**Software Requirements Specification for**

**<**  **Програмна система для**

**медичних закладів «LifeLine».**

**Back-end розробка**

**>**

**Version 1.1 approved**

**Prepared by <Анастасія Чернова>**

**<ПЗПІ-21-6>**

**<09.06.2025>**

**ЗМІСТ**

[1Вступ 2](#_pote365t5k6p)

[1.2 Мета 2](#_d5rolk2ce501)

[1.3 Межі 3](#_uw64tnhc6m90)

[1.4 Посилання 3](#_vdghk6alf4gh)

[1.5 Означення та абревіатури 4](#_gl8motvk7l98)

[2 Загальний опис 5](#_o8wavhsmbl38)

[2.1 Перспективи продукту 5](#_knm2qaypbxxj)

[2.2 Функції продукту 5](#_vz0yet20ed49)

[2.3 Характеристики користувачів 6](#_wdd4kr5tf6zg)

[3 Конкретні вимоги 9](#_pmfxvoyufyd3)

[3.1 Вимоги до зовнішніх інтерфейсів 9](#_74hvck7bmgkj)

[3.1.1 Інтерфейс користувача 9](#_4g6pah9x6if5)

[3.1.2 Апаратний інтерфейс 9](#_b3gp1tmhh8mu)

[3.1.3 Програмний інтерфейс 9](#_9ho4d0il8mj4)

[3.1.4 Комунікаційний протокол 10](#_crfwlsl1mh7i)

[3.1.5 Обмеження пам’яті 10](#_dvruwyzkgso)

[3.1.6 Операції 10](#_e5savuiv6xki)

[3.2 Властивості програмного продукту 11](#_luiwjqtotyco)

[3.3 Атрибути програмного продукту 11](#_ukwulx5dytvl)

[3.3.1 Надійність 11](#_o4nmi9siamxi)

[3.3.2 Масштабованість 12](#_hqurnijhrhet)

[3.3.3 Адаптивність 12](#_y0xp3ov2pucz)

[3.3.4 Підтримка платформ 12](#_hug5036burkg)

[3.3.5 Безпека даних 12](#_wjaeljd3igw6)

[3.3.6 Захист від помилок 12](#_3xvzyhyrfc6u)

[3.4 Вимоги бази даних 13](#_mhuhv87zpalx)

# ВСТУП

* 1. Огляд продукту

Програмна система “LifeLine” — це медична інформаційна система, розроблена для автоматизації ключових процесів у закладах охорони здоров’я. Вона забезпечує ефективне управління електронними медичними картками пацієнтів, призначеннями, розкладами лікарів, лабораторними аналізами та взаємодією між персоналом і пацієнтами.

Система складається з двох основних частин: серверної (back-end), яка відповідає за збереження, обробку та безпеку даних та клієнтської (front-end), що забезпечує доступ користувачів до функціоналу через зручний інтерфейс. Взаємодія відбувається через REST API, що дозволяє розширювати продукт, інтегрувати його з іншими цифровими сервісами та адаптувати під різні типи клієнтів.

LifeLine підтримує багаторівневу систему ролей (адміністратор, лікар, пацієнт), має гнучку структуру, що дозволяє масштабувати функціонал, а також включає модулі для створення звітів, перегляду статистики, резервного копіювання та захисту персональних даних згідно з сучасними вимогами.

## Мета

Метою роботи є реалізація сучасної цифрової системи для медичних закладів, яка дозволить зручно та безпечно організувати роботу між пацієнтами, лікарями та адміністрацією. Проєкт спрямований на те, щоб зробити процес отримання медичних послуг зрозумілим, доступним і більш ефективним як для медичного персоналу, так і для людей, які звертаються по допомогу.

Система охоплює всі ключові процеси — від запису на прийом, ведення медичних карток і розкладів лікарів, до зберігання аналізів та результатів у зручному цифровому форматі. Ролі в системі розподілені так, щоб кожен користувач мав лише необхідний доступ, а конфіденційність і безпека персональних даних були повністю забезпечені.

У межах реалізації планується розробити серверну частину для обробки запитів та логіки роботи, а також клієнтську частину з простим інтерфейсом, яка дозволить взаємодіяти з системою з будь-якого пристрою. Такий підхід дозволить не лише покращити якість медичного обслуговування, а й забезпечити адаптивність системи до потреб конкретного закладу та можливість її подальшого розвитку.

## Межі

У межах цієї роботи реалізовано медичну інформаційну систему «LifeLine», що складається з серверної частини, яка відповідає за обробку запитів, бізнес-логіку, зберігання та захист даних, і клієнтської частини — інтерфейсу користувача, через який здійснюється доступ до функціоналу. Система надає можливості для реєстрації, запису до лікаря, призначення процедур, ведення медичних карток, перегляду результатів аналізів, формування PDF-звітів тощо.

Основними технологіями розробки є: Node.js з використанням Express.js (серверна частина), PostgreSQL як система управління базами даних, Sequelize як ORM, React — для побудови клієнтської частини. Для безпеки реалізовано аутентифікацію через JWT та шифрування даних. Взаємодія між частинами здійснюється через REST API.

## Посилання

Робота спирається на сучасні джерела щодо цифровізації охорони здоров’я, офіційну документацію використаних технологій (Node.js, PostgreSQL, Sequelize, Express), а також рекомендації з побудови архітектури та захисту персональних даних. Використано приклади з реальних медичних систем (Helsi, Health24), профільну літературу та технічну документацію, зокрема SRS для системи «LifeLine» і GitHub-репозиторій з кодом.

## Означення та абревіатури

AES-256-GCM – Advanced Encryption Standard with 256-bit key

API – Application Programming Interface

bcrypt – Blowfish-based crypt

CSS – Cascading Style Sheets

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

IT – Information Technology

JSON – JavaScript Object Notation

JWT – JSON Web Token

OAuth2 – Open Authorization 2.0

ORM – Object-Relational Mapping

PDF – Portable Document Format

SaaS – Software as a Service

SQL – Structured Query Language

# ЗАГАЛЬНИЙ ОПИС

## Перспективи продукту

Медична програмна система «LifeLine» має значний потенціал для подальшого розвитку та масштабування. Її модульна архітектура дозволяє з легкістю додавати нові функції, адаптуватися до потреб конкретних медичних установ, а також інтегруватися з державними платформами, такими як eHealth. У майбутньому продукт може розширюватися за рахунок впровадження мобільного додатку для пацієнтів і лікарів, додавання телемедичних консультацій, системи штучного інтелекту для підтримки діагностичних рішень, а також функцій електронного документообігу. Крім того, система може бути адаптована для використання в інших країнах завдяки можливій підтримці мультимовності та стандартів безпеки.

## Функції продукту

До основних функцій системи належать:

* управління електронними медичними картками;
* реєстрація та обробка прийомів ;
* створення електронних рецептів та направлень;
* адміністрування доступу користувачів за ролями (Admin, Doctor, Patient);
* ведення розкладу лікарів, лабораторних та медичних послуг;
* формування звітності та статистики;
* захист та шифрування даних, автентифікація та авторизація;
* запис на прийом до лікаря або послугу онлайн;
* резервне копіювання даних;
* перегляд завантаженості лікарів, контроль відвідувань та ефективності;
* запис на прийом онлайн;
* модуль обробки медичних послуг;
* автоматичне формування медичних довідок та результатів у форматі PDF;
* інтерфейс API для зовнішніх інтеграцій;
* модуль оплати послуг (інтеграція з платіжними сервісами);
* панель адміністратора для керування установами, працівниками, пацієнтами;
* аналітичні панелі для перегляду активності пацієнтів та лікарів;
* модуль обліку та управління лабораторними аналізами (призначення, результати, розклади);
* управління фінансовими звітами;
* створення, редагування та перегляд медичних послуг за лікарями або закладами;
* система оцінювання та перегляду відгуків на лікарів і клініки;
* підтримка персоналізованих розкладів лікарів і послуг;
* відображення історії дій користувачів та зміни статусів записів.

## Характеристики користувачів

Користувачами системи є представники трьох основних ролей: адміністратори, лікарі та пацієнти. Кожна роль має свої функції та рівень доступу до даних.

Адміністратори - технічно підготовлені користувачі, які відповідають за керування установою, облік працівників, контроль доступу, створення звітів, а також загальне адміністрування системи. Очікується середній або високий рівень комп’ютерної грамотності.

Лікарі - медичні працівники, які використовують систему для перегляду та редагування медичних карток, призначення аналізів і ліків, ведення розкладу. Переважно мають базові навички роботи з комп’ютером і потребують простого, інтуїтивно зрозумілого інтерфейсу.

Пацієнти - звичайні користувачі без спеціальної технічної підготовки, які заходять до системи для запису на прийом, перегляду результатів аналізів і отримання повідомлень. Для них критично важлива зручність, доступність з мобільних пристроїв і підтримка української мови.

* 1. Загальні обмеження

Система може мати такі обмеження та винятки:

* працює лише в умовах наявного інтернет-з’єднання (віддалене або локальне), що обмежує можливість автономної роботи;
* не передбачено підтримки застарілих версій браузерів і операційних систем, які не відповідають сучасним вимогам;
* встановлення системи на застаріле серверне або мережеве обладнання може призвести до часткової втрати функціональності або нестабільної роботи через несумісність із сучасними компонентами;
* деякі функціональні модулі (наприклад, глибока аналітика, інтеграція з лабораторіями, автоматичні повідомлення) поки що не входять до поточної версії системи і заплановані на наступні етапи розробки;
* при використанні нових технологічних рішень (наприклад, REST API останніх версій, сучасні стандарти безпеки) можуть виникати складнощі при взаємодії з зовнішніми системами, що використовують застарілі протоколи чи методи обміну даними.
  1. Припущення й залежності

Для коректної роботи системи прийнято такі припущення та визначено залежності:

* передбачається, що всі користувачі (лікарі, пацієнти, адміністратори) володіють базовими навичками користування веб-інтерфейсами та мають доступ до відповідних пристроїв — комп’ютера або смартфона;
* вважається, що адміністрація закладу охорони здоров’я забезпечить надійне мережеве середовище;
* користувачі будуть дотримуватись правил безпеки, зокрема не передавати свої паролі третім особам, використовувати складні паролі та оновлювати їх у разі потреби;
* у процесі експлуатації передбачається, що всі дії в системі (записи, призначення, оновлення інформації) виконуються авторизованими користувачами, згідно з наданими їм правами доступу;
* подальший розвиток системи залежить від технічної підтримки фреймворків, баз даних та бібліотек, які були обрані під час проєктування (Node.js, Express, PostgreSQL, Sequelize тощо).

# КОНКРЕТНІ ВИМОГИ

## Вимоги до зовнішніх інтерфейсів

### Інтерфейс користувача

Інтерфейс повинен бути інтуїтивно зрозумілим, адаптивним для різних пристроїв (ПК, планшети, смартфони) та відповідати сучасним стандартам UX/UI. Основні компоненти: навігаційне меню, форма входу/реєстрації, панель керування пацієнтами, профілі користувачів, розклад прийомів, доступ до електронних карток, форма бронювання.

### Апаратний інтерфейс

Система не має прямої взаємодії з медичним обладнанням. Однак передбачається, що серверна частина буде розміщена на хмарному або локальному сервері з мінімальною конфігурацією:

* процесор: не нижче 4 ядер;
* ОЗП: від 8 ГБ;
* SSD накопичувач;
* підтримка стабільного мережевого з’єднання з пропускною здатністю не менше 100 Мбіт/с.

### Програмний інтерфейс

Система використовує REST API для зв’язку між клієнтською та серверною частинами. API дозволяє:

* реєструвати та автентифікувати користувачів;
* отримувати, створювати, змінювати та видаляти медичні записи;
* керувати розкладом, записами на прийом, призначеннями, аналітикою;
* підключатися до сторонніх сервісів (наприклад, email, платіжних систем).

Усі запити до API мають бути захищеними та проходити перевірку авторизації (JWT).

### Комунікаційний протокол

Для обміну даними між клієнтською частиною та сервером використовується протокол HTTPS. Усі дані передаються у форматі JSON. REST API забезпечує стандартизовану взаємодію між компонентами системи.

### Обмеження пам’яті

Клієнтська частина не потребує значних ресурсів і може працювати на пристроях із 2 ГБ оперативної пам’яті. Серверна частина потребує щонайменше 8 ГБ оперативної пам’яті для стабільної роботи в умовах середнього навантаження. Дані зберігаються у базі PostgreSQL із підтримкою масштабування.

### Операції

У системі LifeLine взаємодія між клієнтами і серверною частиною реалізується через REST API з використанням захищеного протоколу HTTPS. Користувачі надсилають запити до API, а сервер повертає відповіді у форматі JSON, що забезпечує зручну роботу з даними на клієнтській стороні.

Запити до серверної частини здійснюються за допомогою стандартних HTTP-методів:

* GET - отримання даних (наприклад, список пацієнтів, інформація про лікаря, графік прийому, тощо);
* POST - створення нових записів (наприклад, новий прийом, медична картка, рецепт);
* PUT - оновлення наявної інформації (наприклад, зміна профілю користувача або редагування медичного запису);
* PATCH - часткове оновлення окремих полів (наприклад, зміна статусу результату аналізу);
* DELETE - видалення даних (наприклад, скасування прийому або видалення облікового запису).

Операції обробляються відповідними контролерами, які перевіряють права доступу, валідність даних та виконують необхідну бізнес-логіку.

## Властивості програмного продукту

Програмна система LifeLine розроблена як масштабована, безпечна та багатофункціональна платформа, що забезпечує автоматизацію медичних процесів у закладах охорони здоров’я. Основні властивості системи наведено нижче:

* система гарантує стабільну роботу всіх компонентів за умов навантаження багатьох одночасних користувачів, передбачена обробка помилок і виняткових ситуацій;
* передбачено шифрування персональних даних пацієнтів та авторизацію за допомогою JWT. Доступ до функціоналу регулюється ролями (Admin, Doctor, Patient), що виключає несанкціонований доступ;
* архітектура системи дозволяє легко розширювати функціональність, підключати нові модулі чи установи без значного втручання в існуючу логіку;
* клієнтська частина представлена у вигляді вебінтерфейсу, що коректно відображається на різних пристроях (ПК, планшетах, смартфонах);
* передбачена можливість взаємодії з зовнішніми сервісами через API, у тому числі - з лабораторними модулями, платіжними системами або сторонніми сервісами авторизації;
* інтерфейс користувача розроблений з урахуванням принципів зручності, простоти навігації та доступності, що особливо важливо для медичного персоналу.

## Атрибути програмного продукту

### Надійність

Програмна система LifeLine спроєктована з урахуванням вимог до безперервної роботи медичних закладів. Усі ключові модулі системи мають обробку винятків, контроль збоїв при обміні даними та логування критичних подій. Завдяки резервному копіюванню даних забезпечується збереження інформації навіть у разі аварійного завершення роботи системи.

### Масштабованість

Архітектура LifeLine підтримує горизонтальне та вертикальне масштабування. Це дозволяє легко додавати нові лікарні, сервери чи користувачів без необхідності значної модифікації системи. Також реалізована можливість підключення нових функціональних модулів.

### Адаптивність

Інтерфейс системи розроблений таким чином, щоб підлаштовуватися під різні розміри екранів — від стаціонарних ПК до мобільних пристроїв. У майбутньому можлива реалізація персоналізованих налаштувань інтерфейсу для кожного типу користувача: лікаря, пацієнта або адміністратора.

### Підтримка платформ

Система реалізована як вебзастосунок, що не потребує встановлення сторонніх компонентів на пристроях користувача. Для серверної частини використовується кросплатформне середовище Node.js, що дозволяє розгортання як на Windows, так і на Unix-подібних системах.

### Безпека даних

LifeLine відповідає вимогам щодо захисту персональних даних. Доступ до інформації регулюється ролями, усі паролі зберігаються у хешованому вигляді, передача конфіденційних даних відбувається виключно через захищене HTTPS-з'єднання з використанням JWT-токенів.

### Захист від помилок

Усі основні компоненти системи мають механізми обробки помилок. Передбачено перевірку вхідних даних, валідацію форм, повторні спроби при втраті зв’язку з сервером. Непередбачувані помилки логуються та можуть бути оперативно виправлені.

## Вимоги бази даних

Для зберігання даних використовується реляційна база даних PostgreSQL. Вона забезпечує надійне зберігання великої кількості пов’язаних записів.

Кожна сутність системи (пацієнти, лікарі, прийоми, медичні послуги тощо) має окрему таблицю зі зв’язками типу “один до багатьох” або “багато до багатьох”, залежно від бізнес-логіки.

Для оптимізації роботи використовуються індекси, зовнішні ключі, каскадне оновлення та видалення, а також регулярне резервне копіювання бази.

Доступ до бази даних з боку серверної частини реалізовано через ORM Sequelize, що дозволяє легко розширювати або змінювати модель даних без ручного створення SQL-запитів.