Специфікація програмного забезпечення

Програмна система для управління мережею фастфуд-закладів

Software Requirements Specification

1.0

21.06.2025

Московченко Євгеній Олексійович

**ІСТОРІЯ ЗМІН**

|  |  |  |  |
| --- | --- | --- | --- |
| **Дата** | **Опис** | **Автор** | **Коментарі** |
| 21.06.2025 | Створено пункти 1.1 - 1.5 | Московченко Євгеній Олексійович |  |
| 21.06.2025 | Створено пункти 2.1 - 2.5 | Московченко Євгеній Олексійович |  |
| 21.06.2025 | Створено пункти 3.1 - 3.4 | Московченко Євгеній Олексійович |  |
| 21.06.2025 | Створено пункт 3.5 | Московченко Євгеній Олексійович |  |

**ЗАТВЕРДЖЕННЯ ДОКУМЕНТУ**

Наступну специфікацію вимог до програмного забезпечення було прийнято та схвалено:

|  |  |  |  |
| --- | --- | --- | --- |
| **Підпис** | **Друковане ім’я** | **Назва** | **Дата** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**ЗМІСТ**

ІСТОРІЯ ЗМІН

ЗАТВЕРДЖЕННЯ ДОКУМЕНТУ

1. ВСТУП

1.1 Огляд продукту

1.2 Мета

1.3 Межі

1.4 Посилання

1.5 Означення та абревіатури

2. ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

2.2 Функції продукту

2.3 Характеристики користувачів

2.4 Загальні обмеження

2.5 Припущення й залежності

3. КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувача

3.1.2 Апаратний інтерфейс

3.1.3 Програмний інтерфейс

3.1.4 Комунікаційний протокол

3.1.5 Обмеження пам’яті

3.1.6 Операції

3.1.7 Функції продукту

3.1.8 Припущення й залежності

3.2 Властивості програмного продукту

3.3 Атрибути програмного продукту

3.3.1 Надійність

3.3.2 Доступність

3.3.3 Безпека

3.3.4 Супроводжуваність

3.3.5 Переносимість

3.3.6 Продуктивність

3.4 Вимоги бази даних

3.5 Інші вимоги

1. ВСТУП

1.1 Огляд продукту

Програмна система "StaffFlow" – це інтегрована платформа, призначена для автоматизації управлінських процесів у мережі закладів швидкого харчування. Система надає інструменти для комплексного управління персоналом, включаючи створення та адміністрування робочих графіків, а також обробку запитів співробітників. Особливістю є інтеграція з пристроями Інтернету речей для автоматичного моніторингу та підтримки оптимальних кліматичних умов: температури та вологості, у зонах зберігання продуктів.

Система складається з центральної серверної частини, що реалізує всю бізнес-логіку, реляційної бази даних PostgreSQL для зберігання даних, клієнтського веб-інтерфейсу для користувачів та менеджерів.

1.2 Мета

Основною метою розробки системи є створення надійного, масштабованого та зручного інструменту, який дозволить працівникам та менеджерам фастфуд-закладів підвищити операційну ефективність та якість сервісу.

1.3 Межі

Продукт охоплює:

* Розгортання Backend (ASP.NET Core) та бази даних (PostgreSQL) у контейнерах Docker;
* Реалізацію RESTful API для взаємодії між усіма компонентами системи;
* Розробку SPA веб-клієнта з функціоналом адміністрування та моніторингу;
* Програмну імітацію IoT-пристрою для збору та надсилання даних телеметрії;
* Реалізацію системи автентифікації та авторизації на основі ролей.

За межами поточного проекту залишається:

* Розробка та випуск нативного мобільного застосунку;
* Пряма інтеграція зі сторонніми системами;
* Розробка та виробництво фізичних IoT-пристроїв;
* Механізми оплати.
  1. Посилання
* ISO/IEC/IEEE 29148:2018: Systems and software engineering — Life cycle processes — Requirements engineering – (<https://www.iso.org/standard/35733.html>);
* ASP.NET Core Documentation – (<https://docs.microsoft.com/aspnet/core/>);
* Vue.js 3 Documentation – (<https://vuejs.org/>);
* PostgreSQL 15 Documentation – (<https://www.postgresql.org/docs/15/>);
* Docker Documentation ­– (<https://docs.docker.com/>);
  1. Означення та абревіатури

Означення:

* Backend: Серверна частина системи;
* Frontend: Клієнтська частина системи.

Абревіатури:

* API (Application Programming Interface): Програмний інтерфейс додатку;
* CORS (Cross-Origin Resource Sharing): Механізм безпечної взаємодії між веб-додатками;
* CRUD (Create, Read, Update, Delete): Базові операції з даними;
* DB (Database): База даних;
* DTO (Data Transfer Object): Об'єкт для передачі даних;
* HR (Human Resources): Сфера діяльності, яка охоплює управління людськими ресурсами;
* IoT (Internet of Things): Інтернет речей;
* REST (Representational State Transfer): Архітектурний стиль для API;
* SPA (Single Page Application): Односторінковий веб-додаток;
* SRS (Software Requirements Specification): Специфікація програмних вимог;
* UI/UX (User Interface/User Experience): Користувацький інтерфейс та досвід взаємодії;

2. ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

Система розробляється як масштабована платформа. У майбутньому її функціонал може бути розширений для підтримки інших галузей, що потребують управління персоналом, таких як роздрібна торгівля або складська логістика. Планується поглиблення аналітичних можливостей шляхом впровадження модулів машинного навчання для прогнозування завантаженості та оптимізації ресурсів. Також передбачається інтеграція з зовнішніми HR- та бухгалтерськими системами для повної автоматизації бізнес-процесів.

* 1. Функції продукту
* створення, редагування та видалення облікових записів користувачів (співробітників, менеджерів, адміністраторів) з розподілом прав доступу на основі ролей;
* створення робочих змін для співробітників, а також обробка запитів на відпустки, лікарняні та інші зміни у графіку;
* прийом та зберігання даних про температуру та вологість, що надходять від сенсорів у реальному часі;
* відображення поточних та історичних даних з сенсорів у вигляді інтерактивних карток та графіків на веб-панелі;
* автоматична генерація звітів про відпрацьовані години персоналу та надсилання сповіщень у разі виходу показників сенсорів за межі норми;
* можливість створення та відновлення резервних копій бази даних для забезпечення надійності системи.
  1. Характеристики користувачів

Система розрахована на два основні категорії користувачів.

Менеджер/Адміністратор має повний доступ до всіх функцій системи, включаючи управління користувачами, налаштування ролей, перегляд всіх запитів. Може подивитися графік показників IoT.

Співробітник має доступ до перегляду власного робочого графіка та подання запитів на його зміну. Може подивитися графік показників IoT.

* 1. Загальні обмеження
* Система не контролює апаратне забезпечення безпосередньо; вона покладається на коректну роботу та підключення до мережі сторонніх IoT-пристроїв.
* Поточна версія не передбачає прямої інтеграції з зовнішніми POS, HR або бухгалтерськими системами. Обмін даними з такими системами має відбуватися вручну.
* Для коректної роботи IoT-пристроїв та доступу до веб-клієнта потрібне стабільне інтернет-з'єднання. Система не гарантує роботу в офлайн-режимі, окрім кешування даних на стороні клієнта.
* Система є інструментом для управління та моніторингу. Вона не надає медичних діагнозів, юридичних або фінансових порад.
* Відсутній мобільний застосунок.
  1. Припущення й залежності

Припущення:

* припускається, що користувачі мають базові навички роботи з веб-інтерфейсами;
* припускається, що у закладах є стабільна Wi-Fi мережа (2.4 GHz) для підключення IoT-пристроїв;
* припускається, що адміністратори системи будуть дотримуватися політик безпеки (наприклад, не передавати облікові дані).

Залежності:

* робота системи залежить від доступності та коректного функціонування інфраструктури Docker на хост-сервері;
* система залежить від сторонніх сервісів для надсилання push-сповіщень на мобільні пристрої (наприклад, Firebase Cloud Messaging);
* Frontend-частина залежить від підтримки сучасних веб-стандартів браузерами користувачів.

3. КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

У цьому розділі визначається, як система буде взаємодіяти з користувачами, апаратним забезпеченням та іншими програмними системами. Описуються ключові інтерфейси, які забезпечують функціонування системи.

3.1.1 Інтерфейс користувача

Користувацький інтерфейс системи реалізовано у вигляді SPA, розробленого на фреймворку Vue.js з використанням TypeScript. Дизайн інтерфейсу є адаптивним. Основні елементи включають навігаційну панель для швидкого доступу до ключових розділів, дашборд для моніторингу IoT-даних у реальному часі та адміністративну панель для управління системою.

3.1.2 Апаратний інтерфейс

Система взаємодіє з апаратним забезпеченням через IoT-пристрої, що базуються на мікроконтролерах ESP32. Ці пристрої відповідають за збір телеметрії з навколишнього середовища, показників температури та вологості за допомогою датчиків DHT11/22. Для передачі даних на сервер IoT-модуль використовує вбудований Wi-Fi модуль.

3.1.3 Програмний інтерфейс

Взаємодія між усіма компонентами системи здійснюється через RESTful API. Backend, реалізований на ASP.NET Core, надає набір HTTP-ендпоінтів для виконання всіх необхідних операцій: автентифікації користувачів, управління даними, отримання звітів та прийому телеметрії. Дані передаються у форматі JSON для гнучкості та сумісності з різними клієнтами.

3.1.4 Комунікаційний протокол

Вся комунікація між клієнтськими застосунками та сервером відбувається виключно через безпечний протокол HTTPS з підтримкою TLS 1.2 або вище. IoT-пристрої надсилають зібрані дані на сервер за допомогою HTTP POST-запитів. Для забезпечення безпечної взаємодії між веб-клієнтом, що працює на одному домені та API, що може бути на іншому, на сервері налаштована підтримка Cross-Origin Resource Sharing.

3.1.5 Обмеження пам’яті

Веб-клієнт оптимізовано для ефективного використання ресурсів і не повинен споживати більше 200 МБ оперативної пам'яті на клієнтській машині при стандартному навантаженні. На стороні IoT-пристрою кеш для тимчасового зберігання телеметрії, на випадок втрати зв'язку, обмежений 64 КБ.

* + 1. Операції

Користувачі системи можуть виконувати різні операції залежно від своєї ролі. Співробітники можуть створювати запити на зміну графіка та переглядати свій розклад, передивлятися графік IoT пристроїв. Адміністратори мають повний доступ до CRUD-операцій для всіх головних сутностей системи, включаючи користувачів, робочі дні та типи днів. Вони також можуть схвалювати або відхиляти запити співробітників.

3.1.7 Функції продукту

Система забезпечує повний цикл управління персоналом та моніторингу. Основні функції включають реєстрацію та автентифікацію користувачів, прийом потокових метрик з IoT-пристроїв, візуалізацію даних у вигляді графіків та карток, а також генерацію звітів про діяльність персоналу. Крім того, реалізовано функціонал адміністрування системи, що дозволяє керувати користувачами.

3.1.8 Припущення й залежності

Передбачається, що для коректної роботи всіх компонентів системи забезпечено стабільне інтернет-з'єднання. Система залежить від інфраструктури Docker для розгортання. Він забезпечує її портативність та незалежність від операційної системи хоста. Функціонування IoT-пристроїв залежить від доступності Wi-Fi мережі у закладах та безвідмовність датчиків.

3.2 Властивості програмного продукту

Програмний продукт реалізує логіку для вирішення двох основних завдань: управління персоналом та моніторингу умов зберігання. Підтримка декількох ролей дозволяє розмежовувати доступ до функцій та даних. Система здатна працювати як у конфігурації для одного закладу, так і в мережевій структурі з кількома локаціями та віддаленим адмініструванням. Завдяки мікросервісному підходу, система підтримує легке додавання нових модулів у майбутньому.

3.3 Атрибути програмного продукту

3.3.1 Надійність

Система спроектована з урахуванням високих вимог до надійності. Розгортання Docker Compose включає політику restart: always, що забезпечує автоматичний перезапуск контейнерів у разі збою. Реалізовано функціонал резервного копіювання бази даних через API, що дозволяє регулярно створювати знімки даних для їхнього відновлення у разі збоїв або пошкодження.

3.3.2 Доступність

Доступ до веб-клієнта можливий з будь-якого браузера. Контейнеризація дозволяє розгортати та масштабувати Backend для обробки зростаючого навантаження, наприклад, при збільшенні кількості підключених IoT-пристроїв або активних користувачів.

3.3.3 Безпека

Реалізовано валідацію вхідних даних на сервері, обмеження прав доступу на основі ролей за допомогою атрибутів Authorize, та налаштовано політику CORS для безпечної взаємодії між клієнтом та сервером. Вся комунікація шифрується через HTTPS. Паролі користувачів зберігаються не у відкритому вигляді, а у вигляді безпечних хешів, створених за допомогою алгоритму bcrypt.

3.3.4 Супроводжуваність

Код системи структурований відповідно до принципів багатошарової архітектури: API, сервіси, репозиторії. Таким чином покращується розуміння коду, його модифікацію та додавання нового функціоналу. Використання Docker Compose полегшує процес розгортання та підтримки системи в різних середовищах.

3.3.5 Переносимість

Весь програмний стек (ASP.NET Core, PostgreSQL) запаковано в Docker-контейнери. Це дозволяє перенести систему на будь-який сервер з підтримкою Docker, будь то локальна машина, приватний сервер або хмарна інфраструктура.

3.3.6 Продуктивність

* Час відгуку API: Усі запити до API, що стосуються отримання або оновлення даних (наприклад, графіки, дані користувачів), повинні оброблятися на сервері з часом відгуку не більше 200 мс при середньому навантаженні.
* Обробка IoT-даних: Система спроектована для прийому даних від десятків IoT-пристроїв з інтервалом до 10 секунд. Затримка від моменту відправки даних пристроєм до їх збереження в базі даних не повинна перевищувати 500 мс.
* Продуктивність веб-інтерфейсу: Клієнтський SPA-додаток має завантажуватися та ставати інтерактивним протягом 2-3 секунд на стабільному інтернет-з'єднанні. Оновлення даних на дашборді моніторингу має відбуватися без помітних затримок.
* Масштабованість: Архітектура на базі контейнерів Docker дозволяє горизонтальне масштабування Backend-сервісу шляхом запуску додаткових екземплярів контейнерів для балансування навантаження при зростанні кількості клієнтів.

3.4 Вимоги бази даних

* база даних зберігає інформацію про користувачів, типи робочих днів, робочі дні, зв'язки між ними, запити на зміни та дані з сенсорів;
* для забезпечення цілісності використовуються обмеження зовнішніх ключів, що гарантує коректні зв'язки між таблицями;
* схема бази даних керується за допомогою механізму міграцій Entity Framework Core. Кожна зміна в моделі даних супроводжується створенням нової міграції, що дозволяє версіонувати та послідовно оновлювати структуру бази даних;
* для прискорення запитів ключові поля, що використовуються для пошуку та сортування проіндексовані.

3.5 Інші вимоги

Система розроблена з урахуванням необхідності захисту персональних даних співробітників. Доступ до чутливої інформації обмежений на основі ролей. Паролі користувачів зберігаються виключно у вигляді хешів.

Контейнерний підхід до розгортання та використання легкої версії образу ASP.NET Core Runtime дозволяють оптимізувати споживання ресурсів CPU/RAM на сервері.

Система є інструментом для управління та моніторингу. Вона не формує юридично значущих документів для розрахунку заробітної плати та не надає рекомендацій на основі даних з IoT-пристроїв.