Отчет тестирования функциональностей MVP

Функциональное тестирование проводилось для:

1. Экран авторизации: экран регистрации, экран входа и экран восстановления пароля. 7 тест кейсов из 7 прошли успешно
2. Главный экран: проверяется корректность отображения графика веса и активности, перенаправление на другие экраны. 6 тест кейсов из 6 прошли успешно
3. Блок А (экран тренировок): проверяется возможность начать тренировку, а также создание и редактирование тренировок с аккаунта админа. 9 тест кейсов из 9 прошли успешно
4. Личный кабинет: проверяется возможность редактирования пароля и почты, а также выход из аккаунта. 7 тест кейсов из 7 прошли успешно

Юнит тесты

Юнит-тесты проверяют отдельные компоненты приложения в изоляции, используя моки для замены внешних зависимостей. Эти тесты быстрые и не требуют подключения к базе данных или другим внешним сервисам.

В нашем проекте интеграционные тесты реализованы через моки, эмулирующие поведение внешних зависимостей. Это позволяет проверять взаимодействие компонентов без реальных внешних сервисов.

1. Unit-тесты для сервиса Auth\_service
   1. Test\_auth\_service

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_hash\_password | Проверяет корректность хеширования пароля |
| test\_normalize\_contact | Проверяет нормализацию email (удаление пробелов и приведение к нижнему регистру) |
| test\_generate\_verification\_code | Проверяет генерацию кода верификации (должен возвращать строку длиной 6 символов) |
| test\_register\_user\_success | Проверяет успешную регистрацию пользователя с корректными данными |
| test\_register\_user\_existing\_email | Проверяет обработку попытки регистрации с уже существующим email |
| test\_register\_user\_invalid\_email | Проверяет обработку некорректного email (содержащего пробелы) |
| test\_verify\_user\_success | Проверяет успешную верификацию пользователя по коду |

* 1. Test\_router

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_register\_user\_failure | Проверяет обработку ошибки при регистрации (дубликат email) |
| test\_verify\_code\_success | Проверяет успешную верификацию пользователя по коду |
| test\_verify\_code\_failure | Проверяет обработку неверного кода верификации |
| test\_login\_user\_success | Проверяет успешный вход пользователя с верными учетными данными |
| test\_login\_user\_failure | Проверяет обработку неверных учетных данных при входе |
| test\_refresh\_token\_success | Проверяет успешное обновление токена доступа |
| test\_refresh\_token\_blacklisted | Проверяет обработку попытки обновления отозванного токена |

* 1. Test\_token\_service

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_create\_token\_payload | Проверяет создание payload для токена с правильными полями и временными метками |
| test\_create\_access\_token | Проверяет создание access токена с дефолтным временем жизни |
| test\_create\_access\_token\_with\_custom\_expiry | Проверяет создание access токена с кастомным временем жизни |
| test\_create\_refresh\_token | Проверяет создание refresh токена с правильными параметрами |
| test\_create\_reset\_password\_token | Проверяет создание токена для сброса пароля |
| test\_decode\_jwt\_valid | Проверяет декодирование валидного JWT токена |
| test\_decode\_jwt\_expired | Проверяет обработку истекшего токена |

1. Unit-тесты для сервиса Workout\_service
   1. test\_exercises\_service

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_get\_all\_exercises | проверяет получение всех упражнений |
| test\_get\_exercise\_by\_id | проверяет получение упражнения по ID |
| test\_create\_exercise | проверяет создание нового упражнения |
| test\_update\_exercise | проверяет обновление существующего упражнения |
| test\_delete\_exercise | проверяет удаление упражнения |

* 1. test\_training\_service

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_get\_trainings | проверяет получение списка всех тренировок |
| test\_get\_training\_by\_id | проверяет получение тренировки по ID |

* 1. test\_api

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_read\_root | проверяет ответ от корневого маршрута API |
| test\_get\_exercises\_unauthorized | проверяет поведение API при неавторизованном доступе |
| test\_get\_exercises\_authorized | проверяет получение списка упражнений авторизованным пользователем |
| test\_get\_exercise\_by\_id\_authorized | проверяет получение упражнения по ID авторизованным пользователем |
| test\_get\_muscle\_groups\_authorized | проверяет получение списка групп мышц авторизованным пользователем |

* 1. test\_api\_simple

|  |  |
| --- | --- |
| Тест | Что проверяет |
| test\_get\_exercises | проверяет метод получения всех упражнений |
| test\_get\_exercise\_by\_id | проверяет метод получения упражнения по ID |
| test\_get\_muscle\_groups | проверяет метод получения всех групп мышц |

* 1. test\_database

|  |  |
| --- | --- |
| **Тест** | **Что проверяет** |
| test\_database\_connection | проверяет подключение к базе данных и выполнение простого запроса SELECT 1 |
| test\_get\_exercises\_table\_exists | проверяет существование таблицы exercises |
| test\_get\_muscle\_groups\_table\_exists | проверяет существование таблицы muscle\_groups |
| test\_fetch\_exercises | проверяет получение всех упражнений из базы данных |
| test\_fetch\_exercise\_by\_id | проверяет получение упражнения по ID |

Метрики

Всего на данном этапе было написано 83 теста.

Для реализованного функционала и экранов было выполнено 70 тестов.

13 тестов затрагивают нереализованный функционал.

Найденные баги

**Нашли и пофиксили**

Критичные баги

1. Не работал график из сервиса activity\_service из-за ошибки типа данных
2. Удаление гифок не работало из-за проблем с CORS

**Нашли и работаем над исправлением**

Не критичные баги

1. Если в тренировке два одинаковых упражнения подряд идут, то кол-во повторений не считается
2. При заполнении упражнений возникает ошибка при задании названия более 25 символов, пока нет валидации