МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ И РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего

образования «МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ОТЧЁТ ПО ДИСЦИПЛИНЕ «ПРОЕКТНАЯ ДЕЯТЕЛЬНОСТЬ»

на тему:

“ami” - приложение для здорового   
образа жизни.

Куратор проект: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_ /

подпись ФИО, уч. звание и степень

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Давтаев Артур Русланович, 181-322 /

подпись ФИО, группа

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Злобин Никита Андреевич, 181-322 /

подпись ФИО, группа

Москва, 2021

**СОДЕРЖАНИЕ**

[**1. АННОТАЦИЯ И ИНФОРМАЦИЯ О ПРОЕКТЕ 3**](#_Toc69489859)

[**2. ПЛАН РАБОТЫ 4**](#_Toc69489860)

[**3. УЧАСТНИКИ И ИХ РОЛИ 5**](#_Toc69489861)

[**4. ОПИСАНИЕ ЦЕЛЕВОЙ АУДИТОРИИ 6**](#_Toc69489862)

[**1. Сегмент 1. Женщины 20-25 лет 6**](#_Toc69489863)

[**2. Сегмент 2. Мужчины 25-30 лет 7**](#_Toc69489864)

[**5. АНАЛИЗ РЫНКА И КОНКУРЕНТНОЙ СРЕДЫ 9**](#_Toc69489865)

[**1. 6.1 Apple Здоровье 9**](#_Toc69489866)

[**2. 6.2 Sleep Cycle 9**](#_Toc69489867)

[**3. 6.3 Tappsk 9**](#_Toc69489868)

[**6. ДИЗАЙН СИСТЕМЫ 10**](#_Toc69489869)

[**7. ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ 12**](#_Toc69489870)

[**1. 8.1 Используемые технологии 12**](#_Toc69489871)

[**2. Схема базы данных 12**](#_Toc69489872)

[**3. Разработка 13**](#_Toc69489873)

[**8. РЕЗУЛЬТАТЫ 19**](#_Toc69489874)

[**9. ЗАКЛЮЧЕНИЕ 20**](#_Toc69489875)

[**10. СПИСОК ИСТОЧНИКОВ 21**](#_Toc69489876)

# АННОТАЦИЯ И ИНФОРМАЦИЯ О ПРОЕКТЕ

В современном мире человечество активно наблюдает непрерывную цифровизацию всех возможных сфер жизни и общества. Множество людей даже не задумываются о том, какую часть их жизни занимают технологии, в том числе, смартфоны и всевозможные приложения. Из возникшего желания делать что-то технологичное и помогающее каждому человеку и родилась идея “Ami”.   
 Здоровый образ жизни с каждым годом становится все более популярен в развитых и развивающихся странах, но многим до сих пор кажется, что следить за своим здоровьем - это сложно и дорого. Однако даже небольшой вклад и несколько потраченных минут в день могут начать улучшать здоровье человека.  
 На рынке существует множество похожих приложений, однако ни одно из них не является достаточно комплексным, не охватывает все сферы жизни. Большинство таких приложений направлены на одну конкретную проблему. Однако пользователи не находят удобным большое количество однотипных приложений на своих устройствах. “Ami” создается для того, чтобы стать универсальным решением для каждого.

Целью данного проекта на 4 семестр является создание MVP (minimum viable product минимально жизнеспособного продукта). Данная цель делится на следующие задачи:

* осмысление концепции проекта;
* создание и ведение git-репозитория;
* создание названия и логотипа;
* ознакомление с информацией о здоровом образе жизни;
* создание UX-прототипа;
* создание UI-макетов;
* создание сайта;
* создание мобильного приложения;
* тестирование;
* документирование.

# ПЛАН РАБОТЫ

Были выделены следующие основные этапы разработки программного продукта:

1. Проанализировать идею и ознакомительную информацию;
2. Проанализировать рынок;
3. Разработать и переработать прототипы интерфейса;
4. Первичная файловая организация проекта;
5. Разработка архитектуры БД;
6. Разработка UI-макетов приложения и сайта;
7. Разработка функциональных страниц;
8. Тестирование;
9. Документирование.

# 

# УЧАСТНИКИ И ИХ РОЛИ

Все участники команды являются студентами кафедры Инфокогнитивных технологий, факультета Информационных технологий, направления Информатика и вычислительная техника, специальности Веб-технологии, и выполняют следующие роли в проекте:

1. **А. Р. Давтаев**, студент группы 181-322 - Project manager, UX-дизайнер, UI-дизайнер, разработчик документации.
2. **Н. А. Злобин**, студент группы 181-322 - архитектор, программист.

# ОПИСАНИЕ ЦЕЛЕВОЙ АУДИТОРИИ

Для описания целевой аудитории проекта был выбран метод персон, его цель — создать нескольких персонажей с характеристиками потенциальных пользователей продукта, т.е. каждый персонаж — это собирательный образ одной группы целевой аудитории.

## Сегмент 1. Женщины 20-25 лет

Имя: Маша Миронова (Рис 5.1.1)



Рис. 5.1.1. Сегмент 1. Женщины 20-25 лет.

* Женщина
* 20 лет
* Дизайнер
* Образование:  
  — Еще учится в этой сфере, получает Высшее образование. Возможна магистратура в будущем.
* Доход:

— около 40 тыс. рублей, возможна помощь родителей

* Семейное положен*ие:*

— В отношениях

* Место проживания:

— Живёт в городе-миллионнике

* Интересы:

— играет на музыкальных инструментах, много читает;

— профессиональное саморазвитие (читает большое количество специальной литературы, литературы об искусстве в целом);

— Занимается спортом.

* Ценности:

— осознанность.

— результативность на работе.

— комфорт.

— постоянное саморазвитие.

— хочет получать удовольствие от жизни.

— успех, подтвержденный какими-либо внешними атрибутами.

Черты характера: самодостаточна, авторитетна, увлечена развитием.

Ценностные установки, связанные с индивидуализмом, конкуренцией, амбициями. В пространстве Psychea этот сегмент называется «синий» или «независимые».

## Сегмент 2. Мужчины 25-30 лет

Макс Калинкин (Рис 5.2.2)



Рис. 5.2.2. Сегмент 2. Мужчины 25-30 лет

* Мужчина
* 25-30 лет
* Программист в крупной IT-компании
* Образование:  
  — имеет высшее образование в своей сфере
* Доход:  
  — более 100 тыс. рублей
* Семейное положение:

— свободен или в недавних отношениях

* Место проживания:

— Живёт в городе-миллионнике

* Интересы:

— путешествия (пляжный отдых), экстремальные путешествия

— хобби — чтение, видеоигры

* Ценности:

— эффективность

— результативность на работе

— свобода

— хочет получать удовольствие от жизни

— успех, который подтверждён не только внутренними ощущениями, но и внешними атрибутами

— ценит время

— саморазвитие и личностный рост

— отношения между людьми на профессиональном уровне, не отличается навязчивой заботой о людях, но важна командная работа

Постоянный поиск способов выделиться формирует систему стратегий и установок — демонстрация, гедонизм, новизна, статус, наслаждение. В пространстве Psychea этот сегмент называется «жёлтый» или «гедонисты».

# АНАЛИЗ РЫНКА И КОНКУРЕНТНОЙ СРЕДЫ

## 6.1 Apple Здоровье

Apple Здоровье — приложение, встроенное в операционную систему Apple, которое содержит в себе множество полезных функций.

В процессе обсуждения платформы Apple Здоровье были выявлены следующие преимущества:

* интеграция приложения в Apple Watch;
* множество функций;
* функция “Осознанность”;
* большое количество графиков.

## 6.2 Sleep Cycle

Sleep Cycle - это приложение, которое помогает следить за сном и содержит в себе будильник. Это приложение только с одной функциональной направленностью.

В процессе обсуждения платформы Sleep Cycle были выявлены следующие преимущества:

* “умный будильник”, который с помощью микрофона отслеживает фазы сна;
* успокаивающие звуки для засыпания;
* отражает статистику качества сна.

## 6.3 Tappsk

Tappsk - это таск-трекер, приложение для постановки и выполнения задач, тренировки осознанности. Это приложение только с одной функциональной направленностью.

В процессе обсуждения платформы Tappsk были выявлены следующие преимущества:

* добавление задач голосом;
* приоритеты задач;
* версия для макбуков и синхронизация с ней.

# ДИЗАЙН СИСТЕМЫ

Разработка дизайна приложения “ami” включала в себя несколько этапов: создание и переработка прототипов, отрисовка первичного дизайна, корректировка дизайна, создание финального дизайна. В онлайн-сервисе для разработки интерфейсов Figma был отрисован макет, включающий в себя основные структурные блоки.



Рис. 7.1. Прототипы таск менеджера

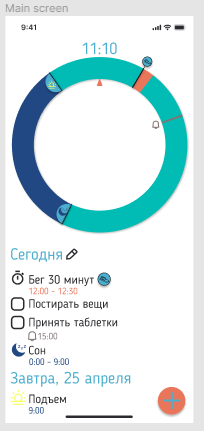
На основе прототипа был разработан первичный дизайн, отражающий наше видение проекта. В качестве основных цветов разрабатываемого сайта была выбрана яркая, но нежная палитра (рис. 7.1).



Рис. 7.2. Палитра цветов

Для представления текстовой информации был выбран шрифт Blogger Sans.

Результат, получившийся после всех исправлений, можно увидеть по ссылке: https://www.figma.com/file/7zzoAG4vYEck8cYM7TVmJT/ami-3.0?node-id=102%3A228. Несколько примеров страниц приложения можно увидеть на рисунках 7.3-7.5



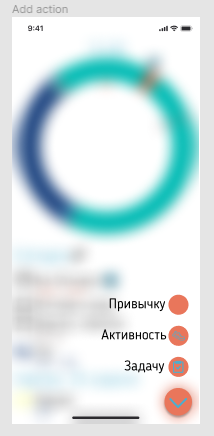


Рис. 7.3

Рис. 7.4

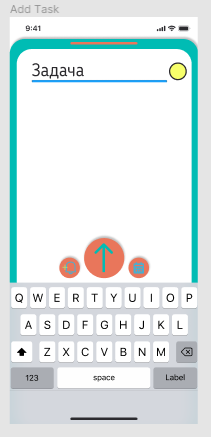


Рис. 7.5

# ТЕХНИЧЕСКАЯ РЕАЛИЗАЦИЯ

## Используемые технологии

В качестве децентрализованного хранилища исходного кода был использован Git, а в качестве удаленного репозитория - GitHub

Для взаимодействия между сервером и клиентом ещё в прошлом семестре была выбрана технология REST API. Для ее реализации был написан сервер на NodeJS с использованием фреймворка Express. Для хранения данных была использована NoSQL база данных MongoDB и SQLite (сейчас по большему счету используется только последняя).

Для взаимодействий с сервером был куплен домен на сайте reg.ru. В качестве хостинга была выбран сервис vscale.io. Однако домен истёк и сейчас разработка сайта временно приостановлена

Мобильное приложение реализовано на языке Dart и, ввиду особенности фреймворка Flutter, способно запускаться как на платформе Android, так и iOS(тестирование проводилось в основном на Android). Структура проекта имеет свою целостность и единый стиль. Весь код написан по принципам ООП и архитектурному паттерну MVP. При реализации интерфейса был использован AutoLayout и относительная верстка для адаптивной верстки приложения на различных устройствах.

## Схема базы данных

Перед разработкой были смоделирована ER-диаграмма схемы основной базы данных и написана документация REST API (<https://github.com/MaximusHeroProger/ami-PD/tree/develop/docs/api-docs>).

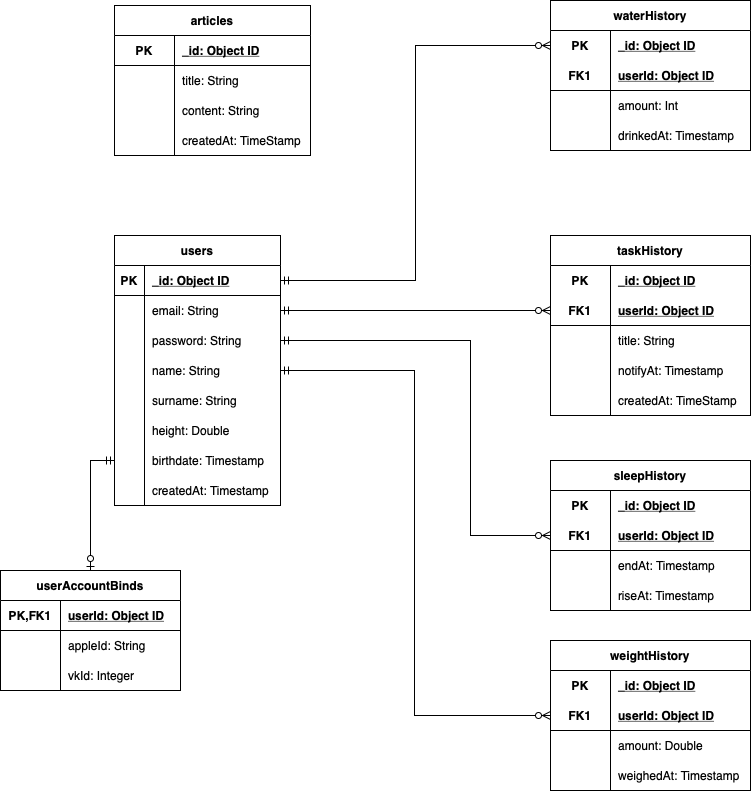


Рис. 7.2.1. ER-диаграмма схемы основной базы данных MongoDB

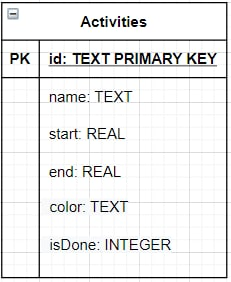


Рис. 7.2.2 – Таблица на SQLite

## Разработка

В первую очередь был разработан виджет – отображение дел пользователя на диаграмме дня Для этого бы использован класс CustomPainter, который на основании класса Canvas и входящих данных отрисовывает промежутки времени, введенные пользователем.

Рис. 7.3.1 – CustomPainter

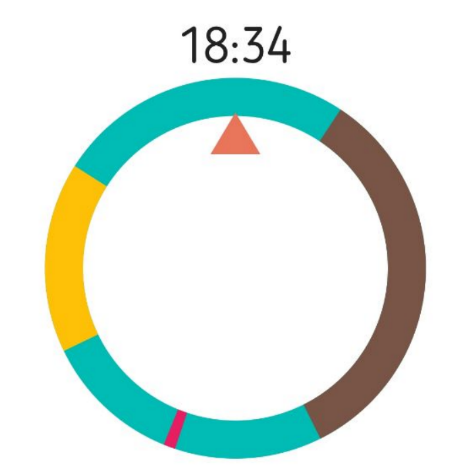


Рис. 7.3.2 – Диаграмма дня

Далее была добавлена база данных. Для этих целей была выбрана база SQLite из библиотеки sqflite. Она удобна тем, что поддерживается как системой android, так и ios. Для работы с ней был создан класс db\_helper. Была создана таблица, в которой будут храниться активности и методы её создания(при первом запуске), добавления, изменения и удаления данных.

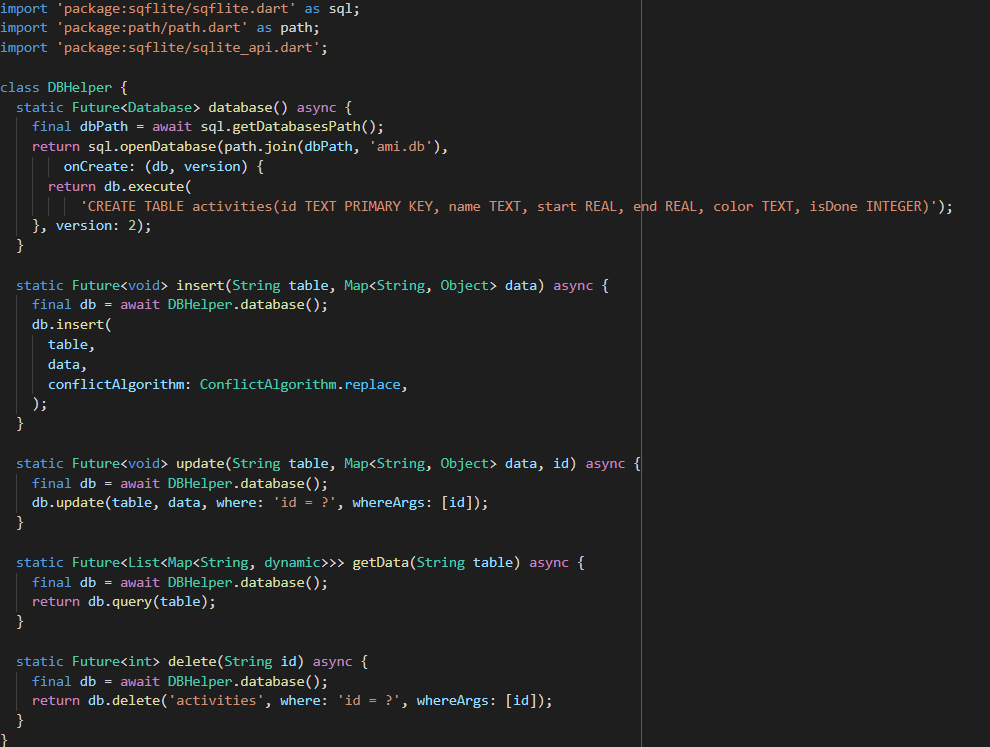


Рис. 7.3.3 – SQLite

Далее были созданы экраны добавления и изменения элементов для пользователя. Они включают в себя поля названия действия, которое пользователь собирается добавить на свою диаграмму, цвет для этого действия, отображаемый на диаграмме, выбор – активность/задача – то есть добавление отсечки времени, либо периода времени.

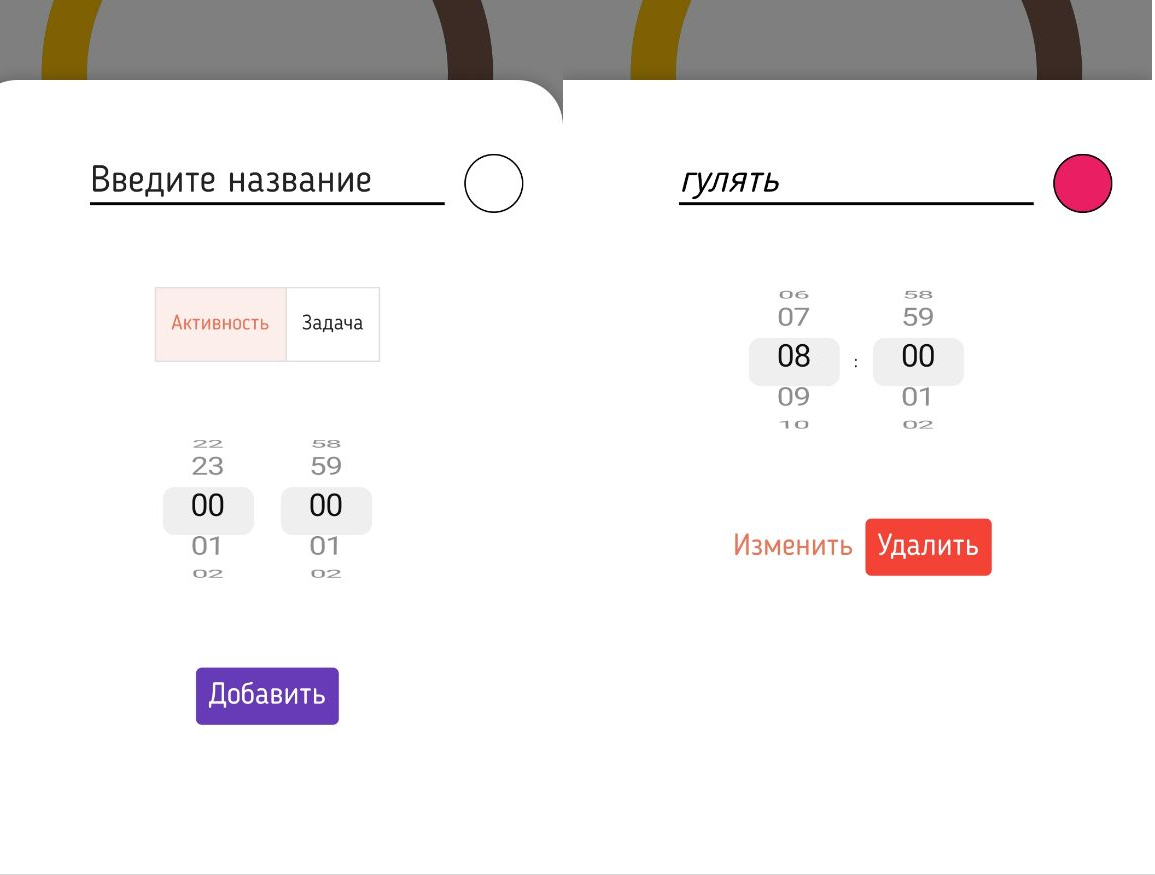


Рис. 7.3.4 – Экраны добавления и изменения элементов

Текст ввода производится с помощью виджета TextField

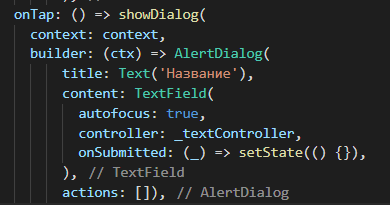


Рис. 7.3.5 – TextField

Для выбора цвета был использован виджет ColorPicker



Рис. 7.3.6 – ColorPicker

Виджеты ввода времени были сконструированы на основе класса CupertinoPicker из библиотеки 'package:flutter/cupertino.dart'.

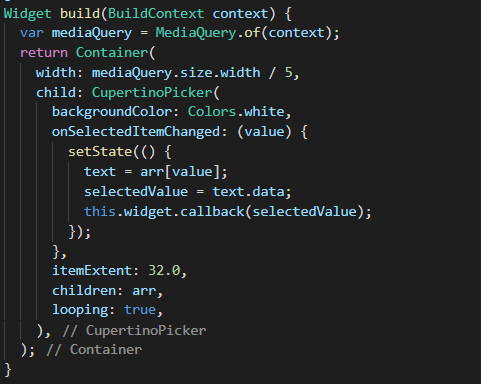


Рис. 7.3.7 – CupertinoPicker

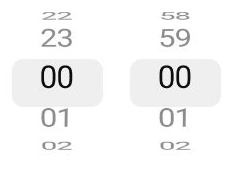


Рис. 7.3.8 – CupertinoPicker

Сама панель с добавлением/изменением элементов из библиотеки sliding\_up\_panel.

Вызывается из виджета - главного экрана. При нажатии кнопки редактирования/добавления элементов в массив виджетов главного экрана добавляется виджет редактирования/ добавления элемента внутри панели. При сворачивании панели или при совершении добавления/удаления виджет редактирования/добавления удаляется из массива, а значит и с главного экрана.

За выбор даты в приложении отвечает виджет из библиотеки material – showDatePicker. Виджет был локализован на русский язык. Для взаимодействия с базой данных на главном экране – отображении и обновлении элементов на главном экране был использован FutureBuilder

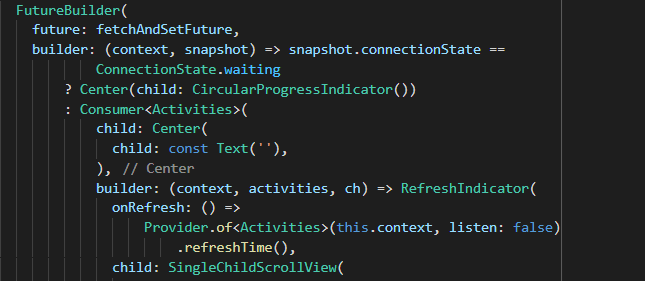


Рис. 7.3.9 – FutureBuilder

Также был разработан алгоритм валидации допустимых промежутков времени



Рис. 7.3.10 – Только часть алгоритма валидации

Для избежания наложения промежутков времени на отсечки времени, при поступлении к диаграмме данные сортируются, отсечки оказываются в конце массива, что позволяет добавлять их поверх промежутков времени и быть им видимыми.

Список активностей, видимой пользователю также сортируется, более ранние помещаются в начало списка

# РЕЗУЛЬТАТЫ

Итоги проектной деятельности по разработке приложения “аmi” могут быть представлены поэтапно в соответствии с утвержденным планом деятельности по созданию информационного продукта. Согласно разработанному плану между участниками проекта были определены следующие роли: разработчик документации, UI-дизайнер, UX-дизайнер, проектировщик, архитектор, IOS-разработчик.

По результатам проведенного анализа потенциальной целевой аудитории, рынка и конкурентной среды был разработан прототип интерфейса и макеты будущего приложения. Следующим этапом работы стало реимплементирование создание архитектуры БД и разработка самого приложения. С результатами проектной деятельности можно ознакомиться по следующим ссылкам:

Репозиторий - https://github.com/cusplayer/ami.pd/tree/app

Презентация - https://github.com/cusplayer/ami.pd/blob/app/ami\_%D0%9F%D1%80%D0%B5%D0%B7%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F%20%D0%9F%D0%94.pptx

Лендинг проекта - <http://ami.std-1071.ist.mospolytech.ru/>

Figma - https://www.figma.com/file/7zzoAG4vYEck8cYM7TVmJT/ami-3.0?node-id=0%3A1

# ЗАКЛЮЧЕНИЕ

В результате проделанной работы было разработано мобильное приложение для сервиса для здорового образа жизни.

Кроме того, в ходе создания данного приложения были получены практические навыки в разработке и изучены множество инструментов для создания мобильных приложений.

В рамках данной работы были получены следующие результаты:

- было разработано множество вариантов макетов для приложения;

- был разработан основной дизайн;

- было сформировано основное ядро работы с данными во всем приложении

- было написано приложение на фреймворке Flutter;

- макеты приложения были проверены на функциональность на реальном устройстве.

# 

# СПИСОК ИСТОЧНИКОВ

1. Акулов, И.Б. SwiftBook [Электронный ресурс] / И.Б. Акулов. — Электрон. текстовые дан. — Режим доступа: <https://swiftbook.ru/contents/doc/>, свободный. — Русская документация Apple Swift Programming Language
2. Core Data в деталях [Электронный ресурс] / IrixV. — Электрон. текстовые дан. — 2019. — Режим доступа: <https://habr.com/ru/post/436510/>, свободный
3. Human Interface Guidelines [Электронный ресурс] / Apple Inc.. — Электрон. текстовые дан. — Режим доступа: <https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/>, свободный
4. UIKit [Электронный ресурс] / Apple Inc.. — Электрон. текстовые дан. — Режим доступа: <https://developer.apple.com/documentation/uikit>, свободный
5. Foundation | Apple Developer Documentation [Электронный ресурс] / Apple Inc.. — Электрон. текстовые дан. — Режим доступа: <https://developer.apple.com/documentation/foundation>, свободный
6. Swift | Apple Developer Documentation [Электронный ресурс] / Apple Inc.. — Электрон. текстовые дан. — Режим доступа: <https://developer.apple.com/documentation/swift>, свободный
7. Не заставляйте меня думать /Стив Круг; пер. с англ. М.А. Райтмана. — 3-е. — Москва: Издательство "Э", 2018. — 256с
8. Stack Overflow [Электронный ресурс] /. — Электрон. текстовые дан. — Режим доступа: <https://stackoverflow.com>, свободный. — Создание сообществ, где каждый, кто кодирует, может учиться и делиться своими знаниями
9. Better Routing for IOS Applications with Router in Swift [Электронный ресурс] / Seyhun AKYÜREK. — Электрон. текстовые дан. — Medium, 2016. — Режим доступа: <https://medium.com/swift-programming/better-routing-for-ios-applications-with-router-in-swift-9b47cc8cb4c7>, свободный
10. Погружение в паттерны проектирование /Александр Швец. — 2-е изд., электрон. — Refactoring Guru, 2018.
11. Pro Git /Scott Chacon, Ben Straub. — 2-е изд. — Apress, 2014. — 419с.
12. Архитектурные паттерны в iOS [Электронный ресурс] / KAIIIAJIOT. — Электрон. текстовые дан. — habr, 2016. — Режим доступа: <https://habr.com/ru/company/badoo/blog/281162/>, свободный
13. How to convert a hex color to a UIColor [Электронный ресурс] / Paul Hudson. — Электрон. текстовые дан. — hacking with swift, 2019. — Режим доступа: <https://www.hackingwithswift.com/example-code/uicolor/how-to-convert-a-hex-color-to-a-uicolor>, свободный
14. How To Delete Every Record Of A Core Data Entity [Электронный ресурс] / Bart Jacobs. — Электрон. текстовые дан. — cocoa casts, 2016. — Режим доступа: <https://cocoacasts.com/how-to-delete-every-record-of-a-core-data-entity>, свободный
15. Hashable [Электронный ресурс] / Apple Inc.. — Электрон. текстовые дан. — Режим доступа: <https://developer.apple.com/documentation/swift/hashable>, свободный
16. Fetching Remote Data With Core Data Background Context in iOS App [Электронный ресурс] / Alfian Losari. — Электрон. текстовые дан. — Medium, 2018. — Режим доступа: <https://medium.com/swift2go/fetching-remote-data-with-core-data-background-context-in-ios-app-224dad15ef6c>, свободный
17. Dark Mode: Adding support to your app in Swift [Электронный ресурс] / SwiftLee. — Электрон. текстовые дан. — AvanderLee, 2019. — Режим доступа: https:// [www.avanderlee.com/swift/dark-mode-support-ios/](http://www.avanderlee.com/swift/dark-mode-support-ios/), свободный
18. RAML 1.0 Specification [Электронный ресурс] / jstoiko. - Электрон. тектовые дан. - GitHub, 2020. - Режим доступа: <https://github.com/raml-org/raml-spec/blob/master/versions/raml-10/raml-10.md/>, свободный
19. How to dismiss the keyboard for a TextField [Электронный ресурс] / Paul Hudson. — Электрон. текстовые дан. — hacking with swift, 2020. — Режим доступа: <https://www.hackingwithswift.com/quick-start/swiftui/how-to-dismiss-the-keyboard-for-a-textfield>, свободный
20. How to Build a Form UI with SwiftUI [Электронный ресурс] / Simon NG. — Электрон. текстовые дан. — APPCODA, 2019. — Режим доступа: <https://www.appcoda.com/swiftui-form-ui/>, свободный
21. How to swipe to delete UITableViewCells [Электронный ресурс] / Paul Hudson. — Электрон. текстовые дан. — hacking with swift, 2019. — Режим доступа: <https://www.hackingwithswift.com/example-code/uikit/how-to-swipe-to-delete-uitableviewcells>, свободный
22. Кастомный Navigation Bar [Электронный ресурс] / Михаил Цейтлин. — Электрон. текстовые дан. — SwiftBook, 2019. — Режим доступа: <https://swiftbook.ru/post/tutorials/custom-navigation-bar/>, свободный
23. Пишем документацию API при помощи RAML [Электронный ресурс] / yuhenobi. — Электрон. текстовые дан. — Habr, 2015. — Режим доступа: <https://habr.com/ru/company/selectel/blog/265337/>, свободный
24. RAML Tutorial [Электронный ресурс] / Akshar Goswami, — Электрон. журн. — YouTube, 2017 — Режим доступа: <https://www.youtube.com/watch?v=rWLJFkKkVHY>, свободный
25. Unit testing asynchronous Swift code [Электронный ресурс] / John Sundell. — Электрон. текстовые дан. — Swift by Sundell, 2018. — Режим доступа: <https://www.swiftbysundell.com/articles/unit-testing-asynchronous-swift-code/>, свободный
26. CryptoSwift Documentation [Электронный ресурс] / krzyzanowskim. — Электрон. текстовые дан. — CryptoSwift, 2020. — Режим доступа: <https://cryptoswift.io>, свободный
27. UICollectionView Tuyorial: Getting Started [Электронный ресурс] / Brody Eller. — Электрон. текстовые дан. — RayWenderlich, 2019. — Режим доступа: <https://www.raywenderlich.com/9334-uicollectionview-tutorial-getting-started>, свободный
28. How to recolor UIImages using template images and withRenderingMode() [Электронный ресурс] / Brody Eller. — Электрон. текстовые дан. — Hacking with Swift, 2019. — Режим доступа: <https://www.hackingwithswift.com/example-code/uikit/how-to-recolor-uiimages-using-template-images-and-withrenderingmode>, свободный
29. Вода [Электронный ресурс] / ВОЗ. — Электрон. текстовые дан. — ВОЗ, 2020. — Режим доступа: <https://www.who.int/topics/water/ru/>, свободный
30. Как быстро и точно рассчитать свой правильный вес [Электронный ресурс] / Андрей Можевеенко. — Электрон. текстовые дан. — Stone Forest, 2020. — Режим доступа: <https://stoneforest.ru/sport/kak-rasschitat-pravilnyj-ves-cheloveka/>, свободный
31. Animate UIWindow’s rootViewController transitions / [Daniele Margutti](https://medium.com/@danielemargutti?source=post_page-----2887ccf3fecc----------------------). — Электрон. текстовые дан. — Medium, 2017. — Режим доступа: <https://medium.com/@danielemargutti/animate-uiwindows-rootviewcontroller-transitions-2887ccf3fecc>, свободный
32. How to adjust a UIScrollView [Электронный ресурс] / Paul Hudson. — Электрон. текстовые дан. — Hacking with Swift, 2019. — Режим доступа: <https://www.hackingwithswift.com/example-code/uikit/how-to-adjust-a-uiscrollview-to-fit-the-keyboard>, свободный
33. An Extension That Shows a Basic Alert - Swift [Электронный ресурс] / JaredDavidson. — Электрон. текстовые дан. — Medium, 2018. — Режим доступа: <https://medium.com/@Archetapp/an-extension-that-shows-a-basic-alert-swift-f0d250224e8c>, свободный
34. IBDesignable and IBInspectable in Swift [Электронный ресурс] / Anand Nimje. — Электрон. текстовые дан. — Medium, 2018. — Режим доступа: <https://medium.com/@nimjea/ibdesignable-and-ibinspectable-in-swift-c12ea557b82b>, свободный
35. Autolayout for ScrollView + Keyboard Handling in iOS [Электронный ресурс] / Andy Nguyen. — Электрон. текстовые дан. — Medium, 2017. — Режим доступа: <https://medium.com/@andy.nguyen.1993/autolayout-for-scrollview-keyboard-handling-in-ios-5a47d73fd023>, свободный
36. Date Picker Mode for UITextField (Swift 5 / Xcode 11) [Электронный ресурс] / PlanetVeracity, — Электрон. журн. — YouTube, 2020 — Режим доступа: <https://www.youtube.com/watch?v=8NngJrVFfUw>, свободный
37. How to move to the next UITextField when the user pressed return [Электронный ресурс] / Paul Hudson, — Электрон. журн. — Hacking with Swift, 2019 — Режим доступа: <https://www.hackingwithswift.com/example-code/uikit/how-to-move-to-the-next-uitextfield-when-the-user-presses-return>, свободный
38. How to Use Custom Fonts Sith SwiftUI [Электронный ресурс] / Vinny Allen, — Электрон. журн. — Medium, 2019 — Режим доступа: <https://medium.com/better-programming/swiftui-basics-importing-custom-fonts-b6396d17424d>, свободный
39. How to draw a border around a view [Электронный ресурс] / Paul Hudson, — Электрон. журн. — Hacking with Swift, 2019 — Режим доступа: <https://www.hackingwithswift.com/quick-start/swiftui/how-to-draw-a-border-around-a-view>, свободный
40. How to add a custom initializer to a struct without losing its memberwise initializer [Электронный ресурс] / Paul Hudson, — Электрон. журн. — Hacking with Swift, 2019 — Режим доступа: <https://www.hackingwithswift.com/example-code/language/how-to-add-a-custom-initializer-to-a-struct-without-losing-its-memberwise-initializer>, свободный
41. Sign In with Apple using Swift [Электронный ресурс] / Priya Talreja, — Электрон. журн. — Medium, 2019 — Режим доступа: <https://medium.com/@priya_talreja/sign-in-with-apple-using-swift-5cd8695a46b6>, свободный
42. Методы HTTP запроса [Электронный ресурс], — Электрон. журн. — MDN web docs— Режим доступа: <https://developer.mozilla.org/ru/docs/Web/HTTP/Methods>, свободный
43. BSON Types [Электронный ресурс], — Электрон. журн. — mongoDB — Режим доступа: <https://docs.mongodb.com/manual/reference/bson-types/>, свободный
44. Тестирование RESTful API на NodeJS с Mocha и Chai [Электронный ресурс] / aiQlagin, — Электрон. журн. — Хабр, 2016 — Режим доступа: <https://habr.com/ru/post/308352/>, свободный
45. RESTful API на Node.js + MongoDB [Электронный ресурс] / BenderRodriguez, — Электрон. журн. — Хабр, 2013 — Режим доступа: <https://habr.com/ru/post/193458/>, свободный
46. REST API. Практический Курс. Пишем Fullstack приложение на JavaScript [Электронный ресурс] / Владилен Минин, — Электрон. журн. — YouTube, 2020 — Режим доступа: <https://www.youtube.com/watch?v=lzQIhjElV_g>, свободный
47. MERN - Сокращение Ссылок с Нуля до Деплоя (Mongo, Express, React, Node) [Электронный ресурс] / Владилен Минин, — Электрон. журн. — YouTube, 2019 — Режим доступа: <https://www.youtube.com/watch?v=ivDjWYcKDZI>, свободный
48. NodeJS, Express, MongoDB - Приложение целиком [Электронный ресурс] / Владилен Минин, — Электрон. журн. — YouTube — Режим доступа: <https://www.youtube.com/watch?v=8bE_PBRriyU>, свободный
49. Принципы построения REST JSON API [Электронный ресурс] / vanxant, — Электрон. журн. — Хабр, 2019 — Режим доступа: <https://habr.com/ru/post/447322/>, свободный
50. Разработка web API [Электронный ресурс] / 57uff3r, — Электрон. журн. — Хабр, 2013 — Режим доступа: <https://habr.com/ru/post/181988/>, свободный
51. REST API Best Practices [Электронный ресурс] / AntonShtabnoy, — Электрон. журн. — Хабр, 2018 — Режим доступа: <https://habr.com/ru/post/351890/>, свободный
52. Node.js v14.5.0 Documentation [Электронный ресурс], — Электрон. журн. — Node.js— Режим доступа: <https://nodejs.org/dist/latest-v14.x/docs/api/>, свободный
53. PM2 API Documentation [Электронный ресурс], — Электрон. журн. — PM2 — Режим доступа: <https://pm2.keymetrics.io/docs/usage/pm2-api/>, свободный