**Документация за Фитнес Приложение**

Никола Методиев

Цели на проекта

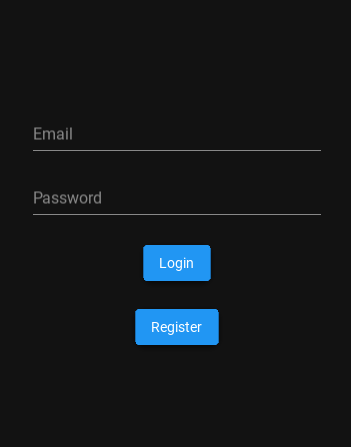
Фитнес приложението е мобилно приложение, изградено с използването на KivyMD. То предоставя функционалности, свързани с управлението на тренировъчни програми, проследяването на упражнения и анализ на данни относно фитнес активността на потребителите. Проекта е насочено към хора, които търсят приложение опростен дизайн за помощник по време на тренировка, помагайки с обяснение и последни резултати от дадено упраженние.

Структура на Проекта

Проектът е организиран в различни компоненти, включително екрани, връзка с база данни, модел на потребителя и потребителски карти и оформления.:

1. Screens:

1.1 `LoginScreen`, `RegisterScreen`: Обработва потребителско влизане с имейл и парола.



`*LoginScreen*`:

Атрибути:

db: База данни.

main\_screen: Основен екран.

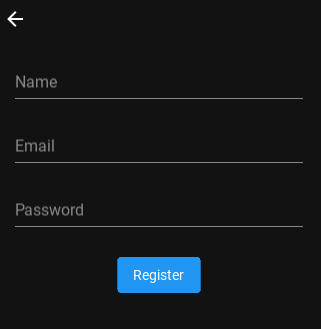
Методи:

\_\_init\_\_(db, main\_screen, \*\*kwargs): Инициализира обекта.

on\_login(\*args): Обработва събитието при натискане на бутона за вход.

show\_login\_popup(message): Показва изскачащ прозорец със съобщение за входа.

on\_register\_button(\*args): Превключва към екрана за регистрация.



`*RegisterScreen*`:

Атрибути:

db: База данни.

main\_screen: Основен екран.

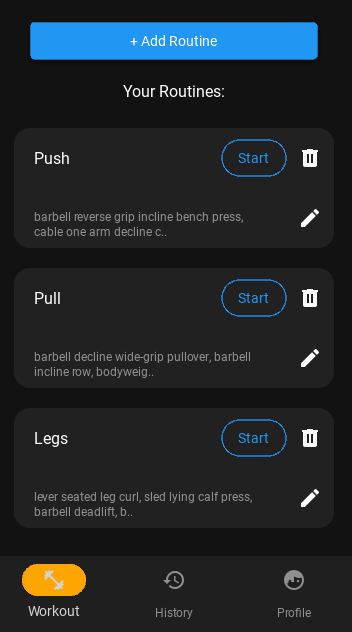
Методи:

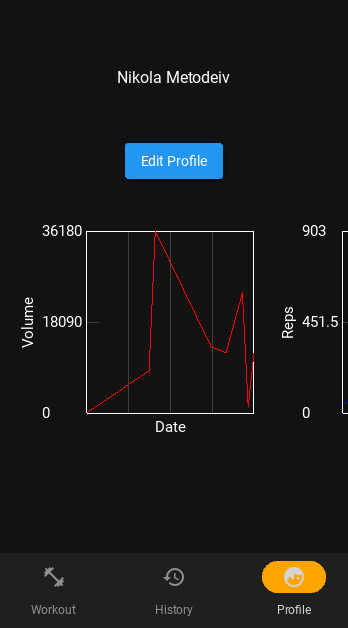
\_\_init\_\_(db, main\_screen, \*\*kwargs): Инициализира обекта.

on\_register(\*args): Обработва събитието при натискане на бутона за регистрация.

show\_register\_popup(message): Показва изскачащ прозорец със съобщение за регистрацията.

1.2 `MainScreen`: Основният екран, съдържащ лента за навигация в долната част с две раздела - "Тренировка", “История“ и "Профил".





Атрибути:

db: База данни.

exercises: Упражнения.

user: Потребител.

loaded: Флаг за зареждане.

Методи:

\_\_init\_\_(db, exercises, \*\*kwargs): Инициализира обекта.

update\_statistics\_graph(workouts): Актуализира графиките за статистика на тренировките.

update\_size(instance, value): Актуализира размерите на графиките спрямо размера на екрана.

update\_user\_data(user): Актуализира данните за потребителя и зарежда упражнения, рутини и минали тренировки.

edit\_profile(): Отваря диалог за редактиране на профила.

load\_user\_exercises(): Зарежда упражненията на потребителя от рутини и минали тренировки.

on\_add\_routine\_button\_press(): Обработва натискането на бутона за добавяне на нова рутина.

create\_routine\_card(routine): Създава карта за рутина и я добавя към изгледа.

start\_workout(routine): Започва тренировка за избраната рутина.

add\_routine\_to\_user(routine): Добавя рутина към профила на потребителя.

delete\_routine\_card(routine): Изтрива карта за рутина и актуализира профила на потребителя.

edit\_routine\_card(routine): Редактира карта за рутина.

RoutineScreen(): Показва рутините на потребителя на основния екран.

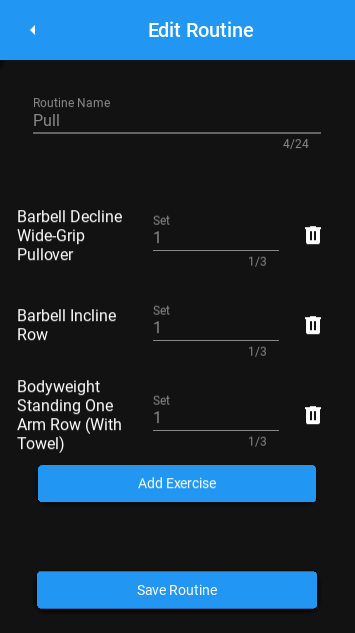
add\_user\_workout\_cards(workouts): Добавя карти за минали тренировки на потребителя на основния екран.

delete\_workout\_card(workout): Изтрива карта за минала тренировка и актуализира профила на потребителя.

PastWorkoutsScreen(): Показва минали тренировки и статистика на основния екран.

1.3 `EditRoutineScreen`: Екран за редактиране и запазване на тренировъчни програми.

Атрибути:

 db: Инстанция на базата данни.

edit\_state (bool): Флаг, показващ дали екранът е в режим на редакция.

routine\_to\_edit: Рутина за редакция при включен режим на редакция.

selected\_exercises (list): Списък от избрани упражнения за рутината.

exercises: Всички налични упражнения от базата данни.

body\_parts: Различни части на тялото от наличните упражнения.

routine\_name\_textfield: MDTextField за въвеждане на името на рутината.

Методи:

\_\_init\_\_(self, db, Window=None, \*\*kwargs): Инициализира EditRoutineScreen.

on\_pre\_enter(self, \*args): Метод, извикан преди влизането на екрана. Актуализира показваните упражнения и бутони.

delete\_exercise(self, exercise, card): Изтрива избраното упражнение от рутината.

show\_body\_part\_selection(self, \*args): Показва падащо меню за избор на упражнения въз основа на части на тялото.

menu\_callback(self, body\_part, \*args): Callback функция за падащото меню. Зарежда упражнения въз основа на избраната част на тялото.

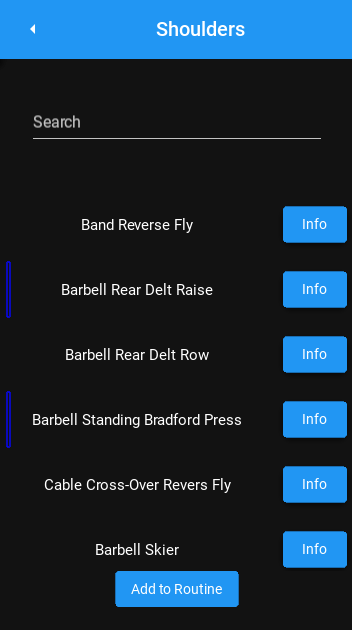
back\_button(self): Навигира обратно към основния екран.

on\_add\_routine\_button\_press(self): Обработва натискането на бутона за добавяне или актуализация на рутина.

reset(self): Нулира екрана до началното му състояние.

1.4 `BodyPartExercisesScreen`: Показва упражнения на база части от тялото.

Атрибути:

 exercises (list): Списък с упражнения за показване.

selected\_exercises (list): Списък за съхранение на избрани упражнения.

search\_text (str): Текст, използван за филтриране на упражненията.

Методи:

on\_pre\_enter(self, \*args): Метод, извикан преди влизането на екрана. Изчиства съществуващите виджети и добавя филтрирани упражнения.

add\_filtered\_exercises(self): Добавя филтрирани упражнения към изгледа въз основа на текста за търсене.

open\_exercise(self, exercise, button\_instance): Навигира към екрана с информация за упражнението за избраното упражнение.

on\_label\_touch(self, exercise): Обработва събитието на докосване на ClickableTwoLineListItem за избор или деселекция на упражнение.

on\_search\_text\_change(self, instance, value): Актуализира текста за търсене и обновява изгледа.

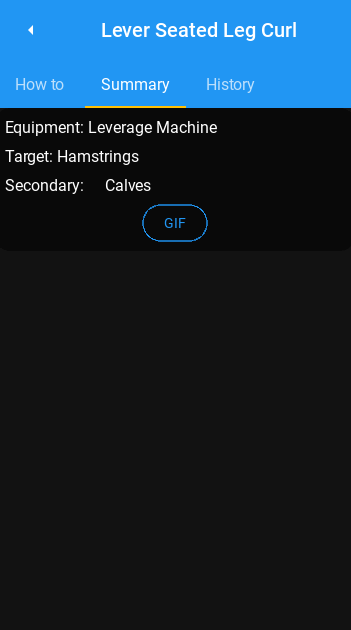
on\_search\_text\_focus(self, instance, value): Нулира изгледа, за да покаже всички упражнения, когато текстът за търсене е празен.

back\_button(self): Навигира обратно към "edit\_routine\_screen."

on\_add(self): Добавя избраните упражнения към "edit\_routine\_screen" и навигира обратно към него.

Изчиства съществуващите виджети и задава scroll\_y обратно на 1 при изчистване на виджетите.

1.5 `ExercisesScreen`: Съдържа горен панел и раздели за организация на упражненията.

Атрибути:

db: Инстанция на базата данни.

exercise (dict): Информация за текущото упражнение.

summary\_layout: MDBoxLayout за съдържанието на раздела "Summary".

history\_layout: MDBoxLayout за съдържанието на раздела "History".

how\_to\_layout: MDBoxLayout за съдържанието на раздела "How to".

last\_screen (str): Име на последния посетен екран.

Методи:

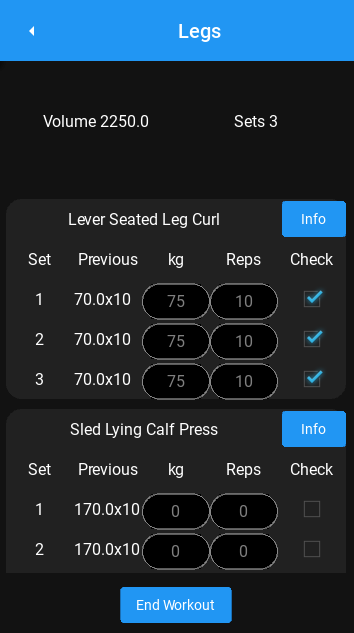
\_\_init\_\_(self, db, \*\*kwargs): Инициализира екрана за упражнения.

on\_tab\_switch(self, instance\_tabs, instance\_tab, instance\_tab\_label, tab\_text): Обработва събитието на смяна на раздел и актуализира съдържанието на раздела съответно.

on\_pre\_enter(self): Метод, извикан преди влизането на екрана. Актуализира UI елементите с детайли за упражнение.

back\_button(self): Навигира обратно към последния посетен екран.

1.6 `WorkoutScreen`: Показва информация за текуща тренировка и позволява на потребителите да добавят упражнения.

Атрибути:

volume\_label\_value (NumericProperty): Числово свойство за стойността на етикета за обем.

sets\_label\_value (NumericProperty): Числово свойство за стойността на етикета за серии.

Методи:

\_\_init\_\_(db, \*\*kwargs): Инициализира инстанция на WorkoutScreen.

load\_workout\_exercises(routine): Зарежда упражненията за дадената тренировка.

end\_workout(): Завършва тренировката и запазва данните за нея.

back\_button(): Навигира обратно към основния екран.

reset(): Нулира стойностите на етикетите за обем и серии.

2. Връзка с Базата Данни:

- `get\_database`: Функция за установяване на връзка с базата данни MongoDB Atlas, използвана за съхранение на данни на потребителите.

3. Модел на Потребителя:

3.1`*User*`:

Атрибути:

name (str): Името на потребителя.

email (str): Имейл адресът на потребителя.

password (str): Паролата на потребителя.

routines (списък): Списък от обекти от тип Рутина.

workouts (списък): Списък от обекти от тип Тренировка.

Методи:

add\_routine(routine): Добавя рутина към списъка на потребителя.

remove\_routine(routine): Премахва рутина от списъка на потребителя.

update\_name(new\_name): Обновява името на потребителя.

update\_email(new\_email): Обновява имейл адреса на потребителя.

update\_password(new\_password): Обновява паролата на потребителя.

add\_workout(workout): Добавя тренировка към списъка на потребителя.

remove\_workout(workout): Премахва тренировка от списъка на потребителя.

get\_exercise\_history(exercise\_id): Връща историята на тренировки за определено упражнение.

get\_all\_exercise\_details(): Извлича детайли за всички упражнения в тренировките на потребителя.

to\_dict(): Преобразува обекта потребител в речник.

from\_dict(data): Създава обект потребител от речник.

3.2 `*Routine*`:

Атрибути:

name (str): Името на рутината.

exercises\_in\_routine (списък): Списък от упражнения в рутината, всяко представено като речник с 'exercise\_id' и 'sets'.

Методи:

to\_dict(): Преобразува обекта рутина в речник.

from\_dict(routine\_data): Създава обект Рутина от речник.

3.3 `Workout`:

Атрибути:

date (str): Дата на тренировката.

title (str): Заглавие на тренировката.

exercises (списък): Списък от упражнения в тренировката, всяко представено като речник с 'exercise\_id' и 'sets'.

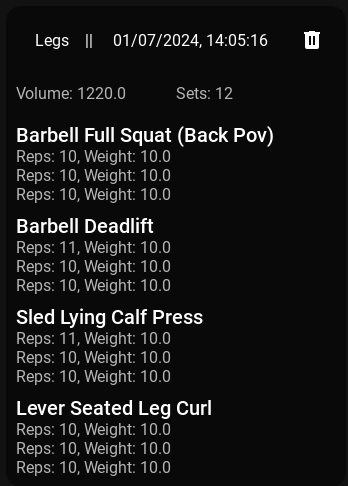
Методи:

add\_exercise(exercise\_id, sets): Добавя ново упражнение към тренировката.

to\_dict(): Преобразува обекта тренировка в речник.

from\_dict(workout\_data): Създава обект Тренировка от речник.

4. Потребителски Карти и Оформления:

 4.1 `UserWorkoutCard`: Показва подробна информация за тренировка на потребителя.

Методи:

\_\_init\_\_(workout, exercise\_dict, delete\_callback, \*\*kwargs): Инициализира инстанция на UserWorkoutCard.

Параметри:

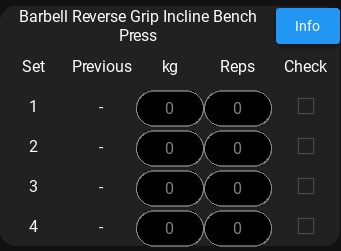
workout (Workout): Обект за тренировка с детайли.

exercise\_dict (dict): Речник с детайли за упражнение добавено за информация за името на упраженението.

delete\_callback (function): Callback функция за изтриване на тренировка.

4.2 `WorkoutExerciseCard`: Показва информация за упражнения в тренировка.

Атрибути:



ws (WorkoutScreen): Справка към родителския WorkoutScreen.

exercise (dict): Потребителската информация за упраженението

exercise\_info (dict): Информация за самото упражнението от бд.

Методи:

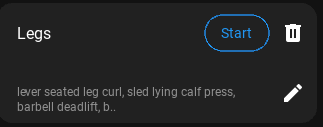
\_\_init\_\_(exercise, exercise\_info, screen\_ref, \*\*kwargs): Инициализира инстанция на WorkoutExerciseCard.

open\_exercise(exercise, button\_instance): Отваря екрана за упражнение.

get\_exercise\_data(): Извлича данни за упражнението от картата.

checkbox\_callback(checkbox, value): Метод за обратно извикване при промяна на състоянието на отметката.

4.3 `RoutineCard`:

Атрибути:

size: Размерът на картата.

orientation: Ориентация на вертикален layout.

size\_hint\_y: Указва височинното поведение спрямо родителя.

height: Височината на картата.

routine: Тренировъчната рутина, свързана с картата.

Методи:

\_\_init\_\_(self, routine, exercises\_dict, delete\_callback=None, start\_callback=None, edit\_button=None, Window=None, \*\*kwargs):

Инициализира RoutineCard с подробности за рутината и колбекове.

Създава вертикален layout с два хоризонтални box layout-а за показване на информацията за рутината.

Изчислява и задава размера и размерностите на базата на ширината на прозореца и density-independent pixels (dp).

Попълва картата с име на рутината, бутон за стартиране, бутон за изтриване и скъсен списък с упражнения.

Изпълнява предоставените колбекове (start\_callback, delete\_callback, edit\_button) при натискане на съответните бутони.

Параметри:

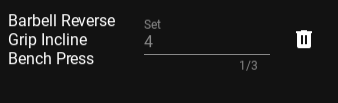
exercise (dict): Детайли за упражнението.

exercise\_info (dict): Информация за упражнението.

screen\_ref (WorkoutScreen): Справка към родителския WorkoutScreen.

\*\*kwargs: Допълнителни именовани аргументи за MDCard.

4.4 `ExerciseCard`:



Атрибути:

title: MDLabel за показване на името на упражнението.

icon: MDIconButton за бутона за изтриване.

set\_input: MDTextField за въвеждане на броя на сериите.

Методи:

\_\_init\_\_(self, exercise\_name, num\_sets=1, \*\*kwargs):

Инициализира ExerciseCard.

set\_delete\_button(self, func):

Задава функцията, която да бъде извикана при натискане на бутона за изтриване.

get\_set\_input\_text(self):

Връща текста, въведен в MDTextField за серии.

Параметри:

exercise\_name (str): Име на упражнението.

num\_sets (int): Брой на сериите (по подразбиране е 1).

\*\*kwargs: Допълнителни именовани аргументи за MDBoxLayout.

4.5 `ClickableTwoLineListItem`:

Атрибути:



exercise (dict): Информация за упражнението, което да се показва.

index (int): Индекс на елемента.

on\_label\_touch (function): Функция за обратно извикване при докосване на етикета.

clicked (bool): Свойство за проследяване дали елементът е щракнат.

Методи:

\_\_init\_\_(self, exercise, index, on\_label\_touch, \*\*kwargs):

Инициализира ClickableTwoLineListItem.

on\_label\_touch\_down(self, instance, touch):

Обработва събитието при докосване на етикета. Превключва състоянието на щракнат и уведомява обратно извикването.

draw\_selection\_line(self):

Изчертава линия за избор върху виджета в зависимост от състоянието му на щракнато.

Параметри:

exercise (dict): Информация за упражнението, което да се показва.

index (int): Индекс на елемента.

on\_label\_touch (function): Функция за обратно извикване при докосване на етикета.

\*\*kwargs: Допълнителни именовани аргументи за BoxLayout.

5. Библиотеки и модули:

* Kivy:

Kivy е отворена библиотека на Python, предназначена за създаване на мултимедийни приложения. Тя предоставя инструменти за разработка на кросплатформени приложения с графичен потребителски интерфейс (GUI). Kivy поддържа различни устройства, включително компютри, смартфони и таблети.

* KivyMD:

KivyMD (Kivy Material Design) е допълнение към Kivy, предоставящо компоненти и стилове в стил Material Design от Google. Това улеснява създаването на приложения със съвременен и привлекателен дизайн.

* PyMongo:

PyMongo е официален драйвер за Python за MongoDB. Този драйвер предоставя прост и удобен начин за взаимодействие с базата данни MongoDB чрез програмиране на Python.

* Kivy Garden:

Kivy Garden представлява набор от допълнителни модули и инструменти, разработени от общността на Kivy. Тези модули предлагат различни функционалности и разширения за Kivy.

* Matplotlib:

Matplotlib е библиотека на Python за визуализация на данни. Тя предоставя разнообразни възможности за създаване на графики, диаграми и други видове визуални представяния на данни.

* Pytz и Tzlocal:

Pytz е библиотека, която предоставя поддръжка за работа с часови зони в Python. Tzlocal е част от Pytz и предоставя инструменти за работа с локалната часова зона на компютъра.

Функционалности

1. Проследяване на Тренировки:

- Потребителите могат да създават и редактират тренировъчни програми.

- Следят упражнения, серии, повторения и тежести по време на тренировка.

2. Информация за Профила:

- Преглеждане и актуализация на информацията за потребителски профил.

- Показване на статистика и история на тренировки.

3. База от Данни с Упражнения:

- Търсене и избор на упражнения на база части от тялото.

- Добавяне на упражнения към тренировъчна програма.

4. Запазване на Данни:

- Потребителските данни се съхраняват в база данни MongoDB Atlas.

- Безпроблемно извличане и актуализиране на информацията на потребителя.

Използване

1. Главен Екран:

- Навигирайте между разделите "Тренировка" и "Профил" чрез лентата за навигация в долната част.

- Достъп и управление на тренировъчни програми.

2. Екран за Редактиране на Програма:

- Редактиране и запазване на промени в тренировъчна програма.

3. Екран с Упражнения за Части от Тялото:

- Търсене и избор на упражнения на база части от тялото.

4. Екран с Упражнения:

- Преглед на упражнения, организирани по раздели.

- Добавяне на упражнения към програма.

5. Екран за Тренировка:

- Прослеждане на текущи тренировки.

- Добавяне и актуализиране на серии упражнения.

**Линк за клониране от репо**

git clone https://github.com/Neldrom/GymProProject.git