**Software Requirements Specification**

для

EnviroScan Research Platform

Дослідження методів просторового сканування навколишнього середовища з використанням Wi‑Fi‑сигналів та алгоритмів машинного навчання

Версія 1.0 – червень 2025 р.

Підготував: Нос М. В., магістрант групи ІПЗм‑23‑2

Харківський національний університет радіоелектроніки, кафедра програмної інженерії

Зміст

1. Вступ
2. Загальний опис
3. Вимоги до зовнішніх інтерфейсів
4. Функціональні можливості системи
5. Інші нефункціональні вимоги
6. Додаткові вимоги  
   Додаток А. Глосарій  
   Додаток B. Аналітичні моделі  
   Додаток C. Перелік питань, що залишилися невизначеними

1. Вступ

1.1 Призначення

Цей документ визначає вимоги до програмно‑апаратної платформи EnviroScan, що створюється в межах магістерського дослідження. SRS охоплює повний обсяг функціональності прототипу та супровідного програмного забезпечення, необхідних для збору Wi‑Fi‑даних, їхнього оброблення, побудови 3D‑моделей та оцінювання алгоритмів машинного навчання.

1.2 Умовні позначення та форматування

* UC‑x – ідентифікатор випадку використання.
* Вимоги позначено префіксами REQ‑Fx (функціональна) або REQ‑NFx (нефункціональна).
* Усі пріоритети мають три рівні: High (H), Medium (M), Low (L).

1.3 Цільова аудиторія та порядок читання

Документ призначений для:

* наукового керівника та рецензента;
* студентів‑розробників;
* технічних консультантів (апаратна частина, радіотехніка);
* потенційних користувачів‑дослідників.

Рекомендується спершу ознайомитися з розділами 1–2, а далі переходити до розділу 4, релевантного сфері відповідальності читача.

1.4 Сфера продукту

EnviroScan — дослідницька платформа, що уніфіковує процес сканування оточення за допомогою Wi‑Fi, подальшу обробку сигналів та побудову 3D‑репрезентації простору. Результати слугуватимуть підґрунтям для застосувань у AR/VR, робототехніці та системах «розумного» простору.

1.5 Посилання

1. DensePose from WiFi, arXiv 2301.00250, 2023.
2. MM‑Fi Dataset, arXiv 2305.10345, 2023.
3. Методичні вказівки до ККП, ХНУРЕ 2024.
4. IEEE 830‑1998 Software Requirements Specification Standard.

2. Загальний опис

2.1 Перспектива продукту

EnviroScan є новим самостійним продуктом, що використовує наявний стек: модулі ESP32, роутери Wi‑Fi 6, Python/TensorFlow та середовище Unity для візуалізації. Прототип взаємодіє з лабораторною мережевою інфраструктурою кафедри.

2.2 Функції продукту (огляд)

* Z1. Збір необроблених I/Q‑даних Wi‑Fi‑кадрів.
* Z2. Синхронізація міток часу та геометрії сцени.
* Z3. Попередня обробка та фільтрація шумів.
* Z4. Побудова карти відбиттів (CSI‑heat‑map).
* Z5. Реконструкція 3D‑моделі об’єктів.
* Z6. Навчання та оцінювання ML‑моделей (CNN, GNN).
* Z7. Візуалізація результатів через Web‑інтерфейс / Unity.
* Z8. Генерація звітів та експорт даних.

2.3 Класи користувачів

|  |  |  |
| --- | --- | --- |
| Клас | Характеристики | К‑сть |
| Дослідник (Researcher) | проводить експерименти, аналізує дані | 1 |
| Технік (Lab Technician) | налаштовує обладнання, стежить за безпекою | 1 |
| Розробник (Developer) | модифікує код та ML‑плагіни | 1 |
| Керівник (Supervisor) | переглядає звіти, задає параметри | 1 |

2.4 Робоче середовище

* Апаратне: ПК ≥ 16 GB RAM, модулі ESP32, антенний масив, роутер Wi‑Fi 6.
* Програмне: Ubuntu 22.04 LTS, Python 3.11, TensorFlow 2.16, PyTorch 2, Arduino IDE 2, Unity 2022 LTS.

2.5 Обмеження розробки

* Використання винятково безплатних або академічних ліцензій.
* Дотримання санітарних норм: потужність випромінювання ≤ 100 mW EIRP.
* Завершення проєкту — червень 2025 р.

2.6 Документація користувача

* Посібник зі встановлення обладнання (PDF).
* Демонстраційний проєкт Unity.

2.7 Припущення та залежності

* Наявність щонайменше трьох активних точок доступу у спектрі 2,4 / 5 GHz.
* Безперебійне живлення.
* Доступ розробників до GPU (NVIDIA RTX 3060 або вище).

3. Вимоги до зовнішніх інтерфейсів

3.1 Користувацькі інтерфейси

* CLI‑1: скрипт з параметрами командного рядка.
* VR‑Viewer: сцена Unity з інтерактивним переміщенням по відсканованій карті.

3.2 Апаратні інтерфейси

* UART 115200 bps для прошивки ESP32.
* SPI для модуля ADC під час зняття показів із зовнішніх датчиків.
* PoE IEEE 802.3af для живлення роутера експериментальної мережі.

3.3 Програмні інтерфейси

* Python‑WebSocket (websockets).
* Arduino → COM‑порт ПК.
* Скрипт Unity C#.

3.4 Комунікаційні інтерфейси

* Wi‑Fi IEEE 802.11ax, канал 36, ширина 80 MHz.
* Дротовий COM‑порт.

4. Функціональні можливості системи

4.1 System Feature 1: Захоплення Wi‑Fi‑RSSI (REQ‑F1)

Опис. Прототип складається з 1 Arduino та 5 ESP‑01. Кожна ESP двічі на секунду надсилає RSSI і таймштамп; Arduino агрегує дані та передає пакет у COM‑порт ПК, де Python‑скрипт використовує ці значення. Пріоритет H.

4.1.1 Сценарій стимул/реакція

UC‑1: студент запускає скрипт, відкриває COM 9600 bps → приймає JSON‑рядки → записує їх у БД або передає далі до моделі.

4.1.2 Функціональні вимоги

* REQ‑F1‑1: мінімум 10 повних записів/с.
* REQ‑F1‑2: втрати кадрів ≤ 10 % під час безперервного запису.
* REQ‑F1‑3: кожен рядок має поля ts\_iso, esp\_id, rssi\_dbm.

4.1.3 Особливі зауваги щодо обладнання

Макет‑плата живиться від 5 V та 3,3 V через модуль живлення; антистатичний корпус відсутній.

4.2 System Feature 2: Підготовка датасета та масштабування (REQ‑F2)

Опис. Скрипт prepup.py виконує повний цикл підготовки даних: зчитує таблицю rssi\_data з MySQL, підвантажує воксельну карту room\_voxels.npy, агрегує RSSI у 34 підгрупи, додає position\_idx та масштабує через StandardScaler (сериалізується у scaler.pkl). Вихідні артефакти: X\_train.npy, X\_test.npy, y\_train.npy, y\_test.npy. Пріоритет H.

4.2.1 Сценарій стимул/реакція

UC‑2: студент виконує python prepup.py → скрипт під’єднується до MySQL → читає ≈ 5 000 рядків RSSI → готує та зберігає набори даних, виводить їхню форму.

4.2.2 Функціональні вимоги

* REQ‑F2‑1: підтримувати MySQL‑рядок підключення у .env або CLI‑параметрі --dsn.
* REQ‑F2‑2: час підготовки ≤ 60 с на CPU i5‑8400 для 10 k рядків.
* REQ‑F2‑3: різниця між кількістю груп RSSI та шарів воксельної карти ≤ 1 %.
* REQ‑F2‑4: position\_idx не нормалізується (mean = 0, scale = 1).

4.3 System Feature 3: Навчання 3D‑автоенкодера (REQ‑F3)

Опис. prepup.py формує та навчає повнозв’язний 3D‑автоенкодер (9 Dense‑шарів, 16384 → 256 → 16384 нейронів) для реконструкції воксельної карти 34×19×41 із 26‑вимірного RSSI‑вектора. Навчання відбувається з ранньою зупинкою (patience 100), найкраща модель зберігається у best\_autoencoder.keras. Пріоритет H.

4.3.1 Сценарій стимул/реакція

UC‑3: python prepup.py --train → 10 000 епох або рання зупинка → консольний лог втрати та точності → файл моделі.

4.3.2 Функціональні вимоги

* REQ‑F3‑1: підтримка відновлення з чекпойнта через --resume.
* REQ‑F3‑2: втрата binary\_crossentropy на валідації ≤ 0,05.
* REQ‑F3‑3: автоматично генерувати train\_history.png із графіками loss та accuracy.

4.4 System Feature 4: Оцінювання моделі та метрики (REQ‑F4)

Опис. Після навчання скрипт обчислює Intersection‑over‑Union та Dice Coefficient для кожного шару тестового набору, виводить середні значення та зберігає metrics.json. Обраний зріз візуалізується у Matplotlib, повна карта — у PyVista. Пріоритет M.

4.4.1 Сценарій стимул/реакція

UC‑4: виконання блоку «Evaluation» у prepup.py → на екран друкуються IoU / Dice → відкриваються два вікна з істинною та передбаченою картами.

4.4.2 Функціональні вимоги

* REQ‑F4‑1: середній IoU ≥ 0,60.
* REQ‑F4‑2: середній Dice ≥ 0,75.
* REQ‑F4‑3: зберігати різницеву воксельну карту у diff\_map.vtk.

4.5 System Feature 5: Режим реального часу — inference loop (REQ‑F5)

Опис. Асинхронна корутина inference\_loop (файл use.py) читає JSON‑повідомлення RSSI з COM‑порту 9600 bps, агрегує кожні 3 рядки, масштабує через scaler.pkl, пропускає через best\_autoencoder.keras, підставляє отриманий зріз до тимчасової 3D‑моделі та формує список заповнених вокселів. Пріоритет H.

4.5.1 Сценарій стимул/реакція

UC‑5: студент запускає python live\_ws\_server.py → після зчитування 34 зрізів (≈ 20 с) формується повна карта → координати транслюються у веб‑клієнти.

4.5.2 Функціональні вимоги

* REQ‑F5‑1: затримка від прийому RSSI до відправлення WebSocket ≤ 0,5 с.
* REQ‑F5‑2: допускається втрата не > 10 % пакетів COM‑порту без переривання циклу.
* REQ‑F5‑3: очищати буфер та рахувати групи mod 34, забезпечуючи безкінечний цикл.

4.6 System Feature 6: Сервер WebSocket‑трансляції (REQ‑F6)

Опис. Скрипт live\_ws\_server.py піднімає WebSocket‑сервер (websockets.serve) на порті 8080, підтримує множину під’єднаних клієнтів та надсилає пакети формату { "vox": [ {x,y,z}, … ] }. Пріоритет M.

4.6.1 Сценарій стимул/реакція

UC‑6: браузер‑клієнт підключається до ws://127.0.0.1:8080 → отримує та рендерить потокові оновлення кожні ≈ 20 с.

4.6.2 Функціональні вимоги

* REQ‑F6‑1: підтримка ≥ 5 активних клієнтів без деградації швидкості.
* REQ‑F6‑2: автоматично видаляти «мертві» клієнти в разі помилки send.
* REQ‑F6‑3: виводити log‑рядок [WS] sent N vox to K clients у stdout.

4.7 System Feature 7: Фінальна 3D‑візуалізація (REQ‑F7)

Опис. Після завершення експерименту (Ctrl + C) функція \_show\_final\_map у use.py відображає у PyVista повну воксельну карту, дозволяючи масштабування й обертання сцени. Пріоритет L.

4.7.1 Сценарій стимул/реакція

UC‑7: користувач завершує збір даних → PyVista відкриває вікно з 3D‑точками → користувач аналізує результат.

4.7.2 Функціональні вимоги

* REQ‑F7‑1: побудова сцени ≤ 2 с при 10 k вокселів.

4.8 System Feature 8: Конфігураційні файли та артефакти (REQ‑F8)

Опис. Усі шляхи та параметри зберігаються у config.json, що містить координати приймачів ESP1…ESP5 та межі кімнати. Скрипти перевіряють наявність критичних артефактів (best\_autoencoder.keras, scaler.pkl, room\_voxels.npy) на старті. Пріоритет M.

4.8.1 Сценарій стимул/реакція

UC‑8: студент редагує config.json → запускає prepup.py або live\_ws\_server.py → у разі відсутності файла програма завершується з кодом 1 та відповідним логом.

4.8.2 Функціональні вимоги

* REQ‑F8‑1: усі шляхи в config.json можуть задаватися відносно змінної $PROJECT\_ROOT.
* REQ‑F8‑2: при старті скрипти виводять контрольні SHA‑256‑суми кожного артефакту.

5. Інші нефункціональні вимоги

5.1 Продуктивність

* REQ‑NF‑PERF‑1: затримка шляху «RSSI → WebSocket» ≤ 500 мс на ПК Intel i5‑8400 16 GB RAM.
* REQ‑NF‑PERF‑2: пікове споживання RAM скрипта use.py ≤ 4 GB.
* REQ‑NF‑PERF‑3: повний цикл навчання (prepup.py --train) завершується ≤ 24 год на GPU RTX 3060 (10 k епох, batch 1).

5.2 Надійність та відмовостійкість

* REQ‑NF‑REL‑1: усі необроблені винятки логуються у data\_validation.log.
* REQ‑NF‑REL‑2: при втраті COM‑порту скрипт виконує автоматичне перепідключення.
* REQ‑NF‑REL‑3: якщо GPU недоступний, навчання переходить у CPU‑режим із попередженням.

5.3 Безпека

* REQ‑NF‑SEC‑1: WebSocket‑сервер слухає лише 127.0.0.1 за замовчуванням; зовнішній доступ увімкнено прапорцем --public.
* REQ‑NF‑SEC‑2: DSN MySQL та ключі зберігаються у .env, не потрапляють до Git.

5.4 Портативність

* REQ‑NF‑PORT‑1: скрипти запускаються без модифікацій під Windows 10 Pro 21H2 та Ubuntu 22.04 LTS.

5.5 Екологічні та безпекові норми

* REQ‑NF‑ENV‑1: максимальний рівень EM‑поля біля антени ≤ 28 V/m (2,4 GHz) на 20 см.
* REQ‑NF‑ENV‑2: усі друковані плати обладнано запобіжником 250 mA.

6. Додаткові вимоги

6.1 Подальший розвиток

|  |  |  |
| --- | --- | --- |
| Код | Опис | ETA |
| FUT‑1 | Підтримка збору CSI (Intel 5300 або Nexmon Pi) | 12.2025 |
| FUT‑2 | Перехід на 3D‑U‑Net для реконструкції | 01.2026 |
| FUT‑3 | Live‑рендеринг у VR‑окулярах через Unity WebGL | 03.2026 |

6.2 Аналіз ризиків

|  |  |  |  |
| --- | --- | --- | --- |
| Ризик | Імовірність | Вплив | План реагування |
| R‑1: відсутність GPU | висока | середній | використовувати безкоштовний Google Colab Pro; скоротити епохи |
| R‑2: зашумлений Wi‑Fi у гуртожитку | середня | високий | тестування в нічний час; резерв каналу 40 MHz |
| R‑3: вихід із ладу ESP‑модуля | низька | середній | тримати один резервний ESP‑01 |

Додаток А. Глосарій

* CSI — Channel State Information.
* IoU — Intersection over Union.
* Dice — Dice Coefficient.
* PyVista — бібліотека для 3D‑візуалізації на Python.
* UART / COM‑порт — послідовний інтерфейс передачі даних.
* ESP32 / ESP‑01 — мікроконтролери з вбудованим Wi‑Fi.

Додаток B. Аналітичні моделі

* UML Sequence UC‑1: Arduino → PC → MySQL (захоплення).
* UML Sequence UC‑5: PC → AutoEncoder → WS‑Clients (live‑stream).
* DFD‑Level 1: вхід COM → buff → scaler → model → voxel map → WS.
* JSON‑Schema пакета vox:  
  { "vox": [ { "x": int, "y": int, "z": int } ] }.

Додаток C. Перелік питань, що залишилися невизначеними

1. Формат компресії для архівування великих послідовностей вокселів.
2. Можливість використання LoRa як допоміжного каналу синхронізації.