ДОДАТОК В

Специфікація програмного забезпечення

1 ВСТУП

1.1 Огляд

Даний документ є специфікацією вимог до програмної системи для оптимального управління робочим персоналом підприємства із застосуванням IoT-пристроїв, веб- і мобільного інтерфейсу. Специфікація містить загальний опис системи, функціональні можливості, характеристики користувачів, зовнішні інтерфейси, обмеження та конкретні вимоги.

1.2 Мета

Метою створення даної специфікації є формалізація функціональності, структури та вимог до системи, що забезпечує автоматизацію обліку працівників, розподілу активностей та контролю фізичного стану персоналу з метою ефективної експлуатації в умовах підприємства.

1.3 Межі

Система охоплює серверну частину (Node.js + Express), веб-застосунок (React), мобільний застосунок (Kotlin/Android), базу даних (PostgreSQL), а також IoT-пристрій на базі ESP32. Вона реалізує REST API, використовує JWT для авторизації, підтримує багаторівневу ролеву модель доступу та алгоритм розподілу активностей на основі показників працівників.

1.4 Посилання

Даний документ створено відповідно до стандартів IEEE Std 830-1998 (IEEE Recommended Practice for Software Requirements Specifications) та ДСТУ ISO/IEC 15910:2002 «Процес створення документації користувача для програмного застосунку». Структура документа відповідає рекомендаціям IEEE для розробки специфікацій вимог до програмного забезпечення.

2 ЗАГАЛЬНИЙ ОПИС

2.1 Перспективи продукту

У сучасних умовах підвищених вимог до ефективності виробничих процесів, здоров'я працівників та цифровізації підприємств особливо актуальною є проблема оптимізації управління персоналом. Часто компанії зіштовхуються з труднощами планування графіку роботи, раціонального розподілу завдань між працівниками та контролю за станом їхнього здоров’я. Відсутність комплексного інструменту для автоматизованого управління цими процесами призводить до неефективного використання людських ресурсів, ризику перевантаження персоналу та низької продуктивності.

Розроблена система є інноваційним цифровим рішенням для підприємств, яке поєднує у собі веб-інтерфейс, мобільний застосунок і IoT-пристрої для збору фізіологічних даних працівників. Система дозволяє автоматизувати підбір оптимальних завдань на основі даних про стан здоров’я співробітників, а також ефективно управляти компанією та працівниками. Завдяки використанню сучасних технологій, таких як Node.js, PostgreSQL, React, Kotlin та ESP32, забезпечується надійна, масштабована та багатокомпонентна архітектура.

У перспективі система може бути масштабована для підприємств різних галузей, а також розширена за рахунок впровадження елементів штучного інтелекту для прогнозування навантажень.

2.2 Функції продукту

Система підтримує наступні функції:

FE-1: можливість зареєструватися та увійти до облікового запису(JWT);

FE-2: створення та редагування профілю користувача;

FE-3: керування компанією (створення, видалення, редагування інформації);

FE-4: управління персоналом (створення, редагування, розподіл активностей);

FE-5: формування оптимального розкладу за допомогою алгоритму оптимізації;

FE-6: моніторинг даних з IoT-пристрою (пульс, температура, тривалість роботи);

FE-7: підписка через PayPal;

FE-8: декілька ролей користувачів у системі: незареєстрований користувач, зареєстрований користувач, працівник, адміністратор компанії, власник підприємства, системний адміністратор.

2.3 Характеристики користувачів

Користувачі мають базові навички користування веб або мобільними застосунками. Для адміністративного доступу потрібен підтверджений акаунт. Різні ролі мають різний рівень доступу та інтерфейс.

2.4 Загальні обмеження

Інтерфейс доступний лише двома мовами: українською та англійською. Необхідний постійний інтернет-зв'язок. Залежність від сервісу PayPal. Обмежена точність IoT-даних за несправності пристрою. Необхідність дотримання норм захисту персональних даних.

2.5 Припущення і залежності

Передбачається, що має доступ до Android-пристрою та браузера. Серверна частина працює на хмарному хості. IoT-пристрої мають доступ до Wi-Fi. Система підтримує HTTPS та зберігає паролі у хешованому вигляді.

3 КОНКРЕТНІ ВИМОГИ

3.1 Вимоги до зовнішніх інтерфейсів

3.1.1 Інтерфейс користувачів

Інтерфейс системи має бути зручним, інтуїтивно зрозумілим і відповідати принципам UI/UX-дизайну. Користувач повинен легко орієнтуватися в навігації. Веб-інтерфейс адаптивний, побудований з використанням React. Мобільний інтерфейс реалізовано на Kotlin (Android). Інтерфейс підтримує англійську та українську мови (i18n).

3.1.2 Програмний інтерфейс

Серверна частина системи реалізована з використанням платформи Node.js та фреймворку Express.js, що забезпечує високу швидкість обробки запитів та гнучку маршрутизацію. Для обміну даними між клієнтською та серверною частинами використовується REST API, де всі запити передаються через протоколи HTTP/HTTPS у форматі JSON.

У системі реалізовано механізм авторизації та автентифікації на основі JWT (JSON Web Token). Кожен користувач після входу в систему отримує токен, який прикріплюється до заголовків наступних запитів для підтвердження прав доступу до ресурсів.

Мобільний застосунок, створений на платформі Android з використанням мови Kotlin та бібліотеки Retrofit, здійснює запити до API, використовуючи ті самі маршрути, що і веб-застосунок. Для збереження токена на мобільному пристрої використовується механізм SharedPreferences.

IoT-пристрій на базі мікроконтролера ESP32 надсилає дані про фізіологічні показники працівника на сервер через HTTP-запити з використанням REST API. Дані надходять у вигляді JSON-об’єктів і обробляються відповідними ендпоінтами, що відповідають за обробку показників здоров’я. Усі API забезпечують обробку помилок із поверненням відповідних кодів стану HTTP (200, 400, 401, 404, 500) та повідомлень про помилки у стандартизованому форматі.

3.2 Властивості програмного продукту

Програмна система повинна забезпечувати реалізацію наступного функціоналу:

* реєстрація та авторизація користувачів із підтримкою ролей;
* редагування профілю користувача та даних підприємства;
* створення нового підприємства із можливістю керування його працівниками;
* додавання, редагування та видалення працівників підприємства;
* автоматичне формування оптимальних активностей для працівників на основі даних фізичного стану та освіти (за допомогою алгоритму математичної згортки);
* інтеграція з IoT-пристроями (на базі ESP32) для збору фізіологічних показників працівників;
* оформлення та керування підпискою через сервіс PayPal;
* перегляд і редагування списку активностей, призначених працівникам;
* перегляд інформації про кожного працівника (загальний стан, тощо);
* керування правами доступу до функцій системи залежно від ролі користувача;
* синхронізація даних між сервером, веб-інтерфейсом, мобільним застосунком і пристроями збору даних.

3.3 Атрибути програмного продукту

3.3.1 Надійність

Система повинна стабільно працювати при стандартному навантаженні. Обробка винятків має здійснюватися централізовано, з повідомленням користувача про причину помилки.

3.3.2 Доступність

Система повинна бути постійно доступною з будь-якої точки світу. Мобільний клієнт функціонує автономно з синхронізацією.

3.3.3 Безпека

Паролі мають зберігатися в хешованому вигляді. Повинно бути обмеження доступу відповідно до ролі. Верифікація токенів JWT.

3.3.4 Супроводжуваність

Архітектура системи побудована з урахуванням принципів чистої архітектури та розділення відповідальностей між компонентами, що значно спрощує подальший супровід і розширення. Основні частини (доменна логіка, інтерфейс, API, база даних) розділені у відповідні шари, що дозволяє модифікувати або тестувати один компонент без впливу на інші.

Веб-застосунок реалізований на React з використанням компонентного підходу. Це дозволяє легко повторно використовувати окремі UI-елементи.

Мобільний застосунок дотримується архітектурного шаблону MVC, що дозволяє чітко розмежувати логіку, представлення та обробку даних. Це також спрощує налагодження та рефакторинг коду.

У системі впроваджено журналювання подій та помилок, що дозволяє аналізувати роботу компонентів у реальному часі. Завдяки використанню TypeScript на серверній стороні, знижено ризики помилок на етапі компіляції.

Система легко розширюється за рахунок модульності API та можливості додавання нових моделей у базі даних без зміни загальної архітектури.