**ДОКУМЕНТАЦИЯ на проект № 426**

1. **ТЕМА:** AutoGuard
2. **АВТОР:**

Тимон Стилиянов Трифонов

* ЕГН: 11.01.2006
* Адрес: гр. Враца,
* E-mail: [timon\_trifonov@abv.bg](mailto:timon_trifonov@abv.bg)
* Тел: +359876872050
* Училище: ППМГ „Акад. Иван Ценов“, гр. Враца
* Клас: XІ

1. **РЪКОВОДИТЕЛИ:**

Станислава Светославова Каменова

* Тел. 0879003288
* E-mail: [kamenova@bitex.bg](mailto:kamenova@bitex.bg)
* Старши учител по информатика и информационни технологии

Албена Кирилова Цекова

* Тел: 0879003176
* E-mail: [albena2008@abv.bg](mailto:albena2008@abv.bg)
* Старши учител по информатика и информационни технологии

**4. РЕЗЮМЕ:**

**4.1. Цел**

Приложение, представляващо цялостна система за откриване на аномалии, специално пригодена за автомобилната индустрия. Използвайки усъвършенствани техники за машинно обучение, включително дълбоко обучение и статистическо моделиране, системата анализира данни от различни сензори в двигателя на колата. Чрез този богат набор от данни, системата може да открива аномалии в реално или почти реално време, позволявайки активна поддръжка, предотвратяване на грешки и подобрени мерки за безопасност.

**4.2. Етапи в реализирането на проекта**

* Избор на тема и конкретна формулировка
* Преглед на съществуващи приложения
* Подбор на информация
* Създаване на подходящ дизайн
* Подбор на методи и средства за реализиране на идеята
* Изработване на авторски компоненти и материали
* Реализация
* Тестване на проекта
* Изработване на рекламни материали
* Представяне на проекта пред публика

**4.3. Логическо и функционално описание на решението**

С нарастващата сложност на съвременните превозни средства и взаимосвързаността на техните системи, се появява необходимост от стабилни решения за откриване на аномалии. Тествани са различни алгоритмични решения на проблема, като в отговор на това търсене, е представен модел базиран на статистически алгоритъм. Приложението позволява на потребители без дори базови познания по машинно обучение да тренират и прилагат в реално време модела. При изполване на приложението потребителя най-напред се свързва с OBD (On-board diagnostics) устройство свързано към автомобила. То предава информация за двигателя към приложението чрез bluetooth. След това се създава модел или се избира по-рано създаден, който трябва да бъде трениран. След като приключи тренирането, моделът и неговите тегла се запаметяват на сървъра и той може да бъде избран за работа. В режим на работа (Inference mode) моделът връща процента аномалия при всяка извадка (sample) в реално време. Потребителя може да настройва големината на модела, кои параметри да използва при тренирането и праговата му стойност. Ако процента аномалия е по-голям от зададената прагова стойност, приложението записва аномалията и информация за нея в специален списък. Потребителя разглежда списъка и си прави изводи върху неговото съдържание, ако грешката е отстранена, то тя може да бъде изтрита.

**4.4. Реализация и ниво на сложност на проекта**

**Връзка с автомобила**

За връзка с прикаченото OBD устройството, приложението използва flutter\_bluetooth\_serial модул, който позволява лесна комуникация дори и с устройствата поддържащи само по-стари версии на Bluetooth, като v2, v2.1 и v3.

Правилата на комуникация между устройството и приложението се определят от стандартът **SAE J1979**. Това е единен протокол, който дефинира методите за получаване на диагностични данни и списъка от стандартни параметри, които могат да бъдат изисквани от устройството. Всеки параметър се адресира чрез собствен идентификационен номер (PID), дефиниран в SAE J1979 протокола. За получаване на информация за конкретни параметър, приложението изпраща съответните идентификационни номера към OBD устройството. Това е така наречената **PID** заявка. След това устройството приема заявката и изпраща обратно желаната информация. След като бъде изпратена информация към приложението, тя се декодира от шестнайсетична бройна систем и към получения резултат се прилага определена формула, която превръща полученото число в желаната мерна единица. Пълният списък с PID заявки и съответните формули може да намерите [тук](https://en.wikipedia.org/wiki/OBD-II_PIDs).

**Пример:**

Искаме да разберем моментната стойност на температурата на двигателя. Изпращаме идентификационния номер на параметъра, който в нашия случай е „05“. Хедърът „01“ пред идентификационния номер сигнализира, че това е заявка.

****

Получаваме низът:

****

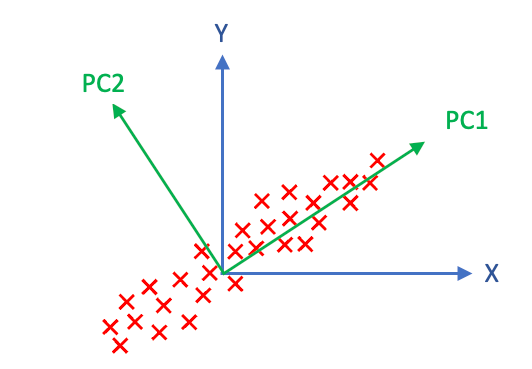
Хедърът „4105“ сигнализира, че това е отговор на заявка с идентификационен номер „05“. „C9” e шестнаисетичното число отговарящо на температурата в двигателя. В десетична бройна система, това е числото 201. Прилагаме съответната формула отSAE J1979 протокола:

****

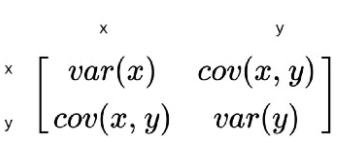
Тук за за превръщане в градус Целзий тя е А-40. Пресмятаме 201 – 40 и намираме температура от 161°C.

**Откриване на аномалии**

Алгоритъмът за откриване на аномалии е основно базиран на метода на главните елементи (МГЕ). Методът на главните елементи е предимно техника за намаляване на размерността. Работи чрез идентифициране на основните компоненти на генералната съвкупност . Основните компоненти са независими вектори на признаци, наричани също собствени вектори на дадени данни, които обясняват максималната дисперсия в данните. Всеки основен компонент е линейна комбинация от съществуващи корелирани характеристики и е ортогонален на други собствени вектори. С помощта на МГЕ можем да намалим броя на векторите на характеристиките, без да губим информационна стойност. Откриването на аномалия разчита на грешка при реконструкция. След като основните компоненти бъдат идентифицирани, чрез избиране на всички основни компоненти можем да реконструираме оригиналните данни от трансформираните данни без загуба. По същия начин, като избираме само основни компоненти, които обясняват по-голямата част от дисперсията, трябва да можем да пресъздадем приближение на оригиналните данни. Грешката, генерирана по време на реконструкция на оригиналните данни, се нарича грешка при реконструкция. За аномалии в данните грешката при реконструкцията е **висока**. За алгоритъмът са използвани Python и библиотеките PyOD, Numpy, Scikit-learn.



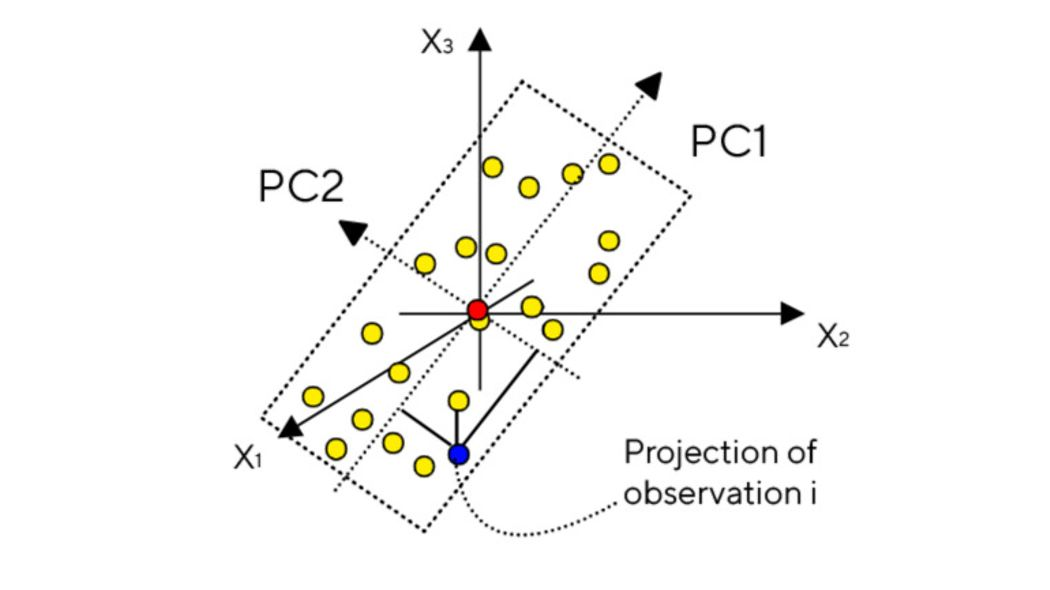
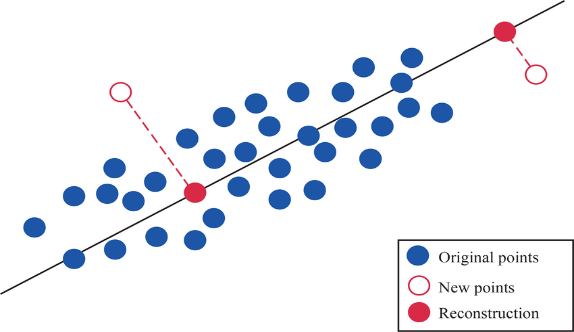
На изображението е представен примерен случай, където броят на характеристики е равен на две. За изчисляване на посоката на разпределението на данните(вектор PC1) се взима ковариационната матрица на извадката. Тук тя е :

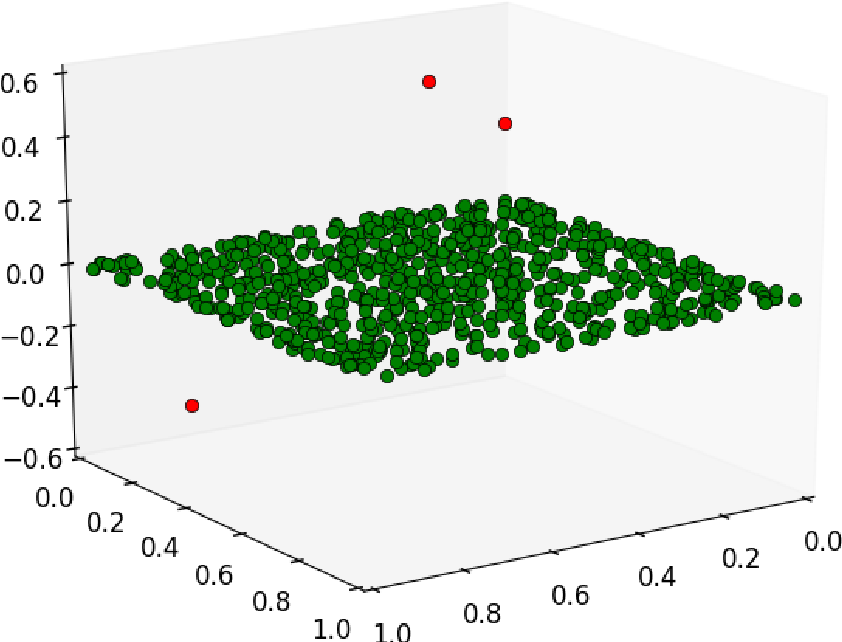


Собствените вектори на матрицата са всъщност самите основни компоненти PC1 и PC2. Точките се проектират последователно върху PC1 и PC2, като за всеки основен компонент се изчислява средната дължина на точките проектирани върху него считана от от центъра (фиг.1).

Тази дължина се нарича още собствена стойност на основния компонент. Отношението между собствената стойност на даден компонент и сбора на собствените стойности на всички основни компоненти дава представа за това как един компонент или един параметър, влияе на общото разпределение на информацията. Избираме оптимално количество основни компоненти така, че по-голямата част от разпределението на данните да може да бъде обяснено чрез тях и реконструираме първоначалните стойности. Грешката при реконструкцията определя процента на аномалията.(фиг.3) При случаи с по-голяма размерност на данните, вместо върху прави, точките се проектират върху хиперравнини.(фиг.2)

фиг.(1)

фиг.(2)



фиг.(3) Пример с 3 параметъра.

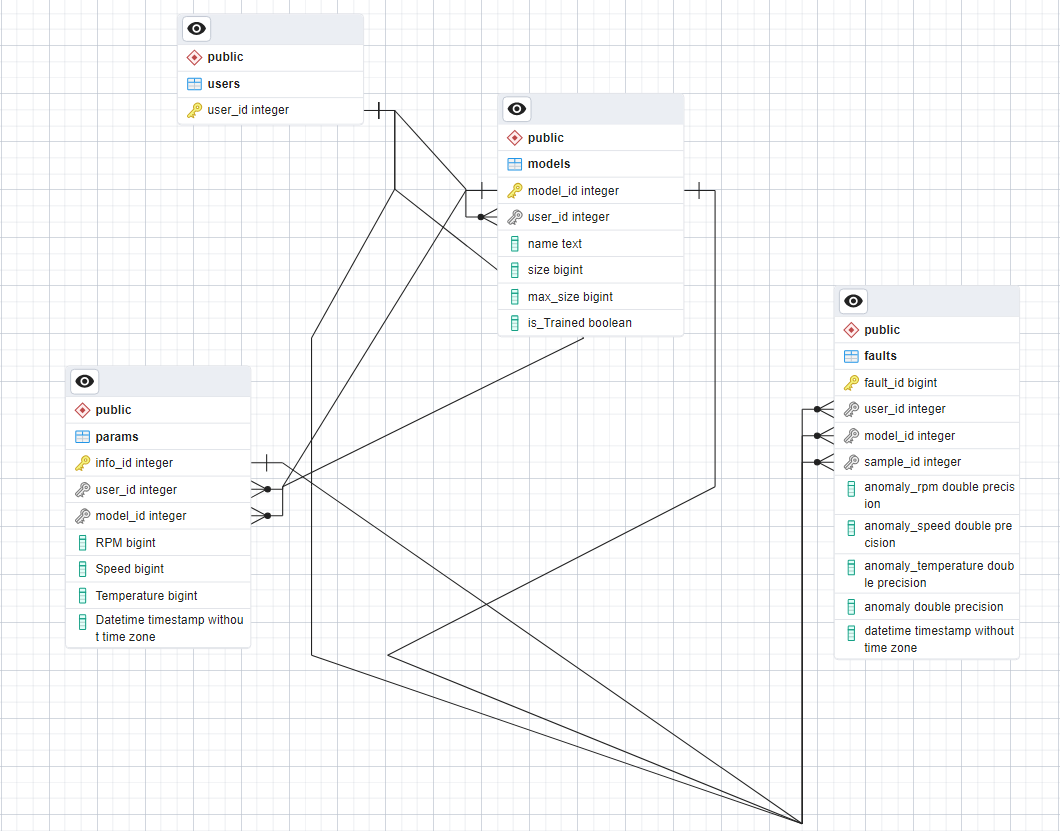
Червените точки са аномалии, тъй като не се намират в една равнина с множеството зелени след последователно прилагане на МГЕ за редукция на размерността и реконструкция. Това е така, защото те немогат да бъдат обяснени само с 2 основни компонента, т.е. в двуизмерно пространство, за разлика от множеството зелени точки.

**База данни**

Базата данни е релационна и се състои от 4 таблици. В първата таблица се намира информация за потребителя, във втората събраните сензорни данни от автомобила, във третата- информация за съществуващи модели, създадени от потребителя, а във четвъртата се събира информация за всички върнати аномалии от модела. Разбира се и четирите са свързани по между си, посредством уникални и външни ключове. Базата данни е имплементирана чрез Postgressql. Връзката между приложението и базата данни се осъществява, чрез postgres.dart конектор модула.

Поради прекалено честото изпращане на информация от сензорните данни към втората таблица, тя е конвертирана в TimescaleDB хипертаблица. Timescale е разширение на Postgres, специализирано в управление на серийна информация. То позволява вмъкването на редове в таблицата да става през много по-малък интервал от време. Имплементирането на TimescaleDB в базата данни, значително подобри работата на приложението като ускори процеса по събиране на информация и предотврати периодичната загуба на данни.

(фиг. 1) Опростен изглед на базата данни.



**Интерфейс**

Интерфейсът на мобилното приложение се състои от 3 раздела. (фиг. 4) Първият раздел е начална страница с три бутона. Горе в дясно се намира бутон, който отваря менюто с настройки. То съдържа различни опции като свързване със bluetooth устройство, нотификации, промяна на език, помощ, но основната му функция е да дава на потребителя пълна свобода при избор на параметри, които модела да следи, неговата големина и праговата стойност, която ако бъде надвишена модела връща аномалии. В центъра на раздела са разположени 2 бутона. Първият води до менюто за събиране на данни и създаване и трениране на модели, втория бутон стартира така наречения Inference mode, който използва вече готов модел за работа в реално време. Втория раздел съдържа таблица с данните от сензорите на автомобила, както и графика отчитаща настоящия процент на аномалия. Третия раздел съдържа списък с всички засечени аномалии, подредени по дата и час, както и основна информация за тях. Долу в ляво са разположени 3 индикатора, 1 и 2 отчитат съответно връзката с интернет и сървъра, а 3 проверява съвместимостта на свързаното устройство и автомобила с приложението.

**Използвани технологии и модули :**

• Flutter, Dart, Bluetooth-serial - за разработка на интерфейса на мобилното приложение

• PostgresSQL - за базата данни

• Python, Sklearn, Numpy, Pyod, PythonOBD - за разработка на сървъра, модела за аномалии и тестване

**Тестване**

В по-голямата си част, приложението е тествано, чрез PythonOBD модула, който симулира сензорни данни от автомобил и ги изпраща на приложението, чрез Bluetooth.

Проведени са и тестове в реални условия на автомобили Opel Meriva B и Mazda CX3 с прикачено ELM327 - OBD2 устройство. Приложението се справя според очакванията, като единствените забележки са по-бавното събиране на данни от автомобила и периодичното подаване на невалидни данни от него.

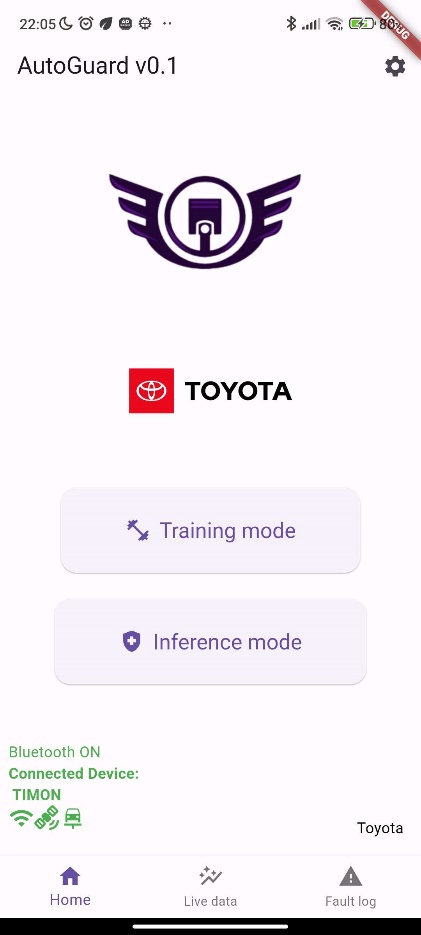
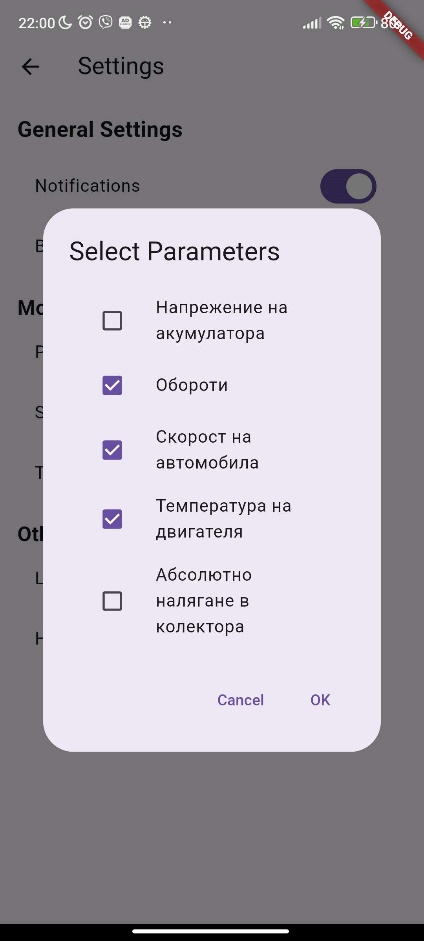
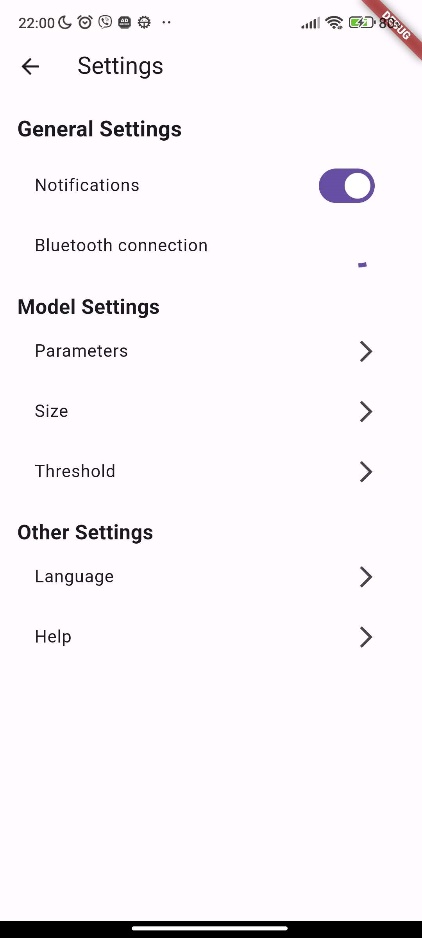
Трябва да се отбележи, че макар и приложението да се справя с повечето изкуствено генерирани аномалии, моделът е възпрепятстван при идентифициране на някои аномалии от реалния живот породени от процеси в двигателя с нелинеен характер, както и грешки простиращи се в голям интервал от време. Това е недостатък породен от характерната за алгоритъма линейност.

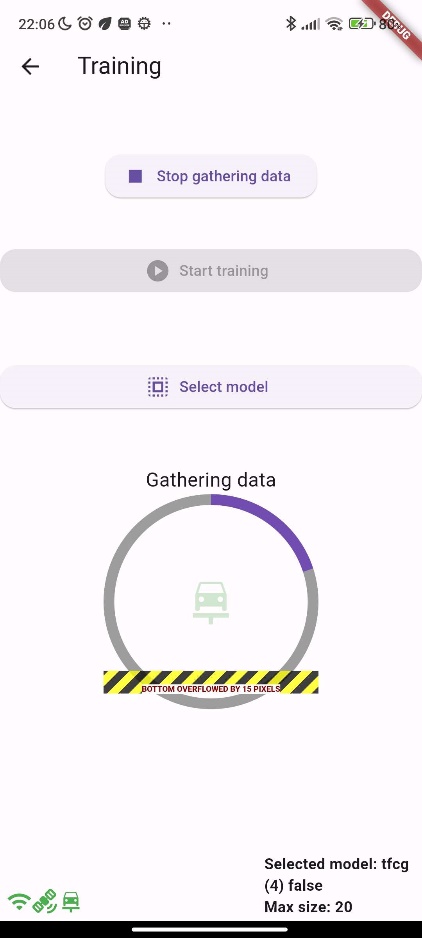
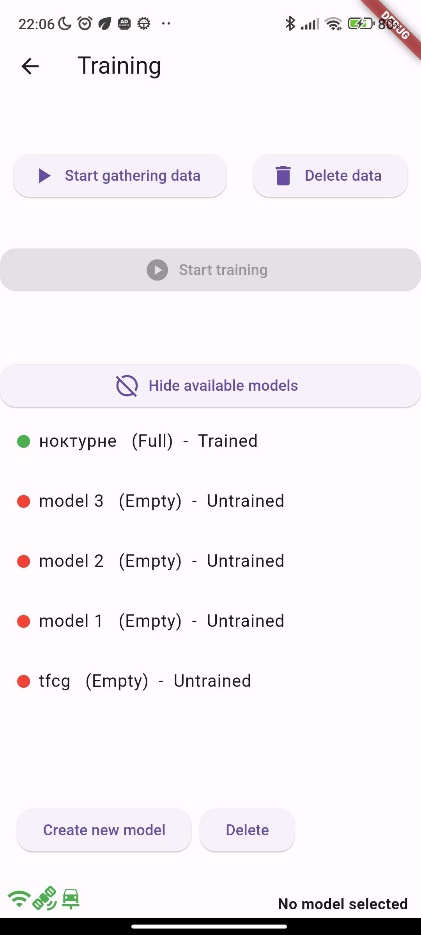
**4.5. Използвани източници**

* https://stackoverflow.com/
* <https://openai.com/blog/chatgpt/>
* Документация на Python, Sklearn, Numpy, Pyod, PythonOBD

**4.6. Изображения от проекта**

**(фиг. 4)**





**4.7. Описание**

Приложението може да бъде разделено на няколко основни части. Първата част осъществява връзката между приложението и OBD устройството. Тя събира и изпраща информация към OBD, като преди това провери за неговата достоверност и кодира (при изпращане) или декодира (при получаване) низа. Втората част представя сензорната информация на потребителя чрез таблици и диаграми и я записва в база данни за потенциално трениране на модел. Третата част използва събраната информация за направата на завършен модел, който да бъде използван при откриването на аномалии в двигателя на колата. Четвъртата и последна част записва всяка регистрирана аномалия и информация свързана с нея в списък, който се предоставя на потребителя.

Приложението ще бъде достъпно на адрес: <http://it.ppmg-vratsa.com/appgallery/>

**4.8. Заключение**

AutoGuard е приложение, което се справя успешно в задачата си да открива основни грешки в работата на двигателя на автомобила. То имплементира и разширява МГЕ алгоритъмът, като го пригодява за откриване на аномалии. Поради линейната същност на МГЕ, приложението не се справя добре при откриване на по-трудно забележими аномалии като някои контекстуални грешки, както и аномалии свързани с нелинейни процеси в работата на двигателя. Това е предпоставка за допълнителни изследвания и разширения по алгоритъма, като например kPCA – МГЕ алгоритъм, който използва кернел методи, придаващи нелинейност на модела.