1 ВСТУП

1.1 Огляд продукту

Клієнтська частина програмної системи призначена для візуалізації та управління процесами освітлення рослин у тепличних умовах. Вона забезпечує зручний інтерфейс для користувачів різних ролей (адміністратор, технічний персонал, звичайний користувач) та взаємодіє з серверною частиною через REST API. Застосунок дозволяє переглядати показники сенсорів у реальному часі, керувати рослинами, лампами, сенсорами, а також здійснювати операції з обліковими записами користувачів. Особливу увагу приділено безпечній авторизації через сесії з використанням httpOnly cookie.

1.2 Мета

Метою розробки є створення клієнтської частини вебзастосунку для моніторингу та керування освітленням рослин, яка дозволить користувачам зручно працювати з системою через браузер, здійснювати автентифікацію, переглядати дані сенсорів, отримувати попередження, та управляти пов’язаними об’єктами.

1.3 Межі

Функціональні межі клієнтської частини:

– Розроблено за допомогою React і TypeScript;

– Інтерфейс побудований з використанням компонентів Ant Design;

– Для маршрутизації застосовано React Router;

– Авторизація користувачів реалізована через сесії з httpOnly cookie (без токенів);

– Робота з API організована через RTK Query із централізованою обробкою помилок і кешуванням;

– Підтримка розмежування прав доступу на рівні інтерфейсу (protected routes, меню);

– Інтерфейс адаптовано під рольову модель доступу: різні можливості для адміністратора, техніка та звичайного користувача;

– Відображення стану системи, подій, сенсорів, рослин і ламп у режимі, наближеному до реального часу;

– Підтримка імпорту/експорту користувачів через JSON.

1.4 Посилання

У процесі розробки використовувались такі ресурси:

– <https://react.dev>

– <https://redux-toolkit.js.org/rtk-query/overview>

– <https://ant.design/docs/react/introduce>

– <https://reactrouter.com/en/main/start/tutorial>

– <https://www.npmjs.com/package/express-session>

– <https://github.com/tj/connect-redis>

1.5 Означення та абревіатури

Користувач — особа, що працює з вебінтерфейсом. Може мати одну з ролей: адміністратор, технік або звичайний користувач.

API — інтерфейс програмного взаємодії клієнтської частини з серверною.

Сесія — механізм збереження стану користувача між HTTP-запитами, реалізований через Redis і httpOnly cookies.

RTK Query — інструмент із Redux Toolkit для запитів до API та кешування.

Ant Design — компонентна бібліотека для створення сучасного інтерфейсу.

SPA (Single Page Application) — архітектурний підхід, за якого вся логіка інтерфейсу працює на одній сторінці.

JSON — формат файлів, що використовується для імпорту/експорту користувачів.

ProtectedRoute — компонент, який обмежує доступ до певних сторінок залежно від статусу автентифікації та ролі користувача.

2 ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

Основні перспективи продукту:

– Фронтенд-частина програмної системи має потенціал для подальшого розвитку в кількох напрямках:

– реалізація мобільного застосунку для Android/iOS для зручного доступу з будь-якої точки;

– додавання інтеграції з іншими типами пристроїв (зволожувачі, системи поливу тощо);

– вдосконалення системи візуалізації, зокрема аналітики стану рослин у вигляді динамічних графіків і діаграм;

– підтримка повідомлень у реальному часі через WebSocket або push-нотифікації;

– інтеграція з хмарними сервісами для централізованого зберігання і синхронізації стану між кількома пристроями.

2.2 Функції продукту

Інтерфейс користувача забезпечує такі можливості:

– авторизація через сесію (httpOnly cookie);

– перевірка ролей і контроль доступу до сторінок;

– перегляд, створення, редагування та видалення користувачів (доступно лише адміністраторам);

– керування рослинами: створення, перегляд, редагування, видалення;

– робота з сенсорами: додавання, редагування, прив'язка до рослин, перегляд, видалення;

– керування джерелами освітлення: додавання, налаштування параметрів, перегляд, видалення;

– перегляд даних сенсорів у реальному часі;

– візуалізація показників (температура, вологість, освітленість тощо);

– відображення попереджень щодо стану рослин;

– експорт і імпорт користувачів у форматі CSV.

2.3 Характеристики користувачів

2.3.1 Адміністратори

– мають повний доступ до управління системою;

– керують ролями користувачів та їх обліковими записами;

– працюють з інтерфейсом експорту/імпорту CSV;

– мають базову технічну підготовку.

2.3.2 Технічний персонал

– обслуговує сенсори, освітлення та рослини;

– використовує систему для додавання/редагування об’єктів;

– очікує простоти інтерфейсу без перевантаження зайвими функціями.

2.3.3 Кінцеві користувачі (фермери/аграрії)

– переглядають поточний стан рослин та сенсорні показники;

– отримують повідомлення про критичні умови;

– зацікавлені в інтуїтивному, мобільному та візуально зрозумілому інтерфейсі.

2.4 Загальні обмеження

На даному етапі присутні такі обмеження:

– клієнтська частина доступна лише через веб-браузер (мобільного застосунку наразі немає);

– доступ до сторінок залежить від ролі користувача;

– підтримуються лише українська та англійська локалізації;

– frontend працює лише за наявності доступу до API-сервера.

2.5 Припущення й залежності

На даному етапі припущення та залежності такі:

– клієнтський застосунок очікує, що сервер видає сесію через cookie при логіні;

– вся логіка авторизації та перевірки прав здійснюється на бекенді, клієнт лише реагує на статуси;

– передбачається наявність стабільного API з передбачуваними відповідями (JSON);

– RTK Query використовується для комунікації з API, тому API має підтримувати REST-ендпоінти з відповідними методами;

– дані сенсорів оновлюються через бекенд у реальному часі або з невеликою затримкою.

3 КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

Клієнтська частина розроблена як односторінковий веб застосунок (SPA) на базі React. Інтерфейс забезпечує зручну навігацію для різних типів користувачів. Основні елементи UI:

– сторінка входу з авторизацією через сесію;

– сторінки для керування рослинами, сенсорами, освітленням, зчитуваннями;

– адміністративна панель для керування користувачами та ролями;

– реалізовано підтримку української та англійської мов;

– доступ до REST API здійснюється через RTK Query.

3.1.2 Апаратний інтерфейс

Вимоги до апаратного інтерфейсу:

– сучасний браузер з підтримкою JavaScript ES6;

– мінімум 2 ГБ оперативної пам’яті;  
– підключення до мережі інтернет.

3.1.3 Програмний інтерфейс

Серверна частина реалізована на основі фреймворку NestJS із застосуванням TypeORM і PostgreSQL. Основні характеристики:

– реалізовано REST API для взаємодії з клієнтською частиною та IoT-пристроями;

– впроваджено сесійну авторизацію зберіганням сесій у Redis (через cookie, без використання токенів);

– для передачі сенсорних зчитувань у реальному часі використовується протокол MQTT.

Клієнтська частина створена з використанням React та TypeScript:

– для обміну даними з бекендом використовується RTK Query (Redux Toolkit Query);

– інтерфейс побудований з використанням Ant Design, що забезпечує сучасний адаптивний вигляд;

– реалізовано підтримку рольової моделі, багатомовності (українська/англійська);

– інтерфейс дозволяє користувачам керувати рослинами, сенсорами, освітлювальними пристроями та переглядати зчитування в реальному часі;

– сесія зберігається у браузері як HttpOnly cookie, що забезпечує додатковий захист.

Таким чином, клієнт та сервер взаємодіють через REST API та захищену сесію, забезпечуючи зручний та безпечний веб-інтерфейс для аграрних користувачів.

3.1.4 Комунікаційний протокол

У якості комунікаційних протоколів було використано:

– HTTPS для захищеного обміну даними між фронтендом і сервером;

– MQTT для комунікації з IoT-пристроями;

– Сесії реалізовані через express-session з RedisStore, cookie-підхід замість JWT.

3.1.5 Обмеження пам’яті

Клієнтська частина є вебзастосунком, який не потребує встановлення на пристрої користувача — достатньо лише сучасного браузера. Це знижує вимоги до апаратного забезпечення користувача і дозволяє працю

3.1.6 Операції

Основні функціональні операції системи:

– вхід/вихід користувача;

– керування користувачами та ролями;

– створення/редагування рослин, сенсорів та освітлення;

– зчитування показників з сенсорів;

– перегляд попереджень та журналу зчитувань;

– імпорт/експорт даних у форматі JSON.

3.1.7 Функції продукту

Програмне забезпечення має такі функції:

– авторизація та керування сесією;

– інтерфейс для керування даними;

– обробка сенсорних зчитувань у реальному часі;

– автоматичне регулювання освітлення;

– рольова модель доступу.

3.1.8 Припущення й залежності

Припущення:

– користувачі мають базові навички користування ПК;

– пристрої підключені до локальної мережі;

– дані з сенсорів є коректними;

Залежності:

– доступність серверної частини;

– стабільна робота MQTT брокера;

– коректність налаштувань Redis та бази даних;

– справність IoT-пристроїв.

3.2 Властивості програмного продукту

Розроблений програмний продукт дозволяє здійснювати моніторинг і автоматизоване керування освітленням для вирощування рослин у контрольованому середовищі на основі показників сенсорів. Система включає зручний веб-інтерфейс, що адаптований для різних типів користувачів (адміністратор, технік, звичайний користувач, підтримка).

Основні властивості продукту:

– клієнтська частина реалізована як SPA (Single Page Application) на основі React з використанням TypeScript і RTK Query для інтеграції з API;

– зчитування параметрів середовища (вологість, температура, освітлення) відображаються на фронтенді в реальному часі завдяки інтеграції з MQTT;

– реалізовано CRUD-операції для управління рослинами, сенсорами та освітлювальними пристроями через зручні інтерфейси;

– підтримується автоматичне керування освітленням відповідно до даних із сенсорів і встановлених параметрів рослин;

– система попереджень інформує користувача про критичні відхилення у стані середовища (наприклад, нестача світла чи перевищення вологості);

– реалізовано рольову модель доступу з гнучким керуванням правами користувачів через адміністративну панель;

– є можливість експорту та імпорту облікових записів користувачів у форматі CSV;

– автентифікація реалізована на основі сесій з використанням Redis, що забезпечує безпечний і масштабований контроль доступу;

– клієнтська частина має адаптивний дизайн і підтримку української та англійської мов для зручності користувачів.

3.3.1 Надійність

Ціль: система повинна стабільно працювати в режимі реального часу, обробляючи сенсорні дані та виконуючи відповідні дії без збоїв.

Метрика: час простою системи не повинен перевищувати 0.1% на місяць при типовому навантаженні. Передбачено обробку критичних ситуацій на фронтенді для інформування користувача про несправності.

3.3.2 Доступність

Ціль: забезпечити постійний доступ до системи через веб-інтерфейс та гарантувати відображення актуальних даних із сенсорів.

Метрики:

– доступність клієнтської та серверної частини — не менше 99.9% на місяць;

– доступ з будь-якого пристрою через браузер у локальній або корпоративній мережі;

– мінімізація часу завантаження інтерфейсу для зручної роботи користувачів.

3.3.3 Безпека

Ціль: гарантувати захист конфіденційних даних користувачів і цілісність операцій керування освітленням.

Метрики:

– використання хешування паролів на сервері;

– автентифікація та авторизація реалізовані через сесії (зберігаються у Redis);

– cookie з прапорцем HttpOnly;

– передача даних через HTTPS;

– обробка помилок доступу на фронтенді та повідомлення користувача.

3.3.4 Супроводжуваність

Ціль: система повинна легко підтримуватись і розширюватись без необхідності повної переробки.

Метрики:

– розділення логіки на фронтенді за допомогою модулів і Redux-слайсів;

– дотримання єдиних принципів оформлення коду (ESLint, Prettier);

– документація API через Swagger;

– покриття основних частин коду unit та e2e тестами.

3.3.5 Переносимість

Ціль: забезпечити можливість розгортання клієнтської та серверної частини на різних платформах (локальні ПК, сервери, хмари).

Метрики:

– використання кросплатформених технологій: React + NestJS + PostgreSQL;

– зручна збірка фронтенду через Vite/webpack;

– підтримка Docker для ізольованого розгортання.

3.3.6 Продуктивність

Ціль: швидка реакція системи на зміни в середовищі, забезпечення плавного інтерфейсу користувача.

Метрики:

– реакція на нові зчитування із затримкою не більше 1 секунди;

– сервер обробляє щонайменше 1000 запитів на хвилину;

– фронтенд показує дані без помітних затримок навіть на пристроях із середніми характеристиками.

3.4 Вимоги бази даних

У системі використовуються дві бази даних: основна реляційна база PostgreSQL та кешуюче сховище Redis.

PostgreSQL призначена для постійного зберігання структурованих даних. У ній зберігається така інформація:

– облікові записи користувачів та їхні ролі

– інформація про рослини, включаючи стадії росту та вимоги до освітлення

– дані сенсорів: тип, стан, прив’язка до рослин

– параметри освітлювальних пристроїв: яскравість, спектр, стан

– зчитування з сенсорів: температура, вологість, освітленість

– попередження щодо відхилення параметрів від встановлених норм

Redis використовується для тимчасового зберігання сесій користувачів. Це забезпечує:

– швидкий доступ до інформації про автентифікованих користувачів

– зменшення навантаження на основну базу даних

– підтримку TTL (часу життя сесії) та автоматичне очищення неактивних сесій

– надійну роботу механізму сесійної авторизації з використанням httpOnly cookie

3.5 Інші вимоги

Інші вимоги до системи:

– підтримка багатомовного інтерфейсу (українська та англійська мови);

– зберігання користувацьких сесій у Redis для підвищення швидкодії;

– фронтенд адаптований для роботи на пристроях із різною роздільною здатністю;

– інтеграція з MQTT для обміну даними з фізичними пристроями в реальному часі;

– забезпечення доступу до системи тільки авторизованим користувачам відповідно до їхніх ролей.