Специфікація програмної системи

Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук

Кафедра програмної інженерії

СПЕЦИФІКАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Програмна система Smart War Drones

Студент гр. ПЗПІ-21-2 Фіненко Арсеній Віталійович

Харків

2025 р.

ЗМІСТ

[1 Вступ 4](#_Toc201502921)

[1.1 Огляд продукту 4](#_Toc201502922)

[1.2 Мета 5](#_Toc201502923)

[1.3 Межі 5](#_Toc201502924)

[1.4 Посилання 7](#_Toc201502925)

[1.5 Означення та абревіатури 7](#_Toc201502926)

[2 Загальний опис 10](#_Toc201502927)

[2.1 Перспективи продукту 10](#_Toc201502928)

[2.2 Функції продукту 11](#_Toc201502929)

[2.3 Характеристики користувачів 12](#_Toc201502930)

[2.4 Загальні обмеження 13](#_Toc201502931)

[2.5 Припущення і залежності 14](#_Toc201502932)

[3 Конкретні вимоги 15](#_Toc201502933)

[3.1 Вимоги до зовнішніх інтерфейсів 15](#_Toc201502934)

[3.1.1 Інтерфейс користувача 15](#_Toc201502935)

[3.1.2 Інтерфейс апаратного забезпечення 15](#_Toc201502936)

[3.1.3 Програмний інтерфейс 16](#_Toc201502937)

[3.1.4 Комунікаційний протокол 17](#_Toc201502938)

[3.1.5 Обмеження пам’яті 17](#_Toc201502939)

[3.1.6 Операційні вимоги 18](#_Toc201502940)

[3.1.7 Функції продукту 18](#_Toc201502941)

[3.1.8 Припущення та залежності 19](#_Toc201502942)

[3.2 Властивості програмного продукту 20](#_Toc201502943)

[3.3 Атрибути програмного продукту 21](#_Toc201502944)

[3.3.1 Надійність 21](#_Toc201502945)

[3.3.2 Доступність 22](#_Toc201502946)

[3.3.3 Безпека 24](#_Toc201502947)

[3.3.4 Супроводжуваність 26](#_Toc201502948)

[3.3.5 Переносимість 27](#_Toc201502949)

[3.3.6 Продуктивність 29](#_Toc201502950)

[3.4 Вимоги бази даних 30](#_Toc201502951)

[3.5 Інші вимоги 32](#_Toc201502952)

# **1 ВСТУП**

## 1.1 Огляд продукту

Програмне забезпечення Smart War Drones є інтегрованою військовою інформаційною системою, що призначена для дистанційного управління безпілотними літальними апаратами (БПЛА), організації їхньої взаємодії в межах визначених бойових напрямків, планування операцій та моніторингу дій на інтерактивній мапі. Система надає інструменти як для операторів дронів, що безпосередньо керують технікою, так і для офіцерів, які здійснюють стратегічне планування, контроль та координацію.

Smart War Drones складається з трьох основних компонентів:

– клієнтської частини (SPA-застосунок на базі React), яка забезпечує взаємодію користувача з системою через зручний та функціональний інтерфейс;

– серверної частини (на ASP.NET Core 8.0), яка обробляє запити, зберігає дані у базі MongoDB, забезпечує авторизацію, обробку операцій, передачу команд та повідомлень;

– Збір і відображення статистики – інформація про стан здоров’я (наприклад, пульс), маршрут та інші показники подорожі.

– IoT-компонента (реалізованого на ESP32 з Wokwi-моделюванням), який емулює або приймає реальні дані з дронів – наприклад, температуру, вологість або інші параметри – та передає їх на сервер у зашифрованому вигляді.

Продукт розробляється з урахуванням високих вимог до інформаційної безпеки, включаючи шифрування даних на клієнті (AES, RSA), використання JWT-токенів для авторизації, обміну ключами через протокол Diffie–Hellman, та застосування стійкого до атак алгоритму хешування Argon2 для облікових даних.

На відміну від класичних цивільних рішень у сфері планування польотів дронів (таких як DJI Terra, UgCS, Mission Planner), Smart War Drones зосереджено на бойовому застосуванні БПЛА в умовах радіоелектронного протистояння, надаючи можливість створювати покрокові сценарії взаємодії дронів, координувати їхні дії у просторі та часі, а також забезпечувати шифровану комунікацію між членами підрозділу.

Таким чином, система забезпечує високий рівень функціональності, захисту даних, та гнучкість у плануванні бойових дій із залученням БПЛА, а також інтеграцію з апаратними засобами.

## 1.2 Мета

Метою створення програмного забезпечення Smart War Drones є розробка надійної, безпечної та функціонально повної системи військового призначення для дистанційного керування безпілотними літальними апаратами, що дозволяє ефективно планувати та координувати дії дронів у межах визначених бойових напрямків.

Система має забезпечити спільну роботу операторів і офіцерів у реальному часі, з можливістю швидкого обміну даними, прийняття рішень на основі інформації з дронів, та інтерактивного керування сценаріями операцій. Особлива увага приділяється інформаційній безпеці, захисту користувацьких даних та безпечному обміну повідомленнями і командами в умовах бойових дій або нестабільного з’єднання.

Програмне рішення повинно поєднувати зручність користування, гнучкість інтеграції з IoT-пристроями (зокрема дронами з ESP32) та розширюваність у майбутньому – для підтримки нових типів місій, аналітики та групових операцій.

## 1.3 Межі

Програмна система Smart War Drones розроблена для вирішення специфічного класу завдань, пов’язаних із управлінням безпілотними літальними апаратами (БПЛА) в умовах військових операцій або тренувальних місій. При цьому система має визначені функціональні, технічні та контекстні обмеження, що окреслюють її сферу застосування.

Функціональні межі:

– система не передбачає фізичне керування дроном на рівні польотної автоматики, а лише організовує логіку завдань, черги дій, відправлення команд, збирання телеметрії та візуалізацію даних;

– робота дронів, які не мають інтеграції через протокол, реалізований в рамках ESP32/HTTP (Wokwi або реальні пристрої), не підтримується;

– виявлення об’єктів, комп’ютерний зір, автонаведення та подібні функції не входять до складу системи, але можуть бути реалізовані як окремі модулі з подальшою інтеграцією.

Технічні межі:

– система працює в локальному середовищі без використання HTTPS, натомість реалізує симетричне/асиметричне шифрування (AES, RSA) та обмін ключами через Diffie–Hellman, що забезпечує базову конфіденційність і цілісність даних;

– база даних реалізована на основі MongoDB, що накладає обмеження на структуру даних (NoSQL), але забезпечує гнучке зберігання інформації про дрони, користувачів та бойові напрямки;

– візуалізація мап реалізована з використанням Leaflet з офлайн-тайлами або локальними стилями. Використання онлайн-джерел (як-от Google Maps) за замовчуванням не передбачено.

Контекстні межі:

– система призначена для використання у внутрішньовідомчому середовищі (військових частинах, навчальних центрах) із наявністю мережевої інфраструктури (WiFi або локальні API);

– підтримка багатокористувацького режиму передбачена лише в межах авторизованого кола осіб, з ролями оператор або офіцер, із відповідними обмеженнями доступу до функціоналу;

* не передбачається масове масштабування системи до рівня тисяч одночасних користувачів або бойових напрямків. Продукт орієнтований на малі та середні групи – до десятків активних користувачів у сесії.

Таким чином, межі системи чітко окреслюють її функціональну зону відповідальності та дозволяють зосередитися на ключових завданнях, пов’язаних із управлінням груповими діями дронів, не перевантажуючи проєкт надмірною складністю або універсальністю.

## 1.4 Посилання

Під час розробки програмної системи Smart War Drones використовувалися такі основні нормативні документи, специфікації, бібліотеки та ресурси:

– RFC 7519 – JSON Web Token (JWT): специфікація формату токенів, що використовується для авторизації та автентифікації користувачів у системі. Доступно за посиланням: <https://datatracker.ietf.org/doc/html/rfc7519>;

– Wokwi Docs – Підключення ESP32 до WiFi: документація щодо підключення мікроконтролера ESP32 до бездротової мережі WiFi, що використовувалась для IoT-компонента системи. Доступно за посиланням: <https://docs.wokwi.com/guides/esp32-wifi#connecting-to-the-wifi>;

– Argon2 – Password Hashing Algorithm (arXiv): стаття з описом алгоритму Argon2, використаного для хешування облікових даних користувачів. Доступно за посиланням: <https://arxiv.org/abs/2504.17121>;

* GitHub – Архів проєкту: вихідний код та документація до розробленого програмного забезпечення. Доступно за посиланням: <https://github.com/NureFinenkoArsenii/2025_B_PI_PZPI-21-2_Finenko_A_V>.

## 1.5 Означення та абревіатури

У цьому документі застосовуються такі основні терміни, означення та абревіатури:

– БПЛА (Безпілотний літальний апарат) – повітряний апарат, який здатен здійснювати політ без безпосередньої участі пілота на борту, керований дистанційно або автономно;

– РЕБ (Радіоелектронна боротьба) – сукупність заходів, спрямованих на придушення, перехоплення або підміну радіосигналів супротивника;

– ПЗ (Програмне забезпечення) – загальний термін для всіх програм і систем, які забезпечують роботу комп'ютера або пристрою;

– SWD (Smart War Drones) – власна назва розробленої системи для планування операцій та керування безпілотними апаратами;

– GPS (Global Positioning System) – система супутникової навігації, що забезпечує визначення географічного положення об’єкта.

– HTTP (Hypertext Transfer Protocol) – протокол прикладного рівня для передавання гіпертексту (веб-сторінок);

– HTTPS (Hypertext Transfer Protocol Secure) розширення HTTP, що використовує шифрування для забезпечення безпеки передаваних даних;

– RSA (Rivest–Shamir–Adleman) – асиметричний криптографічний алгоритм шифрування;

– AES (Advanced Encryption Standard) – стандарт симетричного блочного шифрування даних;

– DH (Diffie–Hellman) – метод обміну криптографічними ключами між двома сторонами;

– JWT (JSON Web Token) – відкритий стандарт, що визначає компактний та самодостатній спосіб безпечної передачі інформації між сторонами у вигляді JSON-об'єкта;

– JSON (JavaScript Object Notation) – текстовий формат обміну даними, заснований на синтаксисі JavaScript;

– JSX (JavaScript XML) – розширення синтаксису JavaScript, що дозволяє використовувати HTML-подібні структури всередині React-компонентів;

– XML (Extensible Markup Language) – мова розмітки для опису структури документів та обміну даними;

– SPA (Single Page Application) – вебзастосунок, який динамічно підвантажує контент без повного перезавантаження сторінки;

– ID (Identifier) – унікальний ідентифікатор об'єкта або користувача в системі;

– UML (Unified Modeling Language) – стандартизована мова моделювання, що використовується для візуального представлення архітектури програмного забезпечення;

– Оператор – роль користувача оператора дронів, який працює з БПЛА та статистикою з датчиків;

– Офіцер – роль користувача офіцера, який займається плануванням операцій з дронами та модерує створені напрямки;

# **2 ЗАГАЛЬНИЙ ОПИС**

## 2.1 Перспективи продукту

Програмний продукт Smart War Drones (SWD) має значний потенціал для подальшого розвитку як у військовому, так і у цивільному секторі. Його модульна архітектура, сучасна реалізація засобами вебтехнологій і підтримка IoT-компонентів створюють надійну основу для масштабування, інтеграції нових функцій та адаптації до різних сценаріїв використання.

На початковому етапі SWD розробляється як засіб для візуального планування бойових операцій з використанням безпілотних літальних апаратів (БПЛА), що дозволяє створювати маршрути, визначати черги взаємодії дронів, забезпечувати командну координацію, а також спостерігати за поточним станом дронів у реальному часі. Завдяки використанню інтерактивної мапи, користувачі мають змогу точно позиціонувати дрони на полі бою та формувати покрокові сценарії їх дій.

У подальшому система може бути розширена шляхом

– впровадження штучного інтелекту для автоматичної оптимізації маршрутів та сценаріїв дій дронів;

– додавання підтримки реальних дронів із вбудованими сенсорами через протоколи типу MAVLink;

– розробки мобільної версії для Android/iOS для оперативного управління підрозділами з планшетів і смартфонів;

– інтеграції з геоінформаційними системами (GIS) для аналізу рельєфу, зон ризику, погодних умов тощо.

– використання машинного навчання для прогнозування можливих загроз або відстеження ворожої активності на основі патернів руху.

Цивільне застосування також є перспективним напрямком. Зокрема, система може бути адаптована для:

– управління дронами в зонах лиха (пожежі, повені, пошуково-рятувальні операції);

– координації транспортних дронів у логістичних хабах;

– управління агродронами в сільському господарстві.

Крім функціонального розвитку, передбачається також удосконалення користувацького інтерфейсу, розширення системи безпеки (впровадження двофакторної автентифікації, підтримка HTTPS, журналювання доступу), а також оптимізація продуктивності при великій кількості активних користувачів.

Таким чином, Smart War Drones є не просто експериментальною системою, а платформою з широкими можливостями для розвитку, яка здатна задовольнити як тактичні потреби військових підрозділів, так і потреби сучасних цивільних секторів, де автономні дрони використовуються для забезпечення безпеки, логістики або моніторингу.

## 2.2 Функції продукту

Програмне забезпечення Smart War Drones (SWD) реалізує набір ключових функцій, що забезпечують повний цикл управління бойовими та розвідувальними операціями із використанням безпілотних літальних апаратів (БПЛА). Система орієнтована на спільну роботу операторів дронів та офіцерів, які відповідають за планування дій у межах визначених бойових напрямків. Основні функціональні можливості продукту включають:

* реєстрація та автентифікація користувачів: користувачі проходять реєстрацію із валідацією електронної пошти та захистом облікових даних через алгоритм хешування Argon2, після входу автентифікація підтримується за допомогою JWT-токенів із механізмом оновлення;
* формування бойових напрямків та інтерактивна карта: користувачі з роллю офіцер можуть створювати напрямки операцій, редагувати їх, прив'язувати дронів, додавати сценарії дій та переглядати їх на мапі; система використовує Leaflet.js для роботи з картографічними плитками, підтримується попередній перегляд і редагування конфігурації;
* управління дроном та сценарії взаємодії: для кожного дрона задається тип, унікальний ідентифікатор, базові характеристики та набір команд; користувачі можуть створювати покрокові анімаційні сценарії із заданими інтервалами часу, чергами дій та умовами виконання;
* месенджер для комунікації: реалізовано приватне листування між користувачами, а також групові чати в межах бойових напрямків. Повідомлення шифруються на клієнті за допомогою гібридного шифрування (AES + RSA + DH); підтримується обмін файлами та індикація непрочитаних повідомлень;
* захист персональних даних: імплементовано змінний ідентифікатор користувача, який оновлюється кожні 5 хвилин, що ускладнює відстеження у зовнішніх системах; ключі шифрування зберігаються лише локально.
* статистика дронів та інтеграція з IoT: система приймає дані з реальних або симульованих сенсорів на базі ESP32 через Wokwi; дані надходять через WiFi-з’єднання й обробляються сервером;
* ролі користувачів та права доступу: в системі реалізовано розмежування прав: оператори можуть керувати лише власними дронами, офіцери – адмініструвати напрямки, додавати користувачів, змінювати мапу та сценарії;
* панель керування та UI: інтерфейс побудований як SPA (Single Page Application) із мінімальним часом реакції; реалізовані адаптивні модальні вікна, інтуїтивно зрозуміла навігація, підсвічування нових подій.

Таким чином, Smart War Drones охоплює не лише візуалізацію та управління дронами, але й забезпечує безпечну взаємодію між учасниками операцій, логіку шифрування, обробку IoT-даних та зручний UI для динамічного керування місіями.

## 2.3 Характеристики користувачів

Програмне забезпечення Smart War Drones призначене для використання військовим персоналом, який має відповідні повноваження щодо керування безпілотними літальними апаратами (БПЛА) та планування бойових операцій. Залежно від ролі у системі, користувачі поділяються на дві основні категорії:

* оператори дронів: виконують безпосереднє керування дронами, переглядають прив’язані до них бойові напрямки, створюють сценарії дій, відправляють повідомлення у чатах; для ефективної роботи користувачу достатньо мати базові навички роботи з інтерфейсом браузера, а також початкові знання у сфері керування БПЛА;
* офіцери: мають розширений функціонал – створюють та адмініструють бойові напрямки, додають користувачів, призначають дронів, планують мапу операції, координують сценарії дій; очікується, що офіцери мають досвід стратегічного планування, аналітичного мислення та навички роботи в команді.

Інтерфейс системи адаптований для роботи на комп’ютерах і не потребує спеціального технічного оснащення. Вся взаємодія відбувається у захищеному середовищі, а вхід в систему вимагає попередньої реєстрації та автентифікації.

## 2.4 Загальні обмеження

Система Smart War Drones розроблена як десктопна веб-застосунок (SPA) і орієнтована на використання в середовищі сучасних браузерів (Google Chrome, Microsoft Edge, Mozilla Firefox). Через це повноцінна робота на мобільних пристроях або в застарілих браузерах не гарантується.

Іншим важливим обмеженням є вимога до наявності стабільного мережевого підключення, оскільки обмін даними між клієнтом, сервером і IoT-пристроями відбувається в реальному часі. Задля безпеки передача даних реалізована через шифрування, однак у локальному середовищі може бути відсутнє повноцінне HTTPS-з'єднання.

Система також передбачає використання MongoDB як основного сховища даних. У зв'язку з цим вона не підтримує прямої роботи з іншими типами СУБД без додаткової адаптації.

У межах IoT-симуляції реалізовано роботу лише з обмеженим набором віртуальних сенсорів (температура, вологість, рівень заряду, напрямок і швидкість вітру), що може не відображати повного спектра даних у реальних умовах бойового використання БПЛА.

## 2.5 Припущення і залежності

Під час розробки програмного забезпечення Smart War Drones були визначені певні обмеження та зроблені ключові припущення щодо середовища його використання. Зокрема, передбачається, що користувачі системи володіють базовими навичками роботи з комп’ютером та здатні самостійно взаємодіяти з картографічним інтерфейсом, інтерфейсом планування сценаріїв, а також функціоналом обміну повідомленнями.

Очікується, що система буде розгортатися в безпечному внутрішньому середовищі (локальній мережі), або на ізольованих серверах, з обмеженим або контрольованим доступом до зовнішньої мережі Інтернет. Таке припущення дозволяє спростити захист на рівні мережевої інфраструктури без повноцінного впровадження PKI чи SSL-сертифікації.

Серед обмежень – припущення, що всі дрони зареєстровані в системі наперед і мають унікальний ідентифікатор. Реєстрація та симуляція роботи IoT-компонента реалізована з використанням емулятора Wokwi, тому вважається, що в реальному застосуванні обладнання буде відповідати заданим електронним схемам і протоколам зв’язку.

# **3 КОНКРЕТНІ ВИМОГИ**

## 3.1 Вимоги до зовнішніх інтерфейсів

### 3.1.1 Інтерфейс користувача

Інтерфейс користувача системи Smart War Drones реалізований як односторінковий вебдодаток (SPA) з використанням фреймворку React. Основний акцент зроблено на зручності взаємодії, інтуїтивно зрозумілому дизайні та візуальній ієрархії елементів. Усі компоненти інтерфейсу розроблені з урахуванням потреб військових операторів і офіцерів, що передбачає швидкий доступ до ключових функцій: авторизації, перегляду мапи, керування дронами, створення бойових сценаріїв та взаємодії через чат.

Головна сторінка містить інформаційно-привітальний блок із випадковим гаслом, форму авторизації та можливість переходу до реєстрації. Навігаційна панель дозволяє перейти до сторінок про систему, майбутній мобільний застосунок або контактної форми. Робоче середовище оператора представлено у вигляді інтерактивної мапи з модальними вікнами для управління напрямками, дронами та сценаріями.

Усі дії користувача супроводжуються анімаційними ефектами, підказками та попередженнями у випадках помилок. Особливу увагу приділено адаптивності: інтерфейс коректно масштабується під різні роздільні здатності дисплеїв, що дозволяє працювати з системою як на десктопах, так і на планшетах.

### 3.1.2 Інтерфейс апаратного забезпечення

У системі Smart War Drones реалізовано інтерфейс апаратного забезпечення для інтеграції з безпілотними літальними апаратами, симульованими у середовищі Wokwi на базі мікроконтролера ESP32. Комунікація з серверною частиною здійснюється за допомогою протоколу HTTP, що використовується для відправлення сенсорних даних на REST-ендпоінт.

Апаратна частина включає в себе підключені датчики температури та вологості (DHT22), потенціометри для симуляції рівня заряду батареї, швидкості та напрямку вітру, а також кнопку присутності РЕБ (радіоелектронної боротьби). Зчитування показників здійснюється з заданою періодичністю, після чого значення передаються у вигляді JSON-об’єкта на сервер для обробки та збереження.

Інтерфейс підтримує підключення будь-якої кількості ESP32-пристроїв, кожен з яких ідентифікується унікальним ідентифікатором дрона. Взаємодія з апаратною частиною є однонапрямленою – дані надсилаються від пристрою до сервера, без потреби у зворотному зв’язку, що спрощує архітектуру та забезпечує стабільну передачу інформації.

### 3.1.3 Програмний інтерфейс

Інтерфейс програмного забезпечення системи Smart War Drones побудований на основі відкритого REST API, що забезпечує взаємодію між клієнтською частиною, сервером та IoT-пристроями. Серверна частина реалізована на платформі ASP.NET Core 8.0 і надає захищені ендпоінти для обробки запитів, включаючи автентифікацію користувачів, збереження та оновлення даних дронів, обробку статистики, обмін повідомленнями та управління бойовими напрямками.

Клієнтська частина, реалізована за допомогою React 19 + Vite, взаємодіє з API через HTTP-запити, використовуючи JWT-токени для авторизації та оновлюваний симетричний ключ для шифрування критично важливої інформації. Крім того, для обміну повідомленнями в реальному часі використовується протокол SignalR, що дозволяє оперативно надсилати оновлення користувачам без необхідності опитування сервера.

API розроблено з дотриманням принципів RESTful-архітектури та підтримує структуровані JSON-запити і відповіді, що забезпечує простоту розширення функціональності системи в майбутньому.

### 3.1.4 Комунікаційний протокол

Програмна система Smart War Drones використовує низку комунікаційних інтерфейсів для забезпечення надійного обміну даними між її компонентами. Основним каналом зв’язку між клієнтською частиною та сервером є протокол HTTP з підтримкою REST API-запитів. Незважаючи на відсутність HTTPS у локальному середовищі розгортання, безпека переданих даних гарантується за рахунок клієнтського шифрування інформації.

Для обміну повідомленнями та оновлення даних у реальному часі між користувачами та сервером використовується протокол WebSocket, реалізований через SignalR, що дозволяє миттєво надсилати повідомлення, статуси та оновлення статистики дронів.

IoT-компоненти, реалізовані на базі ESP32, передають дані на сервер за допомогою HTTP PUT-запитів через вбудований Wi-Fi модуль. Комунікація з периферійними датчиками (DHT22, потенціометри, кнопки) здійснюється через стандартні порти вводу/виводу мікроконтролера. Такий підхід забезпечує гнучкість та масштабованість системи.

### 3.1.5 Обмеження пам’яті

Розробка програмного забезпечення Smart War Drones враховує обмеження обсягів пам’яті, особливо на рівні IoT-пристроїв. Зокрема, мікроконтролер ESP32, який використовується у віртуальній симуляції Wokwi, має обмежену оперативну пам’ять (близько 520 КБ SRAM) та обсяг флеш-пам’яті (до 4 МБ), що вимагає оптимізованого коду та мінімального обсягу даних при кожному запиті.

На клієнтському рівні обмеження пам’яті залежать від ресурсів браузера користувача, однак використання SPA-архітектури (Single Page Application) дозволяє значно зменшити потребу в повторному завантаженні сторінок та даних, зберігаючи лише необхідну інформацію в оперативній пам’яті.

Серверна частина системи розгорнута у середовищі .NET Core, яке підтримує гнучке управління ресурсами та масштабування, тому суттєвих обмежень пам’яті не спостерігається. Проте для оптимальної роботи обрано зберігання тільки актуальних даних у пам’яті, а історичну інформацію – в MongoDB.

### 3.1.6 Операційні вимоги

Програмна система Smart War Drones розроблена таким чином, щоб забезпечити стабільну роботу у стандартних середовищах без необхідності спеціалізованого обладнання. Клієнтська частина є веб-застосунком, який коректно функціонує у сучасних браузерах (Chrome, Firefox, Edge) без додаткових модулів, підтримуючи екрани з різною роздільною здатністю.

Серверна частина може бути розгорнута як на локальному хості, так і на хмарному середовищі з підтримкою .NET 8.0 SDK, MongoDB і стандартного HTTP-сервера. Для коректної роботи потрібно мати доступ до портів, зазначених у конфігураціях.

IoT-компонент базується на ESP32 і потребує бездротового з’єднання Wi-Fi. Програмне середовище симуляції (наприклад, Wokwi) дозволяє відтворити всі операції без реального пристрою. Для роботи реального апаратного забезпечення достатньо середовища Arduino IDE або PlatformIO.

### 3.1.7 Функції продукту

Основні функції програмного забезпечення Smart War Drones охоплюють повний цикл планування та управління операціями з використанням дронів. Серед ключових функціональних можливостей:

* авторизація та автентифікація користувачів, зокрема з використанням JWT та змінного ідентифікатора;
* інтерактивна мапа, що дозволяє створювати напрямки, додавати дрони, вказувати маршрут та черги дій;
* реєстрація дронів та призначення типу (розвідка, атака, транспорт тощо) із прив’язкою до напрямку;
* месенджер між користувачами, як у приватному режимі, так і в контексті певного бойового напрямку;
* візуалізація та перегляд телеметричних даних з IoT-компонентів (температура, вологість, рівень заряду тощо);
* шифрування повідомлень між користувачами та збереження безпечної комунікації.

Кожна з цих функцій реалізована з урахуванням безпеки, ефективності та зручності користувача.

### 3.1.8 Припущення та залежності

У процесі розробки системи Smart War Drones було зроблено низку припущень, що впливають на стабільність та коректність її функціонування. Передбачається, що користувачі мають базові навички взаємодії з веб-застосунками, а доступ до мережі Інтернет є стабільним і не обмежується проксі або фаєрволами, які блокують API-запити.

Залежності системи включають:

* браузерне середовище, яке підтримує сучасні JavaScript API (ES6+) та забезпечує роботу SPA через React;
* серверна інфраструктура з попередньо встановленим .NET 8.0 SDK та MongoDB, а також з відкритими портами для прийому HTTP-запитів;
* IoT-платформа (ESP32 або Wokwi-емулятор) із підтримкою бібліотек для сенсорів DHT22, HTTPClient, WiFi.h тощо;
* використання відкритих криптографічних бібліотек на клієнтській та серверній стороні (CryptoJS, jsEncrypt, System.Security.Cryptography).

Також передбачається, що внутрішня мережа, в якій працює система, є довіреною, оскільки в локальному середовищі не використовується HTTPS. Це компенсується локальним шифруванням даних до їх відправки.

## 3.2 Властивості програмного продукту

Нижче наведено основні характеристики, які визначають можливості, поведінку та обмеження ПЗ:

* модульність: архітектура системи побудована на основі принципу модульності, що дозволяє окремим компонентам (сервер, фронтенд, IoT-пристрій) працювати незалежно;
* інтерактивність: фронтенд реалізований у вигляді SPA (Single Page Application), що забезпечує миттєве оновлення інтерфейсу без перезавантаження сторінок.
* безпека: передача чутливих даних (ідентифікатори користувачів, повідомлення, команди) здійснюється з попереднім шифруванням за допомогою гібридної криптографії: AES – для тіла повідомлення, RSA – для шифрування ключа, Diffie–Hellman – для обміну;
* підтримка реального часу: завдяки використанню SignalR, система дозволяє відображати оновлення статусів дронів, повідомлень у чаті та дій на мапі без потреби ручного оновлення даних.;
* гнучкість автентифікації: у системі реалізовано авторизацію за допомогою JWT-токенів, які підтримують автоматичне оновлення через refresh-токени;
* інтеграція IoT-компонентів: система підтримує симуляцію телеметричних даних з використанням модуля ESP32;
* надійність та стабільність: продукт розроблений із дотриманням сучасних стандартів проєктування та тестувався на кожному з етапів реалізації; присутні перевірки на помилки з відповідною обробкою виняткових ситуацій як на фронтенді, так і на бекенді;
* масштабованість: завдяки використанню NoSQL-бази даних MongoDB, можливо зберігати великі обсяги інформації про користувачів, дрони, мапи та сценарії, що дозволяє масштабувати систему для роботи у великих підрозділах.

Програмний продукт Smart War Drones володіє низкою функціональних та нефункціональних властивостей, які забезпечують ефективне використання системи в умовах бойових або навчальних операцій із залученням безпілотних літальних апаратів.

## 3.3 Атрибути програмного продукту

### 3.3.1 Надійність

Надійність є однією з ключових нефункціональних вимог до будь-якого програмного забезпечення, особливо коли мова йде про військові або критично важливі застосунки.

Архітектурні підходи до забезпечення надійності:

* програмний продукт реалізовано за принципами розділення логіки – frontend (інтерфейс користувача), backend (логіка обробки даних, безпека, база даних) та IoT (збір телеметричних даних із датчиків); такий підхід дозволяє локалізувати помилки, спростити відстеження несправностей і відновлення роботи окремих модулів без потреби зупиняти всю систему;
* серверна частина застосовує RESTful-підхід із дотриманням принципів stateless, що означає: кожен запит містить усю необхідну інформацію для його обробки – це дозволяє серверу зберігати менше станів, бути менш вразливим до помилок з боку клієнта і працювати стабільно під навантаженням;
* кожен API-ендпоінт на сервері містить механізми валідації вхідних параметрів – це запобігає несанкціонованому введенню або ін’єкціям, які могли б призвести до аварійних ситуацій;
* система використовує MongoDB, яка підтримує реплікацію, резервне копіювання та механізми журналювання змін; у разі аварійного завершення роботи системи або серверної частини, можливо відновити останній консистентний стан даних;
* під час передачі інформації з клієнтської частини застосовується локальне шифрування за допомогою AES, що дозволяє мінімізувати ймовірність втрати або компрометації даних у випадку перехоплення запиту;
* дані з IoT та дії користувача обробляються незалежно – це означає, що тимчасові збої одного каналу не впливають на загальну стабільність системи;
* інтерфейс користувача містить захист від помилкових дій: підтвердження перед видаленням, перевірку коректності введення, блокування небезпечних операцій у разі відсутності з’єднання з сервером тощо;
* усі помилки відображаються користувачу у вигляді повідомлень у дружньому форматі; також передбачено логування дій у консолі (на етапі розробки), що дозволяє швидко виявити та усунути несправності;
* у разі реального використання в умовах бойових дій або тренувань, система повинна бути здатна працювати автономно в локальній мережі без підключення до інтернету – це враховано під час розробки: локальне розгортання без HTTPS компенсується шифруванням запитів на стороні клієнта, автономна логіка дозволяє користувачам здійснювати базові операції навіть без надходження нових телеметричних даних, низькі системні вимоги до запуску сервера забезпечують стабільну роботу навіть на слабкому залізі.

У контексті розробки системи Smart War Drones надійність визначається як здатність системи функціонувати без збоїв, помилок або відмов протягом тривалого періоду часу та за умов інтенсивної взаємодії між клієнтською, серверною та IoT-компонентами.

### 3.3.2 Доступність

Доступність програмного забезпечення – це здатність системи бути готовою до використання у визначений або очікуваний час. Система має бути стабільно доступною як у межах локальної мережі, так і в режимі автономної роботи, незалежно від зовнішніх факторів, таких як інтернет-з'єднання, ресурсні обмеження або помилки користувача.

Підходи до забезпечення високої доступності:

* серверна частина застосунку може бути розгорнута в локальній мережі без необхідності підключення до зовнішнього інтернету – це дозволяє використовувати систему у закритих середовищах, де обмежено або відсутнє зовнішнє з’єднання – наприклад, на полігонах, мобільних командних пунктах або в польових умовах;
* клієнтська частина (SPA, реалізована на React) після першого завантаження повністю працює у браузері користувача; у разі тимчасової недоступності серверної частини або втрати зв’язку система продовжує відображати останні завантажені дані – це дозволяє користувачу переглядати інформацію навіть без активного з’єднання.
* програмний продукт не вимагає складної інсталяції чи розгортання. Сервер запускається однією командою з Visual Studio або CLI, фронтенд – за допомогою Vite або будь-якого http-сервера – це дозволяє безперешкодно запускати систему на будь-якому ПК, ноутбуці або навіть на компактних рішеннях на зразок Raspberry Pi;
* API-запити з клієнта та IoT-модуля не залежать від стану інших компонентів; у разі тимчасової недоступності бази даних (MongoDB), сервер має систему повторних спроб і обробки помилок, що знижує ризик втрати інформації.;
* використання SignalR дозволяє забезпечити обмін даними між сервером і клієнтами в реальному часі;
* система підтримує механізми автоматичного повторного підключення як для WebSocket, так і для запитів до API; у разі втрати з’єднання користувач отримає повідомлення про статус підключення, а самі дані будуть синхронізовані після відновлення каналу;
* серверна частина реалізована на .NET Core із використанням асинхронних методів (async/await) – це дозволяє обробляти десятки одночасних запитів без блокування потоку, навіть на слабкому серверному обладнанні;
* ESP32 у симуляції Wokwi надсилає дані на сервер із певним інтервалом; якщо підключення відсутнє, ESP32 просто пропускає цикл передачі та продовжує роботу – це забезпечує безперебійну роботу сенсорної частини та дозволяє уникати зависань або збоїв;
* усі компоненти системи працюють повністю офлайн; не використовуються хмарні API, централізовані служби аутентифікації чи зовнішні бази даних – це гарантує, що доступність не буде залежати від політики сторонніх постачальників чи інтернет-провайдерів.

Для системи Smart War Drones, яка орієнтована на використання в умовах підвищеної відповідальності, зокрема у військових або навчально-тактичних операціях, доступність є критично важливою.

### 3.3.3 Безпека

Безпека є одним із критично важливих аспектів програмного забезпечення Smart War Drones, оскільки система призначена для використання в умовах, де можливе перехоплення, підміна або саботаж інформації. Захист даних, що передаються, обробляються чи зберігаються, реалізований на декількох рівнях – як на стороні клієнта, так і на сервері, а також у каналі передачі між ними.

Уся передача даних відбувається із застосуванням гібридної криптографії. Тіло повідомлень шифрується за допомогою симетричного алгоритму AES, що забезпечує швидкість та ефективність, а сам ключ шифрування передається зашифрованим через RSA. Для обміну симетричними ключами між користувачами, зокрема в модулі чату, використовується протокол Diffie-Hellman, що унеможливлює їхнє перехоплення третьою стороною навіть у разі доступу до каналу зв’язку. Це гарантує, що жоден із користувачів не зможе підробити повідомлення або отримати доступ до конфіденційного листування.

Для збереження облікових даних реалізовано хешування паролів за допомогою алгоритму Argon2 – переможця конкурсу Password Hashing Competition (PHC), що демонструє високу стійкість до атак типу brute-force та side-channel. Навіть у разі компрометації бази даних паролі залишаться недоступними без знання оригінального введення користувача. Усі хеші додатково "соляться" динамічним значенням, що унеможливлює використання rainbow-таблиць.

У системі реалізовано механізм авторизації з використанням JWT (JSON Web Token), що дозволяє підтверджувати справжність користувача без повторного введення пароля. Токени мають обмежений строк дії, а також оновлюються за допомогою окремого refresh-токена. Це дозволяє зберігати баланс між зручністю для користувача та безпекою, знижуючи ризик викрадення сесій.

Особливістю реалізації Smart War Drones є локальне шифрування даних на стороні клієнта перед їх передачею через HTTP, навіть у межах локального середовища, де відсутній HTTPS. Таким чином, навіть у разі несанкціонованого перехоплення трафіку всередині мережі, зловмисник не зможе розшифрувати вміст без відповідного ключа. Також, кожен користувач отримує власну пару ключів – відкритий і закритий, які ніколи не покидають межі його пристрою, що додатково унеможливлює компрометацію конфіденційних даних.

Серверна частина програмного забезпечення має систему логування дій, яка дозволяє відслідковувати підозрілі активності та потенційні спроби несанкціонованого доступу. Водночас, усі критичні операції (наприклад, видалення об’єктів, оновлення профілю, надсилання повідомлень) вимагають автентифікованого запиту, що додатково обмежує поверхню атаки.

Отже, в системі Smart War Drones реалізовано багаторівневу модель безпеки, що поєднує сучасні алгоритми шифрування, захищену автентифікацію, стійке хешування, обмеження доступу та логування. Це дозволяє гарантувати конфіденційність, цілісність і автентичність даних навіть в умовах ворожого оточення та потенційних атак.

### 3.3.4 Супроводжуваність

Супроводжуваність програмного забезпечення Smart War Drones є одним із ключових атрибутів, які забезпечують довготривалу ефективність і життєздатність системи. У процесі розробки враховувалася потреба в простому оновленні, розширенні та підтримці коду, що особливо важливо для систем, які функціонують у складних та динамічних середовищах, таких як військові застосунки.

Для досягнення високого рівня супроводжуваності в архітектурі проєкту було реалізовано модульний підхід. Усі основні компоненти системи – авторизація, обробка користувачів, чат, управління напрямками, робота з мапою, облік дронів та статистики — реалізовані як окремі ізольовані модулі з чітко визначеними зонами відповідальності. Це дозволяє змінювати, виправляти або розширювати функціонал одного з модулів без ризику порушити роботу інших частин системи. Завдяки цьому при виявленні помилки або додаванні нової функції не потрібно переглядати всю систему повністю.

Усі програмні компоненти написані з дотриманням загальноприйнятих принципів чистого коду, таких як зрозумілі назви змінних, послідовне форматування, належне коментування складних ділянок логіки та використання шаблонів проєктування там, де це доречно. Це суттєво полегшує процес читання, розуміння та зміни програмного коду іншими розробниками або підтримуючим персоналом у майбутньому.

Серверна частина системи реалізована на ASP.NET Core 8.0, що забезпечує стабільність і передбачуваність у розгортанні, а також підтримується широкою спільнотою розробників і офіційною документацією. Для роботи з базою даних MongoDB використовується репозиторний підхід, що дозволяє легко адаптувати запити та розширювати структуру даних у разі зміни вимог. Клієнтська частина побудована за допомогою React 19 з інструментом збірки Vite, що дозволяє швидко адаптувати інтерфейс до змін або додавання нових функцій, не переписуючи всю логіку з нуля.

Особливу увагу приділено документації внутрішніх API, структур даних і функціональності. Наявність зрозумілих назв маршрутів, описів у контролерах, а також логування запитів і відповідей робить систему більш передбачуваною і дозволяє швидко діагностувати проблеми під час тестування або в процесі експлуатації.

Ще однією важливою характеристикою є підтримка емуляції та модульного тестування в середовищі Wokwi для IoT-компонента. Завдяки цьому підтримка сенсорної частини системи не вимагає фізичного обладнання – будь-які зміни або перевірки можна провести в онлайн-режимі, а потім легко перенести на реальний пристрій.

Таким чином, завдяки модульності, структурованості, вибору сучасного та підтримуваного інструментарію, належному оформленню коду та документації, система Smart War Drones характеризується високою супроводжуваністю, що дозволяє адаптувати її під нові умови, легко оновлювати, масштабувати та підтримувати протягом усього життєвого циклу.

### 3.3.5 Переносимість

Переносимість програмного забезпечення Smart War Drones визначає його здатність функціонувати в різних обчислювальних середовищах з мінімальними змінами або без них. Цей аспект був врахований ще на етапі проєктування архітектури, щоб гарантувати зручне розгортання системи як у локальних умовах, так і на віддалених серверах, з можливістю масштабування відповідно до потреб користувачів.

Серверна частина системи реалізована на основі кросплатформного фреймворку ASP.NET Core 8.0, який дозволяє розгортати сервер як на Windows, так і на Linux або macOS без необхідності змінювати програмний код. Усі зовнішні залежності, такі як бази даних, сховища ключів або бібліотеки шифрування, є доступними для різних операційних систем і встановлюються за допомогою стандартних менеджерів пакетів. Це дає змогу створити контейнери або сценарії автоматизованого розгортання, які працюють однаково у різних середовищах, включно з хмарною інфраструктурою.

Клієнтська частина, створена з використанням React 19 та збирається за допомогою Vite, також є повністю переносимою. Готовий фронтенд-бандл може бути розміщений на будь-якому вебсервері, включно з Nginx, Apache або навіть сервісами, що надають статичний хостинг, такими як GitHub Pages або Netlify. Завдяки використанню архітектури SPA (Single Page Application), вебінтерфейс не залежить від типу сервера чи мови бекенда, що дозволяє легко масштабувати чи інтегрувати його з іншими системами.

База даних MongoDB також відома своєю кросплатформністю: вона може бути розгорнута як локально, так і у вигляді хмарного рішення, наприклад MongoDB Atlas. Всі колекції та схеми є універсальними і не прив’язані до конкретної операційної системи чи апаратної архітектури, що полегшує міграцію даних та резервне копіювання.

Окрему увагу варто приділити IoT-компоненту, реалізованому на базі мікроконтролера ESP32. Завдяки використанню популярного середовища Arduino та віртуального симулятора Wokwi, прошивку можна запускати, тестувати і відлагоджувати без фізичного пристрою. Переносимість у цьому контексті забезпечується як на рівні середовища розробки (Arduino IDE, PlatformIO), так і на рівні коду, який є незалежним від конкретного виробника плати ESP32. Це дозволяє перенести систему з віртуального симулятора на реальний пристрій без зміни логіки роботи чи повторного написання програмного коду.

Ще одним фактором, що позитивно впливає на переносимість, є мінімальне використання апаратно-залежних функцій або закритих платформ. Усі критичні компоненти побудовані з відкритим вихідним кодом або мають чітко задокументовані API, що дає змогу при необхідності адаптувати систему до нових вимог, наприклад, при переході на іншу хмарну інфраструктуру або апаратну платформу.

Таким чином, завдяки використанню сучасних, кросплатформних технологій, відкритих стандартів, модульній структурі системи, відсутності жорсткої прив’язки до середовища виконання або конкретного обладнання, програмне забезпечення Smart War Drones демонструє високий рівень переносимості, що дозволяє легко адаптувати його до нових умов експлуатації без значних витрат часу та ресурсів.

### 3.3.6 Продуктивність

Продуктивність програмного забезпечення Smart War Drones є критичним атрибутом, оскільки від нього залежить здатність системи ефективно функціонувати в режимі реального часу за умов значного навантаження. У проєкті було приділено особливу увагу забезпеченню швидкого відгуку користувацького інтерфейсу, оптимізації обміну даними між клієнтом і сервером, а також мінімізації затримок при надходженні та обробці телеметричної інформації з дронів або IoT-модулів. Важливо, щоб навіть при збільшенні кількості активних бойових напрямків, користувачів або дронів система не втрачала стабільності та залишалась придатною для безперервного управління.

На стороні клієнта продуктивність досягається за рахунок використання сучасного JavaScript-фреймворку React, що забезпечує віртуальний DOM та ефективне оновлення інтерфейсу. Додатково застосовується Vite як інструмент збірки, що гарантує надзвичайно швидкий запуск розробницького середовища та зменшує обсяг бандлу в продакшн-збірці. Це дозволяє клієнтському додатку працювати швидко навіть на пристроях із середніми характеристиками.

На сервері застосовується високопродуктивний фреймворк ASP.NET Core 8.0, який здатен обслуговувати велику кількість одночасних запитів із мінімальними затримками. Всі API-ендпоінти реалізовані з урахуванням асинхронності, що дозволяє ефективно масштабувати обробку паралельних запитів без блокування ресурсів. Збереження статистики дронів, обробка чатів, оновлення мапи – усе це відбувається з мінімальною затримкою завдяки правильному проєктуванню бази даних MongoDB та її колекцій.

Крім того, продуктивність обміну даними була покращена шляхом застосування легковагових форматів передачі (JSON) та попереднього шифрування повідомлень, що дозволяє не витрачати ресурси сервера на розшифрування безпосередньо під час передачі. Інтеграція із SignalR дає можливість використовувати вебсокети для миттєвого надсилання оновлень, що особливо важливо для бойових напрямків, де швидкість синхронізації між користувачами має вирішальне значення.

IoT-компонент також оптимізований на рівні коду: дані з ESP32 надсилаються на сервер раз на хвилину, а сам обсяг повідомлення мінімізовано для економії пропускної здатності мережі. Завдяки цьому навіть при роботі декількох десятків пристроїв система зберігає стабільність і не перевантажує сервер.

Усі ці фактори – асинхронна архітектура, використання швидкодійних технологій, ефективні протоколи передачі даних та обмеження на обсяг запитів – забезпечують високу продуктивність системи Smart War Drones у типових сценаріях її використання. Це дозволяє забезпечити своєчасне оновлення даних, низький час відгуку, відсутність зависань навіть при піковому навантаженні, що є необхідним для будь-якої військової або критично важливої програмної системи.

## 3.4 Вимоги бази даних

У проєкті Smart War Drones базу даних розроблено з урахуванням потреб зберігання динамічної, структуровано пов’язаної та конфіденційної інформації, яка пов’язана з користувачами, дронами, статистикою польотів, бойовими напрямками, чатами та логами взаємодії. Вибір бази даних MongoDB обумовлений її документно-орієнтованою природою, що дозволяє швидко працювати з вкладеними структурами, динамічно змінювати схеми збереження даних без необхідності складної міграції та забезпечувати гнучкість масштабування. MongoDB добре підходить до сценаріїв, у яких передбачено часте оновлення документів, агрегацію статистичних даних, зберігання історії змін, а також одночасне обслуговування великої кількості користувачів.

База даних зберігає всі сутності, необхідні для функціонування системи. Кожен користувач має унікальний запис, який включає облікові дані, роль у системі (офіцер або оператор), набір токенів для авторизації, список контактів і пов’язані профільні параметри. Для кожного дрона передбачено окремий документ, що зберігає зашифрований ідентифікатор, тип, статус, приналежність до напрямку, а також масив оновлюваної статистики, що надходить із ESP32.

Бойові напрямки зберігаються у вигляді окремих документів, що містять координати фокусної точки мапи, масштаб, набір прив’язаних дронів та список учасників-операторів. Усі ці дані можуть бути оновлені без перезавантаження структури документа, що забезпечує високу гнучкість при редагуванні. Інтерактивна мапа також базується на збережених параметрах тайлів, стилів і координатної сітки. Статистика роботи кожного дрона зберігається у вигляді типізованих записів у відповідній колекції, з прив’язкою до типу (наприклад, температура, вітер, акумулятор, РЕБ), часу останнього оновлення та шифрованого ID дрона.

Особливе місце в базі даних займає система повідомлень. Реалізовано зберігання як приватних, так і групових повідомлень у вигляді масивів або окремих колекцій, що дозволяє ефективно реалізовувати перегляд історії спілкування, фільтрацію за користувачами або напрямками. Усі повідомлення також можуть зберігатися в зашифрованому вигляді, що гарантує конфіденційність навіть на рівні бази.

Ще однією важливою вимогою є забезпечення цілісності даних. Це досягається шляхом використання валідаторів, унікальних індексів та перевірок на рівні сервісів взаємодії з базою. Додатково забезпечено логування критичних змін, що дозволяє відстежити дії користувачів у разі інцидентів або технічних несправностей.

Вимоги до бази даних передбачають також можливість розгортання у хмарному середовищі або на локальному сервері з подальшим масштабуванням. Архітектура дозволяє адаптувати MongoDB до кластерного середовища, резервного копіювання та реплікації даних. Усі ці аспекти спрямовані на забезпечення стабільної, безпечної, продуктивної та масштабованої роботи інформаційної частини системи, незалежно від кількості користувачів або навантаження на систему.

## 3.5 Інші вимоги

Окрім функціональних, архітектурних та безпекових аспектів, програмне забезпечення Smart War Drones має відповідати ряду додаткових вимог, які визначають його повну готовність до використання в умовах, наближених до реального бойового середовища. Ці вимоги охоплюють зручність розгортання, локалізацію, обмеження з боку користувацького обладнання, особливості інтерфейсу взаємодії з IoT-пристроями, а також узгодження роботи програмних компонентів у мережевому середовищі з обмеженим доступом до ресурсів.

Враховуючи сценарій застосування системи у закритих внутрішніх мережах військових частин, важливою вимогою є здатність до швидкого розгортання на локальних серверах без необхідності інтернет-підключення. Усі клієнтські бібліотеки, стилі та картографічні дані мають бути заздалегідь включені до проєкту у вигляді офлайн-ресурсів, що забезпечить повноцінне функціонування навіть у середовищах із суворими мережевими обмеженнями. Також необхідно передбачити можливість конфігурації IP-адреси сервера вручну для взаємодії з IoT-модулями, які не використовують DNS або зовнішній NAT.

Окремо визначено вимоги до надійності обміну даними з IoT-пристроями, що працюють на основі мікроконтролера ESP32. Оскільки симуляція в середовищі Wokwi моделює з’єднання з віртуальними датчиками, розроблена логіка має бути легко портованою на реальні пристрої. Комунікація повинна бути побудована за протоколом HTTP PUT з JSON-тілом запиту, що уможливлює сумісність з більшістю популярних бібліотек ESP-платформи. Також важливою є стійкість до втрати мережевого з'єднання та повторна спроба передачі даних у разі невдалої спроби.

Ще однією вимогою є підтримка читабельного та локалізованого інтерфейсу. Інтерфейс системи має бути адаптований до української мови як основної, при цьому структура компонентів повинна передбачати легке додавання додаткових мовних файлів без переписування логіки. Це реалізовано через зовнішні конфігурації, що зберігають текстові ресурси для компонентів, заголовків, кнопок, повідомлень про помилки тощо.

Система повинна бути максимально інтуїтивно зрозумілою для користувача з базовими навичками роботи з комп’ютером. Тому особлива увага приділяється ергономіці – розміщенню елементів управління, кольоровій палітрі, анімаціям, розмірам шрифтів та адаптивності до різних роздільностей екрану. Усі модальні вікна, панелі та інтерфейси побудовано з урахуванням сценаріїв мобільного використання або застосування на пристроях із невеликою діагоналлю.

Додаткові вимоги включають наявність журналу подій на сервері, що дозволяє фіксувати дії користувачів, спроби авторизації, оновлення статистики дронів та інші критичні дії, що впливають на загальну безпеку. Цей журнал має бути доступним лише для користувачів із відповідними адміністративними правами, а його записи повинні бути захищені від несанкціонованого видалення або підміни.

Таким чином, перелічені інші вимоги спрямовані на те, щоб забезпечити повну готовність програмного забезпечення до використання у складних та відповідальних умовах, гарантувати зручність, безпеку, гнучкість розгортання та надійність комунікацій між усіма підсистемами.