**ПРИЛОЖЕНИЕ 1**

**ТЕМА:** Noctis: Устройство за следене и подобряване на съня.

**Номер на проект: 695**


**АВТОР 1:**

**Име:** Андреан Башкехайов

**ЕГН:** 0850177542

**Адрес:** ул Иван Вазов 15, с. Черганово, област Стара Загора.

**GSM:** 0895269034

**E-mail:** andrean1710taja1234@gmail.com

**Специалност:** Приложно програмиране

**АВТОР 2:**

**Име:** Георги Бенев

**ЕГН:** 0852197548

**Адрес:** с. Дъбово област Стара Загора ул. 9ти септември 70

**GSM:** 0889984210

**E-mail:** bggesha2@gmail.com

**Специалност:** Приложно програмиране

**РЪКОВОДИТЕЛ:**

**Име:** Теодора Димитрова Колева

**GSM:** 0898241570

**E-mail:** teodora.di.koleva@edu.mon.bg

**Позиция:** Учител по информатика и информационни технологии

**РЕЗЮМЕ:**

**Цели:**

- Да се предостави система, която позволява на човек да следи своят сън и промените в него, чрез "**хардуерното устройство, което изпраща данни към API"**.
- Да се поддържа многопотребителска среда (multi-tenancy) със силна изолация на данните и сигурност в софтуерния бекенд.
- Да се имплементира пълен MLOps жизнен цикъл като "**гаранция за дългосрочна точност и надеждност"**. Това включва версиониране на модели, презареждане в реално време и автоматизирана детекция на дрейф на концепцията (concept drift), които гарантират, че системата се адаптира към промени в данните и запазва своята прецизност.
- Да се осигури стабилност и устойчивост на системата чрез функционалности като автоматични прекъсвачи (circuit breakers) към базата данни, таймаути на заявките и тестване чрез инжектиране на грешки (fault injection).
- Да се улесни внедряването и мащабирането на софтуерната част чрез Docker и Gunicorn.

**Основни етапи в реализиране на проекта:**

- **Проектиране и изработка на хардуера:** Избор на компоненти, дизайн на схемата, сглобяване на прототипа и проектиране на 3D модел за кутия.
- **Проектиране на системата:** Дефиниране на софтуерната архитектура, схемата на базата данни, API договора и механизма за предаване на данни от хардуера към API.

- **Разработка на Backend:** Имплементиране на приложението с FastAPI, включително API рутери, сервизен слой и интеграция с базата данни чрез SQLAlchemy и Alembic за миграции.
- **Интеграция на ML модели:** Създаване на регистър за модели (Model Registry) за зареждане и обслужване на различни версии на моделите. Имплементиране на логиката за предсказване.
- **MLOps функционалности:** Изграждане на модули за мониторинг (`/metrics`), детекция на дрейф, оценка на модели и тестове за устойчивост.
- **Тестване:** Написване на цялостни unit и integration тестове за осигуряване на качеството на кода и производителността.
- **Контейнеризация и внедряване:** Създаване на Dockerfile и `docker-compose` конфигурация за лесно и възпроизводимо внедряване.

**Ниво на сложност на проекта:**

- **Оптимизация на позиционирането на сензорите:** Разрешен е фундаментален конфликт между двата основни сензора. Радарният сензор изисква пряка видимост и разстояние (1-1.5м), докато акселерометърът изисква физически контакт с леглото за отчитане на вибрации. Решението се базира на анализ на компромиси, като се препоръчва странично на рамката на леглото или монтаж под матрака (за матраци без метални пружини. Този подход балансира между качеството на сигнала от радара и чувствителността на акселерометъра.
- **Гарантирана производителност в реално време:** Системата е проектирана да отговаря на ключов показател за ефективност (KPI) от "**p95 времезакъснение < 350ms"** при инференция. Това гарантира, че данните могат да бъдат анализирани почти мигновено, което е критично за бъдещи приложения като аларми или нотификации.
- **Надеждност на потока данни и офлайн буфериране:** Осигурена е непрекъснатост на данните чрез интелигентен механизъм за предаване. Хардуерът приоритизира изпращането към API и само при липса на връзка активира резервен механизъм - запис върху **"8MB вградена флаш памет"**. Този капацитет служи

като ключов показател (KPI) за автономност, позволявайки съхранение на данни за до 189 дни без връзка.

- **Безопасно презареждане на ML модели (Zero-Downtime):** Имплементиран е `thread-safe` механизъм за актуализация на моделите в реално време. Това е критично за 24/7 мониторинг, тъй като позволява на системата да се обновява с нови, по-точни модели, без да се прекъсва обслужването на заявки и без да се губят данни от анализ.
- **Мащабируемо управление на времеви редове данни:** Ефективното управление на големи обеми от данни (епохи) в TimescaleDB е решено чрез използването на автоматични политики за компресия и задържане (retention), както и специализирани хипертаблици, които осигуряват висока производителност при запис и четене.
- **Сигурност:** Проектиране и налагане на стабилен модел за многопотребителска автентикация и оторизация чрез JWT.

**Логическо и функционално описание на решението:**

**Архитектура**

Системата Noctis се състои от два основни компонента: **“хардуерно устройство за мониторинг”** и “**софтуерен backend”**.

**1. Хардуерен компонент:** Физическо устройство, което събира данни по време на сън (чрез ESP32 микроконтролер, 60GHz mmWave радар за жизнени показатели и MPU6050 акселерометър за движение). То “**първоначално се опитва да изпрати данните към софтуерния API”**. При невъзможност за връзка, данните временно се съхраняват на вградена 8MB флаш памет, за да бъдат изпратени по-късно.

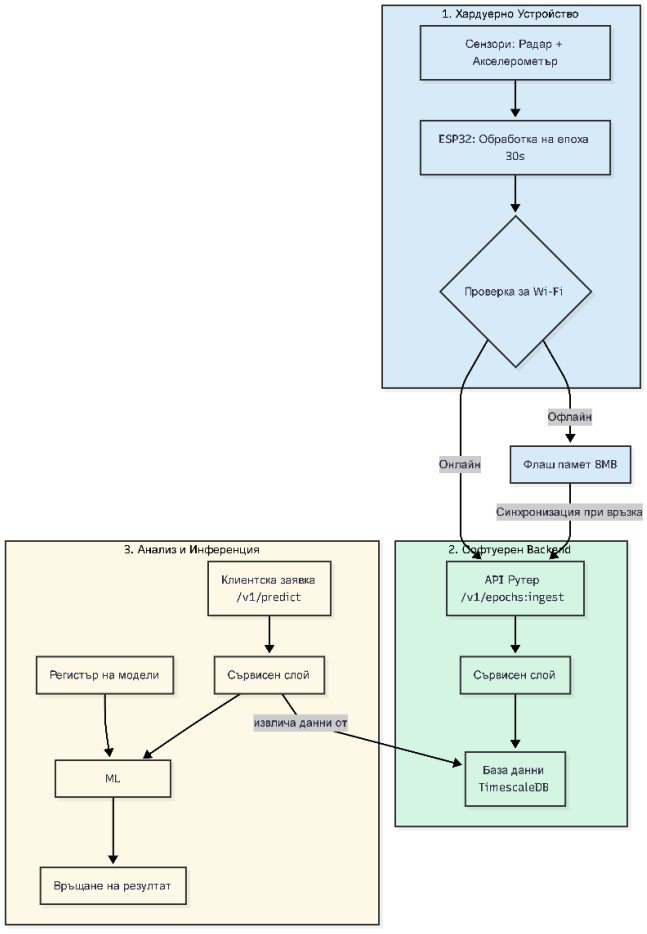
**2. Софтуерен компонент (Backend):** Backend API услуга, която приема данните от хардуера (директно или от флаш паметта), извършва анализ и предсказване. Взаимодействието се осъществява програмно чрез REST API извиквания. Архитектурата на софтуера се състои от няколко ключови компонента:

- **FastAPI рутери:** Обработват входящите HTTP заявки и ги насочват към съответните услуги.

- **Сервизен слой:** Съдържа основната бизнес логика за функционалности като предсказване и анализ на дрейф.
- **SQLAlchemy + TimescaleDB:** Осигурява слоя за съхранение на данни.
- **Регистър на модели (Model Registry):** Файлово-базиран регистър, който съхранява и управлява различни версии на ML моделите. Това служи като ключов елемент за надеждност, позволявайки възпроизводимост на резултатите, лесен `rollback` към предишни версии при нужда и гарантира, че системата винаги използва одобрен и тестван модел.

**Поток на данните (Data Flow)**

Процесът на движение на данните е проектиран за максимална надеждност:




1. **Събиране:** На всеки 30 секунди, ESP32 микроконтролерът събира данни от радарния сензор и акселерометъра.

2. **Обработка:** Суровите данни се обработват в 15-мерен вектор (епоха), готов за анализ.

3. **Опит за предаване:** Устройството се опитва да изпрати новосъздадената епоха към `/v1/epochs:ingest` ендпойнта на FastAPI бекенда.

4. **Резервен запис (Fallback):** Ако API е недостъпен (липса на Wi-Fi), епохата се записва в опашка на вградената 8MB флаш памет.

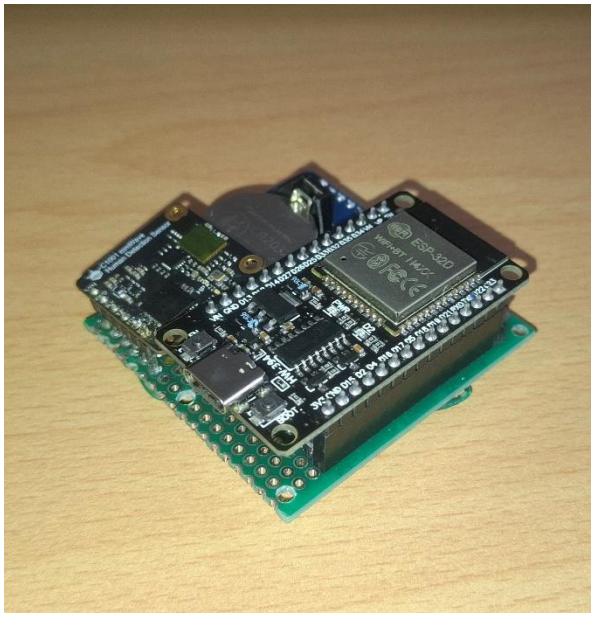
5. **Синхронизация:** При възстановяване на връзката, устройството автоматично изпраща натрупаните в паметта епохи към API, за да се гарантира, че няма загуба на данни.

6. **Инференция и съхранение:** След като данните постъпят в бекенда, те се съхраняват в TimescaleDB и могат да бъдат използвани за инференция чрез `/v1/predict` ендпойнта.

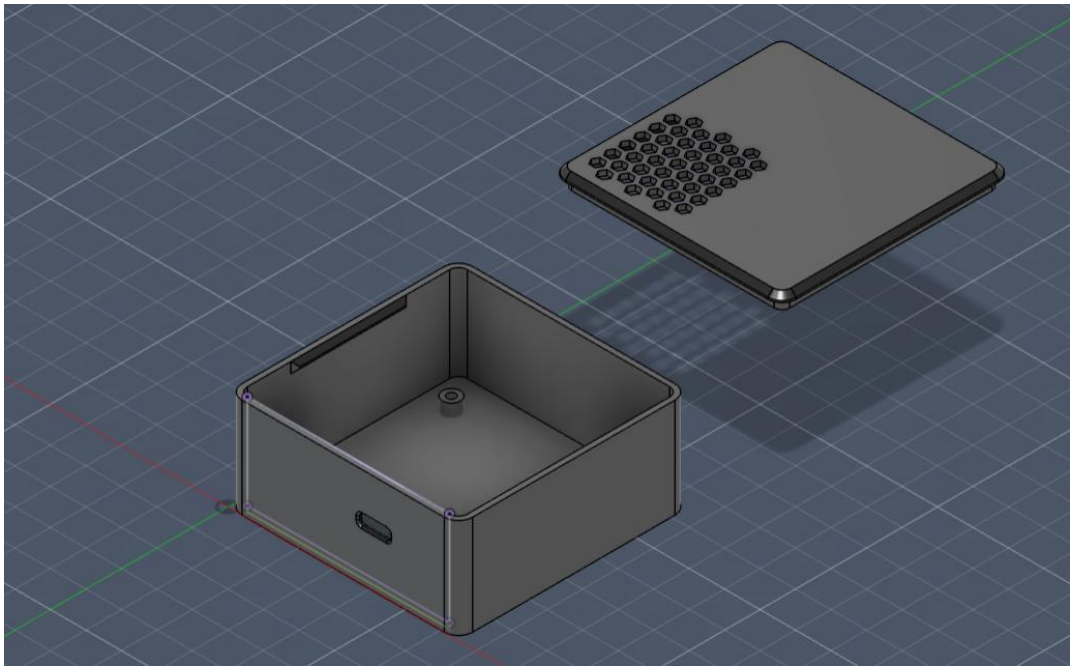
**Хардуер: Устройство за Мониторинг на Съня Noctis**

Източникът на данни за системата е специално създадено хардуерно устройство, което следи съня на потребителя.

- **Основни компоненти:** Устройството е базирано на микроконтролер ESP32 и използва 60GHz милиметров радарен сензор (за отчитане на сърдечен ритъм, дишане и присъствие) и акселерометър MPU6050 (за проследяване на движения и вибрации).
- **Обработка на данни:** Микроконтролерът ESP32 обработва суровите данни от сензорите в 30-секундни епохи, всяка от които съдържа вектор от 15 характеристики.
- **Предаване и съхранение на данни:** Устройството е проектирано да изпраща събраните данни към софтуерния API при наличие на мрежова свързаност. В случай на невъзможност за връзка, данните временно се съхраняват на вградена 8MB флаш памет, за да бъдат изпратени по-късно.

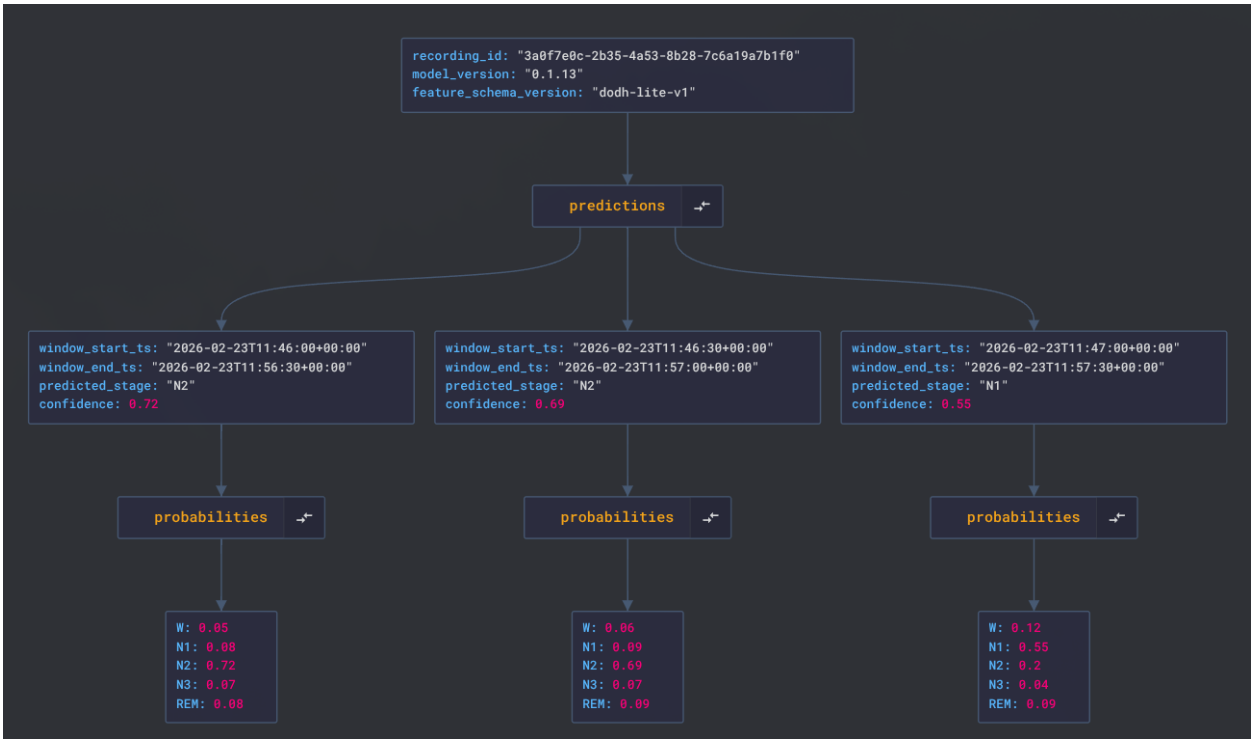


- **Физически дизайн:** Компонентите са поместени в персонализирана кутия, отпечатана на 3D принтер, чиито дизайн файлове са налични в проекта.



**API Ендпойнти (вместо страници)**

- **`/healthz`, `/readyz`, `/metrics`:** Ендпойнти без автентикация за здравни проверки и метрики за Prometheus.
- **`/v1/epochs:ingest`:** Ендпойнт за въвеждане на нови данни за епохи в системата.
- **`/v1/predict`:** Основният ендпойнт за получаване на предсказания за фази на съня от активния модел.
- **`/v1/model/drift`:** Ендпойнт, който служи като система за ранно предупреждение. Той предоставя отчет за текущия дрейф на модела, позволявайки проактивна намеса и преобучение, преди точността на предсказанията да се е влошила значително.
- **`/internal/...`:** Набор от ендпойнти за вътрешни операции като стартиране на стрес тестове, управление на политиките на базата данни и инжектиране на грешки за тестване на устойчивостта.




**Реализация:**

**Използвани технологии:**

- **Backend:** Python 3.11+, FastAPI
- **Уеб сървър:** Uvicorn, Gunicorn
- **База данни:** TimescaleDB (разширение за PostgreSQL)
- **Достъп до данни:** SQLAlchemy (ORM), Alembic (миграции)
- **Машинно обучение:** NumPy, Scikit-learn
- **Контейнеризация:** Docker, Docker Compose
- **Тестване:** Pytest

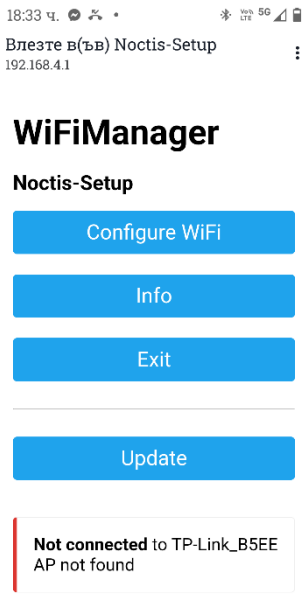
**Описание на приложението:**

- Достъпът до приложението се осъществява през уеб сървър (напр. `http://localhost:8000`, когато се изпълнява локално). Взаимодействието се извършва чрез API извиквания от клиентско приложение или с инструменти като `curl`.
- Основният работен процес включва:

- Автентикация за получаване на JWT токен.
- Въвеждане на данни за записи и епохи чрез ендпойнтите `/v1/devices`, `/v1/recordings`, и `/v1/epochs:ingest`.
- Заявяване на предсказания за въведените данни чрез ендпойнта `/v1/predict`.
- Мониторинг на производителността и дрейфа на модела чрез ендпойнтите `/v1/model/drift` и `/metrics`.

Как се използва приложението:

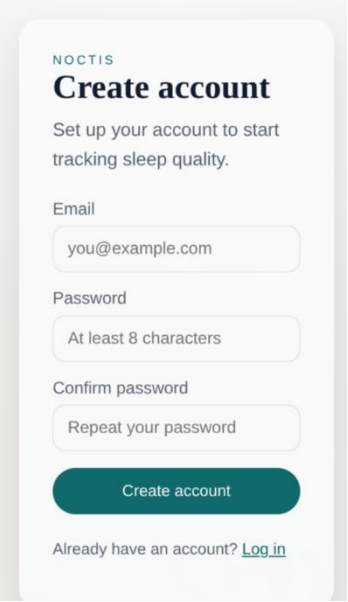
1. Свързване на устройството към Wi-Fi:




Първата стъпка е свързването на хардуерното устройство към домашната Wi-Fi мрежа. Този процес се извършва еднократно при първоначалната настройка.

- Захранване: Потребителят включва устройството към захранване.
- Точка за достъп (Hotspot): Устройството автоматично създава временна Wi-Fi мрежа (hotspot) с име (SSID) от типа Noctis-XXXX.
- Свързване към Hotspot: Потребителят използва своя смартфон или компютър, за да се свърже към тази мрежа.
- Конфигурационен портал: След свързване, автоматично се отваря уеб страница (captive portal), която сканира за налични Wi-Fi мрежи.
- Въвеждане на парола: Потребителят избира своята домашна Wi-Fi мрежа от списъка и въвежда паролата за нея.
- Запазване и рестартиране: Устройството запазва конфигурацията, рестартира се и се свързва автоматично към избраната мрежа. Временният hotspot се деактивира.

2. Създаване на потребителски акаунт:

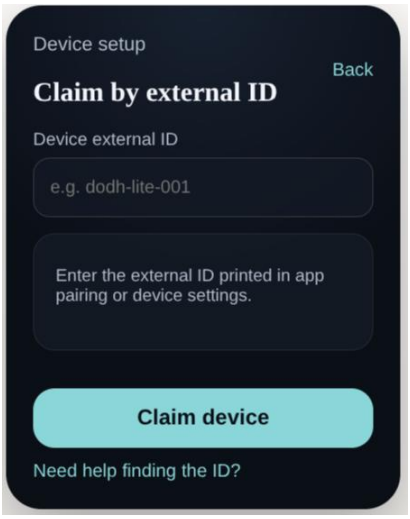



След като устройството е свързано към интернет, потребителят трябва да си създаде акаунт в уеб приложението на Noctis.

- Регистрация: Потребителят отваря уебсайта на Noctis и избира опцията за регистрация ("Register" или "Sign Up").
- Попълване на данни: Въвежда необходимата информация като име, имейл адрес и парола.

- Потвърждение: Системата изпраща имейл за потвърждение на посочения адрес. Потребителят трябва да последва линка в имейла, за да активира своя акаунт.

3. Свързване на устройството с акаунта на човек:




След като се влезе в акаунта си човек въвежда уникалният номера на устройството, което се намира отдолу на кутията.

4. Използване на системата:

С активен акаунт и свързано устройство, потребителят може да използва пълната функционалност на системата.

- Вход в системата: Потребителят влиза в своя акаунт с имейл и парола.
- Визуализация на данни: На главния екран (dashboard) се показват обобщени данни от последния сън – времетраене, разпределение на фазите на съня (лек,
- дълбок, REM) и други показатели.
- История и тенденции: Потребителят има достъп до исторически данни, които може да филтрира по дати, за да проследява тенденции в качеството на своя сън.

- Управление на устройство: В своя профил потребителят може да премахне или смени своето устройство.

**Заключение**

- Проектът предоставя стабилно и мащабируемо backend решение за обслужване на модели за предсказване на фази на съня. Той включва изчерпателен набор от MLOps функционалности, които са от решаващо значение за поддържане на качеството и производителността на модела в производствена среда.

**Бъдеще на проекта:**

- Разширяване на регистъра на моделите, за да поддържа по-голямо разнообразие от архитектури на модели.
- Добавяне на поддръжка за приемане на данни в реално време (напр. чрез WebSockets или Kafka).
- Разработване на по-усъвършенствани функции за детекция на дрейф и обяснимост (explainability).