**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Руководитель проекта,  Приглашенный преподаватель Базовой кафедры «Системное программирование» Института системного программирования им. В.П. Иванникова РАН (ИСП РАН)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Кучук  **«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_** 2021 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов  **«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_** 2021 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | RU.17701729.04.01-01 81 01-1-ЛУ | | **КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ГЕОТАРГЕТИРОВАННОЙ РЕКЛАМЫ «ГЕОФУД». МОБИЛЬНОЕ ПРИЛОЖЕНИЕ**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.04.01-01 81 01-1-ЛУ** | | |
|  |  | |
| Исполнитель:  студент группы БПИ194  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Е. В. Аникеев /  **«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** 2021 г. | |
|  | | |
|  | |  |

**Москва 2021**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.04.01-01 81 01-1-ЛУ |  |  | | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** | RU.17701729.04.01-01 81 01-1-ЛУ | | **КЛИЕНТ-СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ГЕОТАРГЕТИРОВАННОЙ РЕКЛАМЫ «ГЕОФУД». МОБИЛЬНОЕ ПРИЛОЖЕНИЕ**  **Пояснительная записка**  **RU.17701729.04.01-01 81 01-1**  **Листов 53** | | | |
|  |  | | |
|  | | |
|  | | | |
|  | | |  |

**Москва** **2021**

Содержание

[АННОТАЦИЯ 3](#_Toc71389827)

[1. ВВЕДЕНИЕ 4](#_Toc71389828)

[1.1. Наименование программы 4](#_Toc71389829)

[1.2. Документы, на основании которых ведется разработка программ 4](#_Toc71389830)

[2. Назначение и область применения 5](#_Toc71389831)

[2.1. Функциональное назначение 5](#_Toc71389832)

[2.2. Эксплуатационное назначение 5](#_Toc71389833)

[3. Технические характеристики 6](#_Toc71389834)

[3.1. Постановка задачи на разработку программы 6](#_Toc71389835)

[3.2. Описание алгоритма и функционирования программы 6](#_Toc71389836)

[3.3. Обоснование выбора архитектуры приложения 8](#_Toc71389837)

[3.4. Описание и обоснование выбора технических и программных средств 8](#_Toc71389838)

[3.5. Эксплуатационное назначение 9](#_Toc71389839)

[4. ожидаемые технко-экономические показатели 10](#_Toc71389840)

[4.1. Предполагаемая потребность 10](#_Toc71389841)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 10](#_Toc71389842)

[5. Список литературы 11](#_Toc71389843)

[ПРИЛОЖЕНИЕ 1 12](#_Toc71389844)

[ТЕРМИНОЛОГИЯ 12](#_Toc71389845)

[ПРИЛОЖЕНИЕ 2 13](#_Toc71389846)

[ПРЕДСТАВЛЕНИЕ ТАБЛИЦ БАЗЫ ДАННЫХ 13](#_Toc71389847)

[ПРИЛОЖЕНИЕ 3 14](#_Toc71389848)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 14](#_Toc71389849)

[ПРИЛОЖЕНИЕ 4 17](#_Toc71389850)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ КЛИЕНТА 17](#_Toc71389851)

[6. Лист регистрации изменений 51](#_Toc71389852)

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к программе «GeoFood» («Клиент-серверное приложение геотаргетированной рекламы «Геофуд». Мобильное приложение»), предназначенной для применения пользователями в повседневной жизни для облегчения выбора товаров магазинов

В разделе «Введение» указано наименование и краткая характеристика области применения программы.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

* постановка задачи на разработку программ;
* описание алгоритма и функционирования программы с обоснованием выбора схемы алгоритма решения задачи;
* описание и обоснование выбора метода организации входных и выходных данных;
* описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.103-77 Обозначения программ и программных документов [1];
2. ГОСТ 19.104-78 Основные надписи [2];
3. ГОСТ 19.105-78 Общие требования к программным документам [3];
4. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [4];
5. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению [5].

Изменения к техническому заданию оформляются согласно ГОСТ 19.603-78 [6], ГОСТ 19.604- 78 [7].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

1. ВВЕДЕНИЕ
   1. Наименование программы

Наименование темы разработки: «Клиент-серверное приложение геотаргетированной рекламы «ГеоФуд».

Наименование темы разработки на английском языке: «Client-server application of geo-targeted advertising «GeoFood».

* 1. Документы, на основании которых ведется разработка программ

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 "Программная инженерия" и утвержденная академическим руководителем тема курсового проекта.

1. Назначение и область применения
   1. Функциональное назначение

Программа должна выполнять следующие функции:

1. Отслеживать перемещения пользователей;
2. Анализировать полученные данные и строить по ним вероятные маршруты пользователей;
3. Предоставлять возможность компаниям предоставлять уникальные акционные предложения для будущих потенциальных клиентов;
4. Предлагать пользователям персонализированные акции.
   1. Эксплуатационное назначение

Приложение должно помочь свести потенциального потребителя и компанию, предоставляющую услуги питания, предоставляя специализированные акции на основе частых перемещений пользователя вблизи кафе или ресторана.

* 1. Характеристика области применения

Сервис предоставляет акции для пользователей, которые постоянно проходят мимо заведения, но не пользуются его услугами, однако могут стать его постоянными клиентами. Персональные акции могут стать мотиватором для похода в новое кафе, ресторан и т.п. Клиент впервые для себя открывает новое место и возможно останется его постоянным посетителем.

Приложение учитывает индивидуальные предпочтения пользователей и подстраивает рекламу под его потребности, например утром - кофе вечером - рестораны.

1. Технические характеристики
   1. Постановка задачи на разработку программы
      1. Разрабатываемое приложение должно иметь следующие функции:
2. Авторизация по логину и паролю;
3. Создание и редактирование профиля;
4. Просмотр карты с ресторанами
5. Просмотр общих акций.
6. Просмотр персонализированных акций
7. Просмотр заведений по типу заведения;
   1. Описание алгоритма и функционирования программы
      1. Архитектура приложения

Приложение использует архитектуру viper с использованием паттерна посредник для конфигурирования отдельных модулей программы. View отвечает за отображение данных и обработку пользовательского ввода. Presenter является главным связующим звеном в архитектуре и отвечает за логику его работы. Interactor предоставляет обработку и преобразование данных Entity и связывает Presenter с бизнес-логикой программы. Router отвечает за перемещения между модулями.

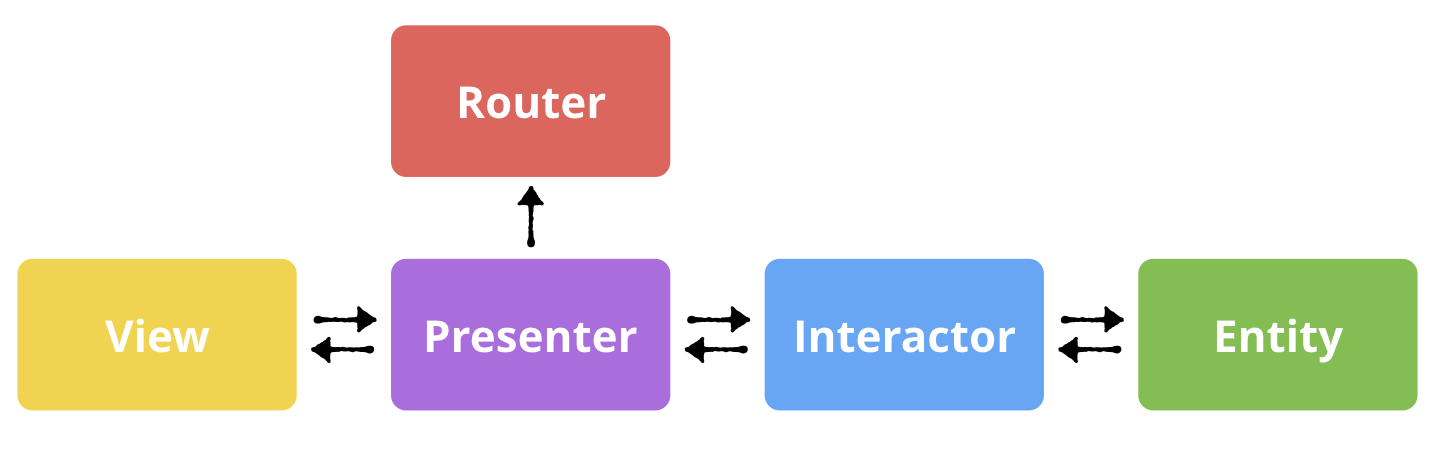


Рисунок 1 - Схема архитектуры Viper

Взаимодействие между компонентами модуля происходит на основе реализаций протоколов входных или выходных методов.

* + 1. Передача данных между модулями

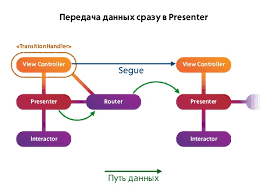
Решение пролемы передачи данных между модулями осуществляется путем передачи их напрямую из роутера в презентер другого модуля через класс, который реализует паттерн посредник в открываемом модуле.

Рисунок 2 - Схема передачи данных между модулями

* + 1. Организация загрузки данных о кафе и их отображение

Загрузка данных о кафе происходила порционно по следующим параметрам: долгота и широта центра видимой области карты на экране и радиус отображения, который равен расстоянию от видимого центра карты до центра верхней точки видимой карты. Загрузка точек происходила при изменении центра видимой области от центра предыдущей на 100 метров.

При загрузке данных о кафе из сети и удалении старых аннотаций на карте происходил небольшой скачок. Решением этого стал алгоритм слияния старой модели представления данных с новой следующим образом: все кафе, которые уже были в старой модели и есть в новой пропускаются и с ними ничего не происходит. Кафе, которые присутствуют в новых данных добавляются в список для отрисовки на карте. Оставшиеся модели добавляются в массив на удаление с карты. Таким образом старые кафе, отображаемые на карте, удалятся, а все новые добавятся, причем кафе которые были в обоих моделях останутся и не произойдет визуальных проблем при отрисовке.

* + 1. Организация хранения данных

Для хранения данных использовались встроенные возможности CoreData - NoSql база данных, со следующими таблицами:

1. User - таблица для хранения информации о текущем пользователе. Поля таблицы:
   1. token - String, Bearer токен для авторизации на сервере;
   2. login - String, Логин пользователя;
   3. password - String, пароль пользователя;
   4. untrackedLocations - Array, позиции пользователя, которые не удалось отправить на сервер.
2. Location - таблица для хранения точек положения пользователя. Поля таблицы:
   1. longitude - Double, долгота положения точки;
   2. latitude - Double, широта положения точки;
   3. date - Date, дата нахождения пользователя в этой точке
      1. Синхронизация положения пользователя с сервером

При перемещении пользователя каждые 100 метров на сервер отсылается POST запрос о новом положении пользователя с данными следующего формата:

{ longitude: Double, latitude: Double, date: Date }

Если запрос оказался неудачным, то эти координаты записываются в локальную базу данных и при следующей успешной попытке отправить данные о положении пользователя, все сохраненные геоточки также будут отправлены на сервер.

* + 1. Проверка данных при регистрации или авторизации

При попытке пользователя авторизоваться или зарегистрироваться перед тем, как отправить данные на сервер происходит проверка почты и пароля на заданный формат. Для почты это проверка на совпадение со следующим паттерном регулярного выражения: [A-Z0-9a-z.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,64}, для пароля это просто проверка на то, что в веденной строке больше 8 символов.

* + 1. Генерация qr-кода для промокода специальной акции

Для генерации изображения qr-кода, который содержит в себе строку промокода текущей акции, использовались возможности встроенной библиотеки CoreImage.CIFilterBuiltins со следующим кодом:

**let** data = Data(string.utf8)

**guard** **let** qrFilter = CIFilter(name: "CIQRCodeGenerator") **else** { **return** **nil** }

qrFilter.setValue(data, forKey: "inputMessage")

**guard** **let** qrImage = qrFilter.outputImage?.transformed(by: CGAffineTransform(scaleX: 10, y: 10)) **else** { **return** **nil** }

**return** UIImage(ciImage: qrImage)

* 1. Обоснование выбора архитектуры приложения

Архитектура Viper была выбрана из-за ряда достоинств:

1. Масштабируемость - такую архитектуру легко поддерживать и развивать по причине разбиения программы на модули, а их в свою очередь на компоненты, которые отвечают за свою логику: View за отображение данных и обработку пользовательского ввода, Presenter - за представление данных, Interactor - за обработку и преобразование данных и взаимодействие с внешними сущностями, Router - за переключение между модулями.
2. Тестируемость - каждый модуль можно тестировать независимо от другого.
3. Личный интерес - хотелось ознакомиться и научиться реализовывать мобильной приложение на основе этой популярной архитектуры

Однако, архитектура также обладает и своими минусами:

1. Большое количество кода для реализации даже самой простой логики.
2. Количество кода и различных подклассов увеличивается очень быстро, что со временем увеличивает время поддержки и разработки модуля
3. Разделение абстракций работы компонент модуля не всегда очевидно
   1. Описание и обоснование выбора технических и программных средств
      1. Состав технических и программных средств

Для работы программы необходимо устройство Apple iPhone версии не ниже 7 или 5se, а также Apple iPod 7 с установленной операционной системой iOS версии не ниже 14.

* + 1. Обоснование выбора технических и программных средств

1. Для разработки использовался язык Swift вместе с фреймворком UIKit и встроенными возможности языка. Выбор языка не в пользу Objective-C заключается в том, что Swift является более современным, популярным и простым языком для разработки. Также не был выбран SwiftUI по причине его молодости и малой используемости в больших проектах.
2. Для взаимодействия с сетью был использован встроенный функционал URLSession по причине возможности настроить конфигурацию запроса и отправлять их по протоколу http.
3. Для генерации qr-кода использовалась встроенная библиотека CoreImage.CIFilterBuiltins, чтобы не загружать картинку с сервера, тем самым уменьшив затрачиваемые ресурсы.
4. ожидаемые технко-экономические показатели
   1. Предполагаемая потребность

Компании-рекламодатели смогу привлечь новых покупателей благодаря созданию персонализированных рекламных предложений, которые дают плюсы обоим сторонам: пользователи-рекламодатели находят новых клиентов, а клиенты получают индивидуальные акционные предложения.

* 1. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

На зарубежном рынке существуют приложения "Instagram", "Facebook", "Google Maps", "Apple Maps ", обладающие подобным функционалом: https://www.instagram.com/, https://www.facebook.com/, https://www.google.ru/maps/, https://www.apple.com/ru/maps/.

На отечественном рынке существуют интернет ресурсы "Вконтакте", "Яндекс.Карты", "Яндекс.Еда", "2ГИС", "Деливери-клаб" с подобным функционалом: https://vk.com/ , https://yandex.ru/maps/ , https://eda.yandex.ru/ , https://2gis.ru/ , https://www.delivery-club.ru/ .

Приложение обладает преимуществами:

1. Позволяет пользователям получать как общие, так и индивидуальные акционные предложения;
2. Позволяет компаниям-рекламодателям создавать как общие, так и индивидуальные акционные предложения;
3. Позволяет компаниям-рекламодателям просматривать статистику по общим и индивидуальным рекламным предложениям;
4. Подбор индивидуальных предложений происходит на основе каждодневных маршрутов пользователя.
5. Список литературы
   * 1. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
     2. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
     3. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
     4. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению. – М.: ИПК Издательство стандартов, 2001.
     5. ГОСТ 19.404-79. ЕСПД. Пояснительная записка. Требования к содержанию и оформлению. – М.: ИПК Издательство стандартов, 2001.
     6. ГОСТ 19.505-79. ЕСПД. Руководство оператора. Требования к содержанию и оформлению. . – М.: ИПК Издательство стандартов, 2001.
     7. Документация по UIKit [Электронный ресурс]// URL: https://developer.apple.com/documentation/uikit (Дата обращения: 5.05.2021, режим доступа: свободный)
     8. Документация по Swift [Электронный ресурс]// URL: https://swift.org/documentation/ (Дата обращения: 5.05.2021, режим доступа свободный)
     9. The book of Viper [Электронный ресурс]// URL: https://github.com/strongself/The-Book-of-VIPER/blob/master/russian/contents.md (Дата обращения: 12.02.2021, режим доступа свободный)

# **ПРИЛОЖЕНИЕ 1**

# **ТЕРМИНОЛОГИЯ**

**Http** - протокол передачи данных.

**Авторизация** - предоставление определённому лицу или группе лиц определённых прав.

**База данных** - организованная структура, предназначенная для хранения, изменения и обработки взаимосвязанной информации

**Пользователь** - человек, который использует данную программу

**Сервер** - компонент вычислительной системы, выполняющий функции по запросу клиента, предоставляя ему доступ к определенным ресурсам

**Клиент** - программа, работающая на устройстве пользователя и позволяющая получать и отправлять информацию на сервер и предоставлять ее пользователю в интерактивном виде

**Qr-код** - тип матричных штрихкодов

**UIKit** - фреймворк для создания приложений под iPhone, iPad и iPod

**Модуль** - отдельная часть программы, относящаяся к слою представления, которая реализует архитектуру Viper и предоставляется реализацией набором компонент (View, Presenter, Router, Interactor, Configurator).

**Компонент модуля** - набор input и output протоколов и класс, который их реализует, из которых строится viper модуль.

**View** - компонент модуля, который отвечает за представление данных пользователю и обработку пользовательского ввода.

**Presenter** - компонент модуля, который отвечает за логику работы модуля и обработки данных, полученных из View и Interactor.

Interactor - компонент модуля, который отвечает за связь модуля с бизнес-логикой программы и преобразованием полученных оттуда данных для их дальнейшей передачи в Presenter.

**Router** - компонент модуля, который отвечает за переключение между модулями.

**CoreData** - фреймворк, встроенный в операционную систему iOS, который предоставляет возможность взаимодействия SQLite, бинарной или xml базой данных.

# **ПРИЛОЖЕНИЕ 2**

# **ПРЕДСТАВЛЕНИЕ ТАБЛИЦ БАЗЫ ДАННЫХ**

|  |  |
| --- | --- |
| Название таблицы | Представление таблицы |
| User |  |
| Location |  |

# **ПРИЛОЖЕНИЕ 3**

# **ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

|  |  |
| --- | --- |
| Класс | Назначение |
| AccountConfigurator | Конфигуратор модуля аккаунта |
| AccountInteractor | Интерактор аккаунта |
| AccountPresenter | Презентер аккаунта |
| AccountRouter | Роутер аккаунта |
| AccountViewController | Контроллер аккаунта |
| AuthorizationConfigurator | Конфигуратор авторизации |
| AuthorizationInteractor | Интерактор авторизации |
| AuthorizationPresenter | Презентер авторизации |
| AuthorizationRouter | Роутер авторизации |
| AuthorizationViewController | Контроллер авторизации |
| CoreData | Сервис взаимодействия с CoreData |
| ImageLoader | Загрузчик фотографий из сети |
| InputTextField | Поле ввода текста |
| Location | Позиция пользователя |
| LocationManager | Сервис позиции пользователя |
| LoginEntryChecker | Объект для проверки введенных данных по формату |
| MainViewController | TabBar контроллер для карты и авторизации |
| MapConfigurator | Конфигуратор модуля карты |
| MapInteractor | Интерактор карты |
| MapPresenter | Презентер карты |
| MapRestaurantViewModel | Модель представления кафе на карте |
| MapRouter | Роутер карты |
| MapViewController | Контроллер карты |
| MapViewModel | Модель представления карты |
| NextButton | Кнопка со стрелкой |
| RegistrationConfigurator | Конфигуратор модуля |
| RegistrationInteractor | Интерактор регистрации |
| RegistrationPresenter | Презентер регистрации |
| RegistrationRouter | Роутер регистрации |
| RegistrationViewController | Контроллер регитсрации |
| RequestFactory | Фабрика запросов |
| Restaurant | Модель кафе |
| RestaurantAnnotation | Аннотация кафе на карте |
| RestaurantConfigurator | Конфигуратор модуля кафе |
| RestaurantHeaderView | Вью информации о кафе |
| RestaurantInteractor | Интерактор кафе |
| RestaurantModel | Модель кафе |
| RestaurantPresenter | Презентер кафе |
| RestaurantSaleModel | Модель акции кафе |
| RestaurantTypeCell | Ячейка типа ресторана |
| RestaurantViewController | Контроллер кафе |
| RestaurantViewModel | Модель представления данных кафе |
| Sale | Модель акции |
| SaleCell | Ячейка таблицы с акцией |
| SaleViewModel | Модель представления данных акции |
| SpecialSaleCollectionViewCell | Ячейка специальной акции |
| TitledTextField | Поле ввода с заголовком |
| User | Модель пользователя |
| UserModel | Модель пользователя |
| UserService | Сервис пользователя |
| UserViewModel | Модель представления пользователя |
| Endpoints | Маппинг url сервера |
| RestaurantType | Перечисление типов ресторана |

# **ПРИЛОЖЕНИЕ 4**

# **ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ КЛИЕНТА**

Описание полей, методов и свойств класса AccountConfigurator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| assembly | internal | AccountViewController |  | Создает зависимости между компонентами модуля |

Описание полей, методов и свойств класса AccountInteractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| userService | internal | UserService | Сервис пользователя | |
| presenter | internal | AccountPresenterInput | Презентер аккаунта | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | with presenter: AccountPresenterInput | Конструктор |
| logoutUser | internal | Void |  | Выйти из аккаунта пользователя |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| currentUser | internal | UserModel | get | Текущий пользователь |

Описание полей, методов и свойств класса AccountPresenter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | AccountViewInput | Контроллер аккаунта | |
| interactor | internal | AccountInteractor | Интерактор аккаунта | |
| router | internal | AccountRouterInput | Роутер аккаунта | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | with view: AccountViewInput | Конструктор с контроллером |
| logoutSuccess | internal | Void |  | Метод успешного выхода из аккаунта |
| logoutTapped | internal | Void |  | Событие нажатия кнопки выхода |
| viewDidLoad | internal | Void |  | Контроллер загрузился |

Описание полей, методов и свойств класса AccountRouter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | with view: AccountViewController | Конструктор с контроллером |
| popBack | internal | Void |  | Вернуться в корневой вью |

Описание полей, методов и свойств класса AccountViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| logoutButton | private | UILabel | Кнопка выхода из аккаунта | |
| emailTitle | private | UILabel | Заголовок поля почты | |
| userEmail | private | UILabel | Текст почты пользователя | |
| output | internal | AccountViewOutput | Presenter модуля | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLoad | internal | Void |  | Контроллер загружен |
| viewDidAppear | internal | Void | animated: Bool | Контроллер появился на экране |
| configure | internal | void | with viewModel: UserViewModel | Сконфигурировать контроллер по модели данных пользователя |

Описание полей, методов и свойств класса AuthorizationConfigurator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| configure | internal | Void | view: AuthorizationViewController | Конфигурировать модуль авторизации |

Описание полей, методов и свойств класса AuthorizationInteractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| presenter | internal | AuthorizationInteractorOutputProtocol | Презентер авторизации | |
| userService | private | UserService | Сервис пользователя | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | presenter: AuthorizationInteractorOutputProtocol | Конструктор |
| authUser | internal | Void | withEmail: String, password: String | Авторизовать пользователя |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| isUserAuth | internal | Bool | get | Авторизован ли пользователь |

Описание полей, методов и свойств класса AuthorizationPresenter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | AuthorizationViewOutput | Контроллер авторизации | |
| interactor | internal | AuthorizationInteractorProtocol | Интерактор авторизации | |
| router | internal | AuthorizationRouterProtocol | Роутер авторизации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: AuthorizationViewProtocol | Конструктор |
| viewDidLoad | internal | Void |  | Контроллер загрузился |
| authorizationButtonTapped | private | Void | withEmail: String, password: String | Событие нажатия на кнопку авторизации |
| registrationButtonTapped | private | Void |  | Событие нажатия на кнопку регистрации |
| authorizationUnsuccessfully | internal | Void |  | Авторизация прошла неуспешно |
| authorizationSuccessfully | internal | Void |  | Авторизация прошла успешно |

Описание полей, методов и свойств класса AuthorizationRouter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | AuthorizationViewProtocol | Контроллер авторизации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: AuthorizationViewController | Конструктор |
| openRegistrationView | internal | Void |  | Открыть модуль регистрации |
| openAccountView | internal | Void | animated: Bool | Открыть модуль аккаунта |

Описание полей, методов и свойств класса AuthorizationViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| configurator | private | AuthorizationConfigurator | Конфигуратор модуля | |
| presenter | internal | AuthorizationPresenterProtocol | Презентер авторизации | |
| backgroundImage | private | UIImageView | Картинка фона | |
| emailTextField | private | TitledTextField | Поле ввода почты | |
| passwordTextField | private | TitledTextField | Поле ввода пароля | |
| authorizationButton | private | NextButton | Кнопка авторизации | |
| registrationButton | private | UIButton | Кнопка открытия модуля регистрации | |
| activityIndicator | private | UIActivityIndicatorView | Индикатор загрузки | |
| subtitleLabel | private | UILabel | Текст подзаголовка | |
| titleLabel | private | UILabel | Текст заголовка | |
| smallSubtitleLabel | private | UILabel | 2 уровень подзаголовка | |
| footerLabel | private | UILabel | Нижняя подпись | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLoad | internal | Void |  | Контроллер загрузился |
| viewDidAppear | internal | Void | animated: Bool | Контроллер отобразился |
| startAnimatingActivityIndicator | internal | Void |  | Начать анимацию индикатора загрузки |
| stopAnimatingActivityIndicator | internal | Void |  | Завершить анимацию индикатора загрузки |
| showAlert | internal | Void | title: String, message: String | Показать сообщение |
| setNavigationBarHidden | internal | Void | isHidden: Bool, animated: Bool | Изменить видимость навигации |
| getNavigationController | internal | UINavigationController? |  | Получить контроллер навигации |

Описание полей, методов и свойств класса CoreData

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| shared | internal | CoreData | Общий экземпляр типа | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| loadUser | internal | User? |  | Получить сохраненного пользователя |
| loadRestaurants | internal | [Restaurant]? |  | Получить сохраненные кафе |
| saveData | internal | Void |  | Сохранить изменения |
| loadUntrackedLocations | internal | [Location]? |  | Получить сохраненные позиции пользователя |
| removeObject | internal | Void | object: NSManagedObject | Удалить объект из база |
| clearUser | internal | Void |  | Удалить текущего пользователя |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| viewContext | internal | NSManagedObjectContext | get | Контекст CoreData |

Описание полей, методов и свойств класса ImageLoader

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| loadRestaurantImage | internal | Void | restId: Int32, completion: @escaping (\_ data: Data?) -> () | Загрузить картинку для кафе |
| loadSaleImage | internal | Void | saleId: Int32, completion: @escaping (\_ data: Data?) -> () | Загрузить картинку для акции |

Описание полей, методов и свойств класса InputTextField

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| leftPadding | internal | CGFloat | Левый отступ | |
| gapPadding | internal | CGFloat | Дополнительный отступ от картинки | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| leftViewRect | private | CGRect | bounds: CGRect | Область начала для текста |
| init | internal | init |  | Конструктор класса |
| init | internal | required init? | coder: NSCoder | Конструктор класса |
| textRect | open | CGRect | bounds: CGRect | Размеры текста |
| placeholderRect | open | CGRect | bounds: CGRect | Размеры placeholder |
| editingRect | open | CGRect | bounds: CGRect | Размеры редактируемого текста |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| color | internal | UIColor? | get set | Цвет placeholder |
| leftImage | internal | UIImage? | get set | Картинка слева |

Описание полей, методов и свойств класса Location

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| fetch | public | NSFetchRequest<Location> |  | Запрос на получение объектов |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| latitude | public | Double | get set | Широта |
| longitude | public | Double | get set | Долгота |
| date | public | Date | get set | Дата, когда пользователь был на позиции |
| user | public | User? | get set | Пользователь |

Описание полей, методов и свойств класса LocationManager

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| shared | internal | LocationManager | Общий объект сервиса | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| userDistance | internal | Double | location: CLLocation | Расстояние от позиции пользователя до геоточки |

Описание полей, методов и свойств класса LoginEntryChecker

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| checkEmail | internal | Bool | email: String | Проверка формата строки для почты |
| checkPassword | internal | Bool | password: String | Провекра формата строки с паролем |

Описание полей, методов и свойств класса MainViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLoad | internal | Void |  | Вью загрузилось |
| viewDidLayoutSubviews | internal | Void |  | Перерисовка tabbar |

Описание полей, методов и свойств класса MapConfigurator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| configure | internal | Void | view: MapViewController | Конфигурировать модуль по контроллеру |

Описание полей, методов и свойств класса MapInteractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | presenter: MapInteractorOutput, service: UserService | Конструктор |
| loadRestaurants | internal | Void | in region: CLLocationCoordinate2D, radius: Double | Загрузить кафе |
| sendUserLocationUpdate | internal | Void | location: CLLocation | Отправить изменение позиции пользователя |

Описание полей, методов и свойств класса MapPresenter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| interactor | internal | MapInteractorInput! | Интерактор кафе | |
| router | internal | MapRouterProtocol! | Роутер кафе | |
| restaurants | private | [RestaurantModel] | Все кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: MapViewController | Конструктор |
| regionDidChange | internal | Void | region: MKCoordinateRegion, radius: Double | Регион, отображаемый на карте, изменен |
| annotationTapped | internal | Void | annotation: MKAnnotationView | Событие нажатия на аннотацию |
| updateUserLocation | internal | Void | locations: [CLLocation] | Обновить позицию пользователя если она изменилась на 100 метров |
| changeFilter | internal | Void | index: Int, state: Bool | Событие изменения выбранных фильров |
| viewDidAppear | internal | Void |  | Контроллер отобразился |
| restaurantsLoad | internal | Void | restaurants: [RestaurantModel] | Кафе загружены |

Описание полей, методов и свойств класса MapRestaurantViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| id | internal | Int32 | id кафе | |
| image | internal | UIImage | Картинка кафе | |
| annotation | internal | MKPointAnnotation | Аннотация кафе | |
| type | internal | RestaurantType | Тип кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: RestaurantModel | Конструктор из модели кафе |

Описание полей, методов и свойств класса MapRouter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: MapViewController | Конструктор |
| openRestaurantView | internal | Void | restaurant: RestaurantModel | Открыть модуль кафе |

Описание полей, методов и свойств класса MapViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| configurator | internal | MapConfiguratorProtocol | Конфигуратор модуля карты | |
| lastLocation | internal | CLLocation? | Последняя позиция пользователя | |
| presenter | internal | MapPresenterInput! | Презентер карты | |
| mapView | private | MKMapView | Карта | |
| locationManager | private | LocationManager | Сервис положения пользователя | |
| viewModel | private | MapViewModel? | Модель представления карты | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLoad | internal | Void |  | Контроллер загрузился |
| viewDidAppear | internal | Void | animated: Bool | Контроллер отобразился |
| changeFollowMode | private | Void |  | Изменить способ следования за пользователем |
| configure | internal | Void | vm: MapViewModel | Конфигурировать по модели представления карты |
| mapView | internal | Void | mapView: MKMapView, regionDidChangeAnimated animated: Bool | Событие изменения отображаемого региона на карте |
| mapView | internal | MKAnnotationView? | mapView: MKMapView, viewFor annotation: MKAnnotation | Сконфигурировать вью для аннотации |
| mapView | internal | Void | mapView: MKMapView, didSelect view: MKAnnotationView | Событие клика на аннотацию |
| mapView | internal | Void | mapView: MKMapView, didChange mode: MKUserTrackingMode, animated: Bool | Событие изменения способа следования за пользователем |
| locationManager | internal | Void | manager: CLLocationManager, didUpdateLocations locations: [CLLocation] | Событие изменения позиции пользователя |
| locationManagerDidChangeAuthorization | internal | Void | manager: CLLocationManager | Событие изменения доступа к отслеживанию позиции пользователя |
| collectionView | internal | Int | collectionView: UICollectionView, numberOfItemsInSection section: Int | Вычисление количества ячеек в коллекции |
| collectionView | internal | UICollectionViewCell | collectionView: UICollectionView, cellForItemAt indexPath: IndexPath | Конфигурация ячейки коллекции |
| collectionView | internal | Void | collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath | Событие выбора ячейки коллекции |
| collectionView | internal | CGSize | collectionView: UICollectionView, layout collectionViewLayout: UICollectionViewLayout, sizeForItemAt indexPath: IndexPath | Вычисление размера ячейки коллекции |

Описание полей, методов и свойств класса MapViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| mapRestaurants | internal | [MapRestaurantViewModel] | Кафе, которые необходимо добавить на карту | |
| deletedRestaurants | internal | [MapRestaurantViewModel] | Кафе, которые необходимо удалить с карты | |
| allRests | internal | [MapRestaurantViewModel] | Все кафе | |
| deleteAll | internal | Bool | Необходимо ли удалить все кафе с карта | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | restaurants: [RestaurantModel], oldModel: MapViewModel? | Конструктор слияния старой модели с новыми кафе |
| updateForTypes | internal | Void | types: [RestaurantType] | Обновить кафе по фильтру |

Описание полей, методов и свойств класса NextButton

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init |  | Конструктор класса |
| init | internal | required init? | coder: NSCoder | Конструктор класса |
| layoutSubviews | internal | Void |  | Пересчитать размеры и положение объектов |

Описание полей, методов и свойств класса RegistrationConfigurator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| configure | internal | Void | view: RegistrationViewController | Конфигурировать модуль регистрации |

Описание полей, методов и свойств класса RegistrationInteractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| presenter | internal | RegistrationPresenterOutput! | Презентер регистрации | |
| userService | internal | UserService | Сервис пользователя | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | presenter: RegistrationPresenterOutput | Конструктор |
| registerUser | internal | Void | withEmail: String, password: String | Зарегистрировать пользователя |

Описание полей, методов и свойств класса i

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | RegistrationViewOutput! | Контроллер регистрации | |
| interactor | internal | RegistrationInteractorInput! | Интерактор регистрации | |
| router | internal | RegistrationRouterInput! | Роутер регистрации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: RegistrationViewOutput | Конструктор |
| viewDidAppear | internal | Void |  | Контроллер отобразился |
| registrationButtonTapped | internal | Void | withEmail: String, password: String, passwordRepeat: String | Кнопка регистрации нажата |
| backButtonTapped | internal | Void |  | Событие нажатия кнопки возврата |
| registrationSuccessfully | internal | Void |  | Регистрация прошла успешно |
| registrationUnsuccessfully | internal | Void |  | Регистрация не удалась |

Описание полей, методов и свойств класса RegistrationRouter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | RegistrationViewOutput | Контроллер регистрации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | required init | view: RegistrationViewOutput | Конструктор |
| popBack | internal | Void |  | Вернуться на прошлый модуль |
| openAccountView | internal | Void |  | Открыть модуль аккаунта |

Описание полей, методов и свойств класса RegistrationViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| configurator | internal | RegistrationConfiguratorProtocol | Конфигуратор модуля | |
| presenter | internal | RegistrationPresenterInput | Презентер регистрации | |
| emailTextField | private | TitledTextField | Поле ввода почты | |
| passwordTextField | private | TitledTextField | Поле ввода пароля | |
| repeatPasswordTextField | private | TitledTextField | Поле повторного ввода пароля | |
| registrationButton | private | UIButton | Кнопка регистрации | |
| backButton | private | UIButton | Кнопка возвращения на экран авторизации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLoad | internal | Void |  | Контроллер загружен |
| viewDidAppear | internal | Void | animated: Bool | Контроллер отобразился |
| registrationButtonTapped | internal | Void |  | Событие нажатия на кнопку регистрации |
| backButtonTapped | internal | Void |  | Событие нажатия на кнопку выхода из вью |
| setNavigationBarHidden | internal | Void | isHidden: Bool, animated: Bool | Поставить видимость навигации |
| showAlert | internal | Void | title: String, message: String | Показать сообщение |
| getNavigationController | internal | UINavigationController? |  | Получить контроллер навигации |

Описание полей, методов и свойств класса RequestFactory

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| authRequest | internal | URLRequest | form: LoginForm | Запрос авторизации пользователя |
| registrationRequest | internal | URLRequest | form: LoginForm | Запрос регистрации пользователя |
| restaurantSalesRequest | internal | URLRequest | with token: String, restaurantId: Int32 | Запрос получения акций кафе |
| restaurantsRequest | internal | URLRequest | token: String, coordinate: CoordinateRequestModel | Запрос кафе рядом с позицией |
| locationRequest | internal | URLRequest | token: String, longitude: Double, latitude: Double, date: Date | Запрос обновления позиции пользователя |
| loadRestaurantImage | internal | URLRequest | restaurantId: Int32 | Запрос загрузки картинки кафе |
| loadSaleImage | internal | URLRequest | saleId: Int32 | Запрос загрузки картинки акции |

Описание полей, методов и свойств класса RestaurantAnnotaition

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| restaurant | internal | MapRestaurantViewModel! | Кафе для аннотации | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | annotation: MKAnnotation?, reuseIdentifier: String? | Конструктор |
| configure | internal | Void | vm: MapRestaurantViewModel | Конфигурировать аннотацию по модели представления кафе |
| init | internal | required init? | aDecoder: NSCoder | Конструктор |

Описание полей, методов и свойств класса RestaurantConfigurator

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| assembly | internal | RestaurantViewController | restaurant: RestaurantModel) | Конфигурировать модуль |

Описание полей, методов и свойств класса RestaurantHeaderView

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init |  | Конструктор |
| init | internal | init | frame: CGRect | Конструктор |
| init | internal | required init? | coder: NSCoder | Конструктор |
| layoutSubviews | internal | Void |  | Перерисовать вью |
| configure | internal | Void | vm: RestaurantViewModel | Конфигурировать вью по модели представления кафе |

Описание полей, методов и свойств класса RestaurantInteractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| userService | internal | UserService | Сервис пользователя | |
| output | internal | RestaurantInteractorOutput! | Презентер кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | output: RestaurantInteractorOutput | Конструктор |
| loadSales | internal | Void | restId: Int32 | Загрузить акции |

Описание полей, методов и свойств класса RestaurantModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| id | internal | Int32 | id кафе | |
| longitude | internal | String | долгота | |
| latitude | internal | Double | широта | |
| type | internal | RestaurantType | Тип кафе | |
| location | internal | String | Адрес кафе | |
| sales | internal | [RestaurantSaleModel]? | Акции кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: Restaurant | Конструктор из модели кафе для CoreData |
| init | internal | init |  | Конструктор без параметров |

Описание полей, методов и свойств класса RestaurantPresenter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| view | internal | RestaurantViewController! | Контроллер кафе | |
| interactor | internal | RestaurantInteractorInput! | Интерактор кафе | |
| currentRestaurant | internal | RestaurantModel | Текущее кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | view: RestaurantViewController, currentRestaurant: RestaurantModel | Конструктор |
| viewDidLoad | internal | Void |  | Контроллер загрузился |
| loadedSales | internal | Void | sales: [RestaurantSaleModel] | Акции кафе загружены |

Описание полей, методов и свойств класса RestaurantSaleModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| id | internal | Int32 | id акции | |
| name | internal | String | Название акции | |
| promo | internal | String | Промокод акции | |
| image | internal | UIImage? | Картинка акции | |
| oldPrice | internal | Double | Изначальная цена товара | |
| newPrice | internal | Double | Новая цена по акции | |
| special | internal | Bool | Является ли акция специальной | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init |  | Конструктор класса |
| init | internal | init | model:Sale | Конструктор из модели акции для CoreData |

Описание полей, методов и свойств класса RestaurantTypeCell

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| selectionMode | internal | Bool | Выбран ли тип | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | frame: CGRect | Конструктор |
| layoutSubviews | internal | Void |  | Перерисовать вью |
| configure | internal | Void | type: RestaurantType | Конфигурировать ячейку по типу кафе |
| init | internal | required init? | code: NSCoder | Конструктор |

Описание полей, методов и свойств класса RestaurantViewController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| isDragging | private | Bool | Взаимодействует ли пользователь с скроллом | |
| isDecelerating | private | Bool | Скроллится ли скролл | |
| lastSalesTableScrollY | private | CGFloat | Последняя позиция контента скролла | |
| presenter | internal | RestaurantViewControllerOutput! | Презентер кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| viewDidLayoutSubviews | internal | Void |  | Перерисовать вью |
| viewDidLoad | internal | Void |  | Контроллер загрузился |
| configureSubviews | private | Void |  | Конфигурация отображения внутренних вью |
| addAllSubviews | private | Void |  | Добавить все вью в контроллер |
| initConstraints | private | Void |  | Активировать констреинты |
| configure | internal | Void | vm: RestaurantViewModel | Сконфигурировать вью по модели данных кафе |
| tableView | internal | UITableViewCell | tableView: UITableView, cellForRowAt indexPath: IndexPath | Создание ячейки для таблицы |
| tableView | internal | Int32 | tableView: UITableView, numberOfRowsInSection section: Int | Количество ячеек |
| tableView | internal | CGFloat | tableView: UITableView, heightForRowAt indexPath: IndexPath | Вычисление высоты для ячейки таблицы |
| tableView | internal | UIView? | tableView: UITableView, viewForHeaderInSection section: Int | Создание вью с заголовком для таблицы |
| tableView | internal | CGFloat | tableView: UITableView, heightForHeaderInSection section: Int | Высота вью с заголовком |
| collectionView | internal | Int32 | collectionView: UICollectionView, numberOfItemsInSection section: Int | Вычисление количества ячеек в коллекции |
| collectionView | internal | UICcollectionViewCell | collectionView: UICollectionView, cellForItemAt indexPath: IndexPath | Конфигурация ячейки для коллекции |
| getCurrentPage | private | Int |  | Индекс отображаемой ячейки коллекции |
| scrollViewDidEndDragging | internal | Void | scrollView: UIScrollView, willDecelerate decelerate: Bool | Событие завершения скролла |
| scrollViewDidEndDecelerating | internal | Void | scrollView: UIScrollView | Скролл завершил пролистывание |
| scrollViewDidScroll | internal | Void | scrollView: UIScrollView | Скролл скролится пользователем |
| scrollViewWillEndDragging | internal | Void | scrollView: UIScrollView, withVelocity velocity: CGPoint, targetContentOffset: UnsafeMutablePointer<CGPoint> | Скролл закончил касание пользователя |

Описание полей, методов и свойств класса RestaurantViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| name | internal | String | Названия кафе | |
| address | internal | String | Адрес кафе | |
| distance | internal | Double | Расстояние до кафе | |
| typeImage | internal | UIImage? | Картинка типа акции | |
| logoImage | internal | UIImage? | Логотип кафе | |
| sales | internal | [SaleViewModel] | Акции кафе | |
| specialSales | internal | [SaleViewModel] | Специальные кафе | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: RestaurantModel, allSales: [RestaurantSaleModel]) | Конструктор |

Описание полей, методов и свойств класса SaleCell

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| backImage | private | UIImage | Задняя картинка для акции | |
| frontView | private | UIView | Внешняя вью | |
| saleName | private | UILabel | Название акции | |
| saleCode | private | UILabel | Промокод акции | |
| salePrice | private | UILabel | Цена акции | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | style: UITableViewCell.CellStyle, reuseIdentifier: String? | Конструктор |
| init | internal | required init? | code: NSCoder | Конструктор |
| awakeFromNib | internal | Void |  | Вью загрузилось из storyboard |
| layoutSubviews | internal | Void |  | Перерисовать вью |
| configure | internal | Void | vm: SaleViewModel | Сконфигурировать вью по модели представления акции |

Описание полей, методов и свойств класса SaleViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| name | internal | String | Имя акции | |
| price | internal | Double | Цена акции | |
| code | internal | String | Промокод акции | |
| image | internal | UIImage? | Картинка акции | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: RestaurantSaleModel | Конструктор из модели акции |

Описание полей, методов и свойств класса SpecialSaleCollectionViewCell

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | frame: CGRect | Конструктор |
| init | internal | required init? | coder: NSCoder | Конструктор |
| layoutSubviews | internal | Void | Void | Отрисовать внутренние вью |
| configure | internal | Void | Void | Сконфигурировать вью по модели представления акции |

Описание полей, методов и свойств класса TitledTextField

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| color | internak | UIColor? | Цвет заголовка и поля ввода | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init |  | Конструктор |
| init | internal | required init? | code: NSCoder | Конструктор |
| layoutSubviews | internal | Void |  | Инициализация констреинтов |
| emialTextField | internal | TitledTextField |  | Фабрика поля ввода для почты |
| passwordTextField | internal | TitledTextField |  | Фабрика поля ввода для пароля |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| titleText | internal | String? | get set | Текст заголовка |
| text | internal | String | get | Введенный текст |
| isSecure | internal | Bool | get set | Является ли текстовое поле защищенным |
| textFieldImage | internal | UIImage? | get set | Картинка поля ввода |
| placeholder | internal | String? | get set | Placeholder поля ввода |

Описание полей, методов и свойств класса User

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| fetch | internal | NSFetchRequest<User> |  | Запрос для получения объектов |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| token | internal | String? | get set | Токен |
| login | internal | String | get set | Почта |
| password | internal | String | get set | Пароль |
| restaurants | internal | NSSet? | get set | Кафе |
| untrackedLocations | internal | NSSet? | get set | Неотправленные позиции |

Описание полей, методов и свойств класса UserModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| token | internal | String | Токен авторизации пользователя | |
| login | internal | String | Почта пользователя | |
| password | internal | String | Пароль польователя | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: User | Инициализация из модели пользователя из CoreData |

Описание полей, методов и свойств класса UserService

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| shared | internal | UserService | Общий объект сервиса | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| authUser | internal | Void | form: LoginForm, completion: @escaping (\_ isSuccess: Bool) -> () | Авторизировать пользователя |
| registerUser | internal | Void | form: LoginForm, completion: @escaping (\_ isSuccess: Bool) -> () | Регистрация пользователя |
| getRestaurantsNear | internal | Void | coordinate: CoordinateRequestModel, completion: @escaping ([RestaurantModel]?) -> () | Получить кафе рядом с положением пользователя |
| getRestaurantSales | internal | Void | restaurantId: Int32, completion: @escaping (\_ sales: [RestaurantSaleModel]?) -> () | Получить акции кафе |
| updateUserLocation | internal | Void | latitude: Double, longitude: Double | Отправить обновление позиции пользователя |
| logout | internal | Void | completion: (\_ isSuccess: Bool) -> () | Выйти из аккаунта пользователя |
| **Свойства** | | | | |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| isUserAuth | internal | Bool | get | Авторизован ли пользователь |
| currentUser | internal | User? | get set | Текущий пользователь |

Описание полей, методов и свойств класса UserViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля** | | | | |
| Имя | Модификатор доступа | Тип | Назначение | |
| login | internal | String | Логин пользователя | |
| **Методы** | | | | |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | internal | init | model: UserModel | Конструктор из модели пользователя |

1. Лист регистрации изменений

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |