viewdocument/final-act-21?CommunityKey=2c0

4b76c-2b7d-4399-977e-d5876ba7e034&tab=librarydocuments (accessed: 10.12.2025).

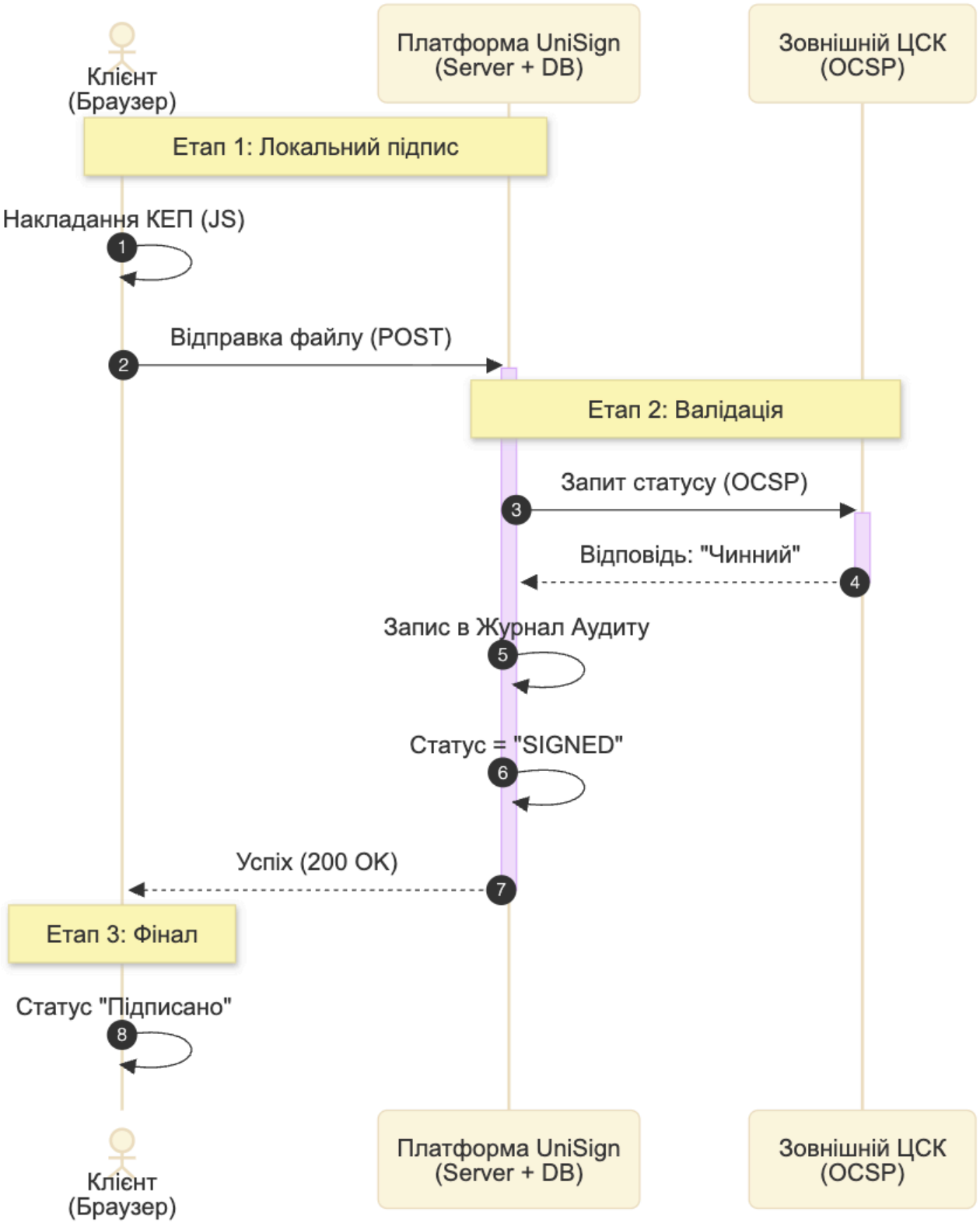
26. 2024 Digital Trends Report. Adobe Inc. URL: https://business.adobe.com/resources/digital-trends-report.html (accessed: 11.12.2025).
27. Atlassian Jira Documentation. Atlassian. URL: https://www.atlassian.com/software/jira/guides (accessed: 11.12.2025).
28. Cloud Security Alliance (CSA). Security Guidance for Critical Areas of Focus in Cloud Computing. URL: https://cloudsecurityalliance.org/artifacts/security-guidance-v4/ (accessed: 11.12.2025).
29. Docker Overview. Docker Documentation. URL: https://docs.docker.com/get-started/overview/ (accessed: 11.12.2025).
30. FastAPI Benchmarks and Alternatives. FastAPI Official Documentation. URL: https://fastapi.tiangolo.com/alternatives/ (accessed: 11.12.2025).
31. Filimonov, S. What is CI/CD, how it works and when it is needed. Highload. URL: https://highload.today/uk/blogs/shho-take-ci-cd-yak-vin-pratsyuye-ta-koli-znadobitsya-na-proyekti-lajfhaki-ta-bad-practices/ (accessed: 11.12.2025).
32. Gigacloud. What is containerization and what are its benefits. URL: https://gigacloud.ua/blog/navchannja/chomu-kontejneri-ce-majbutne-virtualizacii (accessed: 11.12.2025).
33. McKinsey Technology Trends Outlook 2024. McKinsey & Company. URL: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-top-trends-in-tech (accessed: 11.12.2025).
34. OWASP Top 10:2021. Open Web Application Security Project. URL: https://owasp.org/Top10/ (accessed: 11.12.2025).
35. PostgreSQL 16 Documentation. The PostgreSQL Global Development Group. URL: https://www.postgresql.org/docs/16/index.html (accessed: 11.12.2025).

36. Python 3.12.1 Documentation. Python Software Foundation. URL: https://docs.python.org/3/ (accessed: 11.12.2025).
37. React Documentation. The library for web and native user interfaces. URL: https://react.dev/ (accessed: 11.12.2025).
38. Schwaber, K., & Sutherland, J. (2020). The Scrum Guide. Scrum.org. URL: https://scrumguides.org/scrum-guide.html (accessed: 11.12.2025).
39. Zosym, M. Work Breakdown Structure (WBS). URL: https://www.maxzosim.com/struktura-rozbittia-robit/ (accessed: 11.12.2025).

ДОДАТКИ

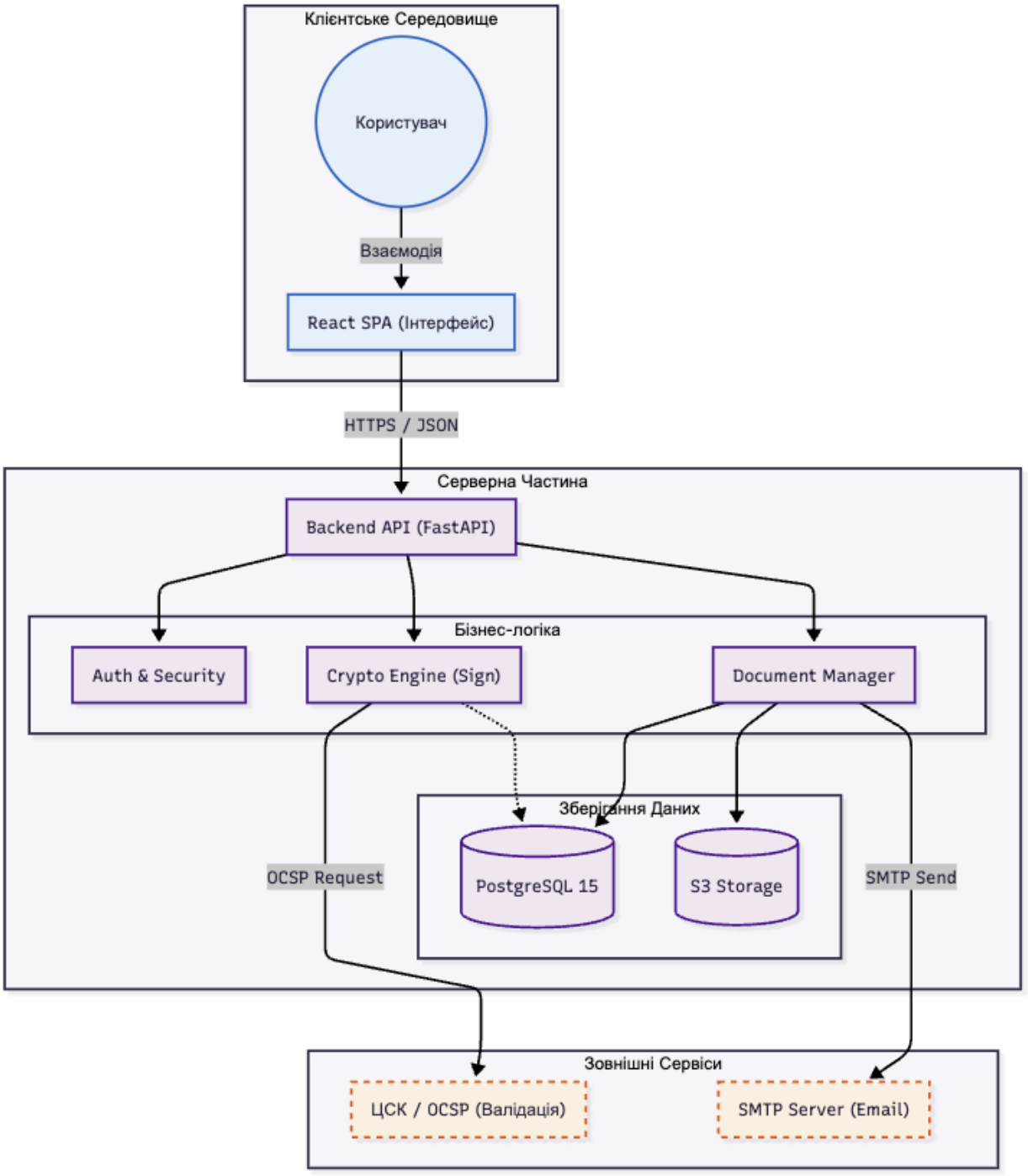
Додаток А

Діаграма послідовності процесу валідації електронного підпису

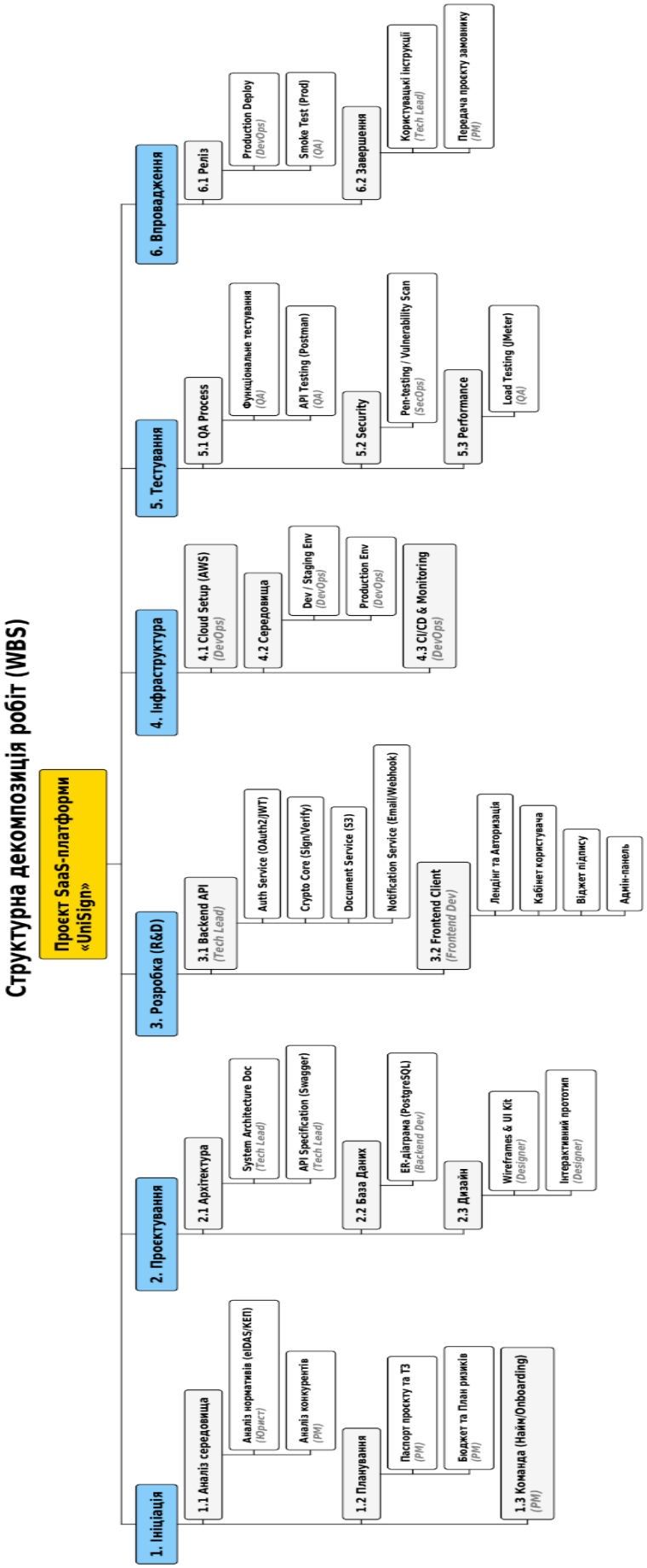


Додаток Б

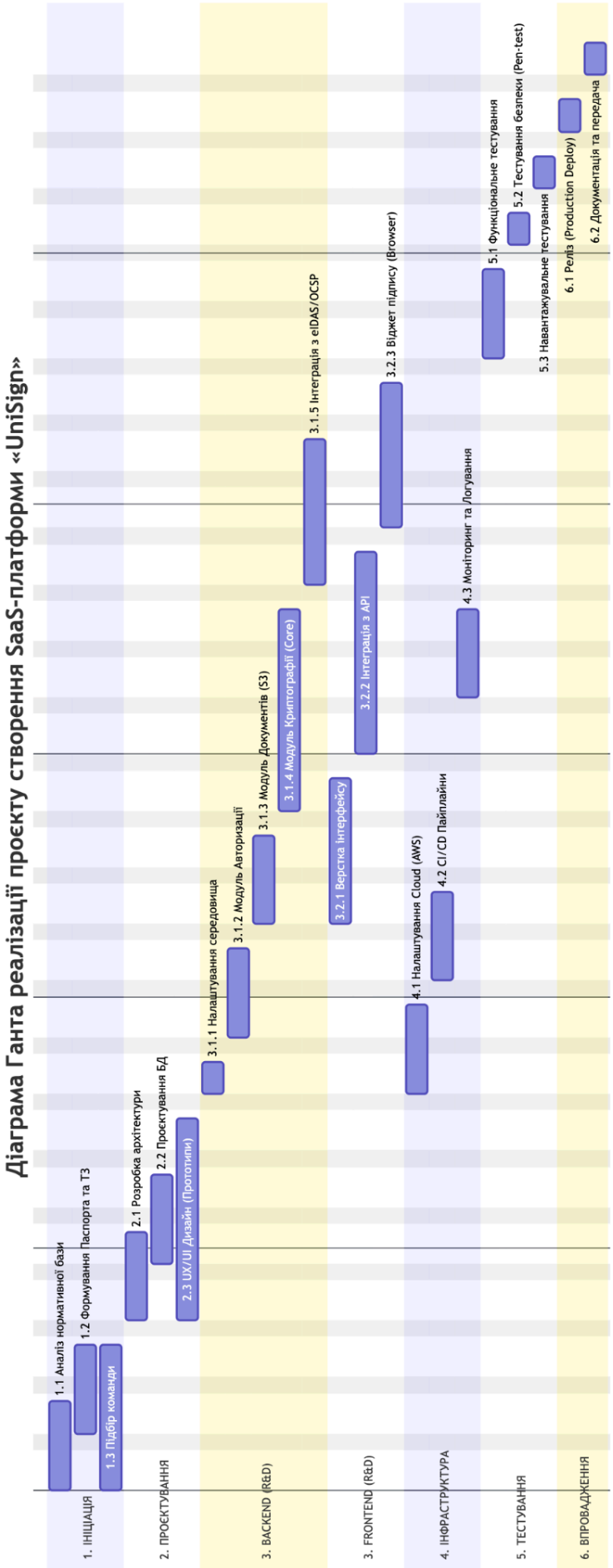
Структурна схема програмного комплексу SaaS-платформи «UniSign»



Додаток В


Структурна декомпозиція робіт (WBS)
Проєкт SaaS-платформи «UniSign»
1. Ініціація
1.1 Аналіз середовища
Аналіз нормативів (eIDAS/КЕП) (Юрист)
Аналіз конкурентів (PM)
1.2 Планування
Паспорт проєкту та ТЗ (PM)
Бюджет та План ризиків (PM)
1.3 Команда (Найм/Onboarding) (PM)
2. Проєктування
2.1 Архітектура
System Architecture Doc (Tech Lead)
API Specification (Swagger) (Tech Lead)
2.2 База Даних
ER-діаграма (PostgreSQL) (Backend Dev)
2.3 Дизайн
Wireframes & UI Kit (Designer)
Інтерактивний прототип (Designer)
3. Розробка (R&D)
3.1 Backend API (Tech Lead)
Auth Service (OAuth2/JWT)
Crypto Core (Sign/Verify)
Document Service (S3)
Notification Service (Email/Webhook)
3.2 Frontend Client (Frontend Dev)
Лендінг та Авторизація
Кабінет користувача
Віджет підпису
Адмін-панель
4. Інфраструктура
4.1 Cloud Setup (AWS) (DevOps)
4.2 Середовища
Dev / Staging Env (DevOps)
Production Env (DevOps)
4.3 CI/CD & Monitoring (DevOps)
5. Тестування
5.1 QA Process
Функціональне тестування (QA)
API Testing (Postman) (QA)
5.2 Security
Pen-testing / Vulnerability Scan (SecOps)
5.3 Performance
Load Testing (JMeter) (QA)
6. Впровадження
6.1 Реліз
Production Deploy (DevOps)
Smoke Test (Prod) (QA)
6.2 Завершення
Користувацькі інструкції (Tech Lead)
Передача проєкту замовнику (PM)

Додаток Г


Діаграма Ганта реалізації проєкту створення SaaS-платформи «UniSign»
1. ІНІЦІАЦІЯ
1.1 Аналіз нормативної бази
1.2 Формування Паспорта та ТЗ
1.3 Підбір команди
2. ПРОЄКТУВАННЯ
2.1 Розробка архітектури
2.2 Проєктування БД
2.3 UX/UI Дизайн (Прототипи)
3. BACKEND (R&D)
3.1.1 Налаштування середовища
3.1.2 Модуль Авторизації
3.1.3 Модуль Документів (S3)
3.1.4 Модуль Криптографії (Core)
3.1.5 Інтеграція з eIDAS/OCSP
3. FRONTEND (R&D)
3.2.1 Верстка інтерфейсу
3.2.2 Інтеграція з API
3.2.3 Віджет підпису (Browser)
4. ІНФРАСТРУКТУРА
4.1 Налаштування Cloud (AWS)
4.2 CI/CD Пайплайни
4.3 Моніторинг та Логування
5. ТЕСТУВАННЯ
5.1 Функціональне тестування
5.2 Тестування безпеки (Pen-test)
5.3 Навантажувальне тестування
6. ВПРОВАДЖЕННЯ
6.1 Реліз (Production Deploy)
6.2 Документація та передача
09-2025
10-2025
11-2025
12-2025
01-2026
02-2026