**МУНИЦИПАЛЬНОЕ БЮДЖЕТНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**
**ОДИНЦОВСКАЯ СРЕДНЯЯ ОБЩЕОБРАЗОВАТЕЛЬНАЯ ШКОЛА №1**
(143000, Московская область, г. Одинцово, ул. Солнечная, д.14)
тел. (495)593-69-24

ИТОГОВАЯ ПРОЕКТНАЯ РАБОТА

по информатике

# «Разработка мобильных приложений. Приложение для планирования поездок»

Автор:

Лебеденко Вадим Евгеньевич, 9-Г

Руководитель:

Тепаносян Эдуард Гомеросович,

учитель информатики

МБОУ Одинцовской СОШ №1

Одинцово

2023

**Паспорт проекта**

**Название проекта:** “Разработка мобильных приложений. Приложение для планирования поездок”.

**Автор:** Лебеденко Вадим Евгеньевич.

**Научный руководитель:** Тепаносян Эдуард Гомеросович.

**Предмет проекта:** информатика.

**Объект проекта:** мобильное приложение.

**Проблема:** создание плана поездки.

**Цель работы:** создать мобильное приложение для планирования поездок.

**Задачи:**

1. Изучить предметную область;
2. Изучить тему создания мобильных приложений;
3. Разработать программный код для приложения и серверной части;
4. Выпустить тестовую версию приложения;
5. Провести тестирование на ограниченной аудитории;
6. Выпустить приложение в общий доступ.

**Этапы проекта:**

1. Подготовительный этап (сентябрь – октябрь) – определение темы, поиск и изучение необходимой информации.
2. Аналитический этап (октябрь – декабрь) – просмотр собранной информации, выделение нужного.
3. Практический этап (с декабря) – окончательная обработка данных и осуществление проекта.

**Продукт проекта:** мобильное приложение, в котором можно создать план своей поездки.

**Методы:** описательный, иллюстративный, анализ, обобщение, систематизация информации, создание итогового приложения**.**

**Оборудование:** компьютер.

**Содержание**

## Введение

Тема проекта – разработка мобильных приложений. Сейчас без мобильных приложений не обойдется ни один современный человек. Они решают самые разные задачи – от просмотра коротких видео до управления финансами. Для своего проекта я выбрал продукт – приложение, которое будет помогать человеку запланировать свой отпуск, отдых или любую другую поездку.

Цель моего проекта – создать мобильное приложение, в котором можно создать план поездки и использовать его.

Задачи проекта:

1. Изучить, как разрабатываются мобильные приложения.
2. Изучить тему приложения, т.е. понять, чем оно будет полезно людям. На основе этих данных продумать функционал приложения.
3. Познакомиться с понятием бизнес-логика и продумать её для моего проекта.

4. Изучить фреймворк для создания мобильных приложений Flutter.
5. Написать приложение для планирования поездок с помощью Flutter.
6. Написать сервер, который будет работать вместе с приложением.
7. Провести тестирование на ограниченной аудитории.
8. Выложить приложение в общий доступ.

## Теоретическая часть

### Ожидаемые результаты

Успешным результатом в моём проекте будет считаться выполнение следующих критериев:

- Полностью рабочее мобильное приложение (для платформы Android и IOS), в котором реализовано планирование поездки, выбор активностей, внос расходов.
- Полностью рабочий сервер, который будет выполнять запросы с мобильного приложения.
- Удобный, понятный и красивый дизайн приложения.

### Целевая аудитория

Одна из главных частей любого глобального проекта – это выбор целевой аудитории, ведь чтобы продумать все аспекты приложения, нужно знать о его пользователях. В моем случае это все люди, которые любят где-либо отдыхать и хотят заранее знать, что они могут поделать на отдыхе.

### Разработка приложения

Для разработки приложения я выбрал фреймворк[1] Flutter, код для которого пишется на языке Dart. С помощью него можно создавать большие и многофункциональные приложения сразу для двух платформ – IOS и Android, что значительно облегчает разработку.

*Фреймворк[1] - программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.*

Приложение будет включать в себя следующие функции:

1. Создание поездки. Здесь можно: запланировать место, дату, количество людей.
2. Создание плана активностей[2]. Здесь можно найти любое нужно место и добавить его в свой план.
3. Запланировать дату и время посещения мест.
4. Возможность просмотра фото и отзывов о каком-либо месте, а также возможность оставить свой отзыв.
5. Внесение расходов для дальнейшего анализа.

*Активность[2] – в данном контексте это пункт плана, куда пользователь хочет отправиться, т.е. какое-либо место.*

### Бизнес-логика

Бизнес логика – это определенный набор правил и ограничений для пользователя в автоматизированных операциях. В случае моего проекта – это ограничения, связанные с количеством поездок. Чтобы не засорять базу данных огромным количеством информации, пользователь может создать не более 5 поездок.

### UI/UX дизайн

UX дизайн – это то, каким образом пользователь взаимодействует с интерфейсом и насколько сайт или приложение для него удобны. UI дизайн – это оформление приложения.

Для того, чтобы приложение было не только функциональным, но и красивым, был разработан макет дизайна приложения в программе Figma. Весь макет составляют несколько разделов:

1. Страницы авторизации;
2. Главная страница;
3. Страницы создания поездки;
4. Страницы редактирования и просмотра поездки.

## Практическая часть

### Подготовительный этап

Во время работы я изучил следующие материалы:

1. Множество информации о том, как пишется код для Flutter;
2. Необходимую информацию для создания сервера;
3. Информацию о бизнес-логике;
4. Информацию о финансах и расчете стартовых инвестиций.

### Мобильное приложение

Я узнал о структуре приложений, написанных на Flutter. Каждый элемент – это объект класса *Widget* (то есть виджет[3]), располагающийся внутри другого виджета. Основными дочерними[4] от Widget классами являются *StatefulWidget* и *StatelessWidget*, то есть виджет с состоянием и без. Их различие в динамичности: без состояния объект не будет меняться. При его наличии объект может менять цвет, внутреннее содержимое, размеры и другие его свойства.

*Виджет*[3] *– маленькая программа или ее часть, расположенная на определенной части экрана и занимающая эту часть.*

*Дочерний класс*[4] *– класс объекта, который наследует все свойства от своего родителя (родительского класса) и добавляет к ним новые.*

Многим объектам, таким как кнопки, тексты, вводы, для красоты даются стили. В них задается цвет объекта, текста, параметры теней, обводки и многое другое, что дает возможность полностью повторить макет приложения.

Для того, чтобы код каждой страницы можно было быстро найти, я расположил файлы по соответствующим папкам – это основные страницы (главная и настройки), авторизация, страницы создания и редактирования поездки. Также, чтобы не засорять файлы со страницами, я создал отдельный – *"uikit.dart"*, где расположены основные виджеты приложения, а также вспомогательные классы. Структура файлов показана на *рис. 1.*

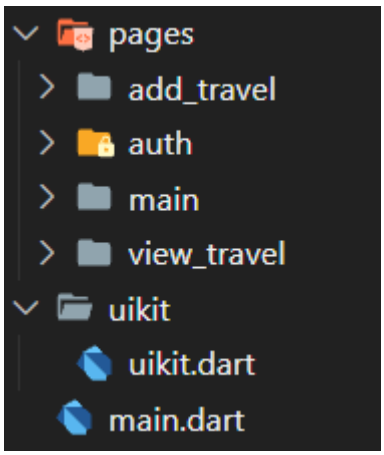



*Рисунок 1. Папки в проекте Flutter*

**Серверная часть**

Теперь поговорим о серверной части. Сервер был написан на языке Go. В нем имеется два файла: *"main.go"* – основной файл, и *"db.go"* – файл с функциями для управления базой данных MySQL (далее БД), в которой хранится необходимые данные:

1. Информация о пользователях (имя, почта, пароль от аккаунта, срок подписки);
2. Поездки пользователей (название, активности, город и т.д.);
3. Активности;
4. Отзывы об активностях.

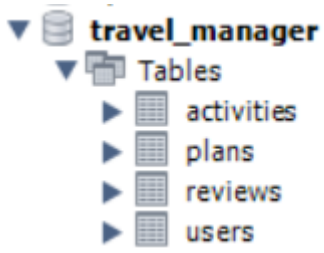



*Рисунок 2. Таблицы в базе данных*

Подключение к серверу и дальнейшее взаимодействие с ним происходит по протоколу HTTP[5]. У сервера также есть панель управление в браузере, через которую администраторы могут настраивать сервер, смотреть логи[6].

*Протокол HTTP[5] (hypertext transfer protocol) – веб-протокол, набор правил и методов, позволяющий передавать данные по сети интернет в виде текста.*

*Логи[6] – это журнал вывода сервера, который используется для его отладки при возникновении ошибок. В логи записываются ошибки и предупреждения при исполнении команд, время запуска и другая информация.*

Для безопасности пользователей их пароли хорошо защищены двойным шифрованием с помощью алгоритмов AES (шифрование с ключом) и SHA-256 (хеширование, т.е. шифрование в одну сторону без возможности расшифровки). В будущем для ещё большего улучшения безопасности планируется добавить “соль” – случайно сгенерированный набор символов, добавляющийся к паролю при шифровании. Он позволяет ещё больше запутать мошенников при попытке расшифровки пароля.

В каждом запросе на сервер нужно указывать имя пользователя и пароль для того, чтобы защитить данные от стороннего воздействия. То есть, человек, у которого нет аккаунта, не сможет получить никакого доступа к серверу, а если аккаунт и есть, то он может взять с сервера только информацию которая принадлежит ему.

Доступ к методам API сервера осуществляется по пути */api/v1*. Сервер может не только выполнять команды приложения, но и отдавать какие-либо файлы браузеру. Если вписать в адресную строку браузера адрес сервера, то откроется обычная страница, где рассказывается об этом проекте.

Весь исходный код находится на платформе GitHub. Ссылка на исходный код приложения и сервера.

## Практический этап

На этом этапе нужно было создать дизайн приложения, само приложение и сервер, которые будут работать вместе. Был создан проект в Figma, а в нем спроектирован дизайн с учетом бизнес-логики и нужного уровня удобства.

Далее был создан новый проект Flutter и Go. Для мобильного приложения был написан дополнительный код тестов, который позволяет довольно быстро проверить функциональность приложения, обнаружить ошибки и быстро исправить их. Тесты выполняют определенную последовательность действий (заданную в коде) и проверяют их результат. Если он совпадает с нужным, тест пройден, иначе выходит ошибка.

Чтобы создать поездку, пользователю нужно пройти следующие этапы:

1. Войти/зарегистрироваться (если до этого пользователь не входил в аккаунт, иначе вход произойдет автоматически);
2. Ввести в поле на главной странице город, в который человек хочет поехать;
3. Выбрать город из списка предложенных (названия городов берутся из открытого API базы данных ВКонтакте);
4. Ввести основную информацию о поездке (название, дата начала/конца, количество детей и взрослых);
5. Выбрать активности (их поиск расположен на сервере);
6. Завершить создание.

После этого, у пользователя на главном экране появится созданная поездка, которую можно отредактировать, нажав на неё.

## Детальное описание работы (мобильное приложение)

Теперь более детально разберем проделанную над приложением работу. Я начал с создания основных виджетов – *UI-кита*[7]. Это были кнопки двух видов – прозрачные и заполненные, а также ввод данных. На экране входа есть два поля для ввода – логин (почта) и пароль, кнопка “Войти”, которая отвечает за авторизацию и переход на следующую страницу, и кнопка “Регистрация”, переносящая пользователя на экран регистрации. В регистрации всё практически так-же, как и на экране входа, только добавляется поле “Имя”.

*UI-кит*[7] *– набор виджетов, используемый для создания интерфейса.*

Перейдем к главной странице. На ней я расположил несколько блоков – верхняя часть с приветствием и поиском города и основная часть, где располагаются планы пользователя. Я создал класс для блоков информации (подсказок) и класс для содержания поездок. Чтобы

пользователю было удобнее просматривать свои планы, я написал код для перелистывания – когда пользователь проводит пальцем по виджету с поездками влево или вправо, поездки плавно перемещаются и останавливаются ровно посередине экрана, чтобы их можно было легко рассмотреть и не мучаться с расположением.

При создании новой поездки пользователь нажимает на поле ввода города, и нажимает на кнопку далее *(см. рис. 3)*.

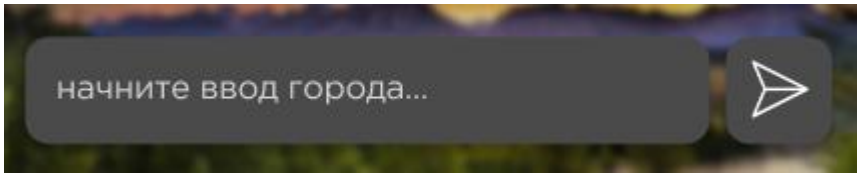



*Рисунок 3. Ввод города и кнопка "Далее"*

После нажатия приложение открывает новую вкладку – выбор города. Чтобы сохранить читаемость кода для этой вкладки я написал отдельный класс. На вкладке есть ввод, при изменении текста в котором обновляется список городов. Города берутся из открытой базы данных VK с помощью HTTP запроса, далее ответ *парсится*[8] и превращается в кнопки, при нажатии на одну из которых выбирается город. Чтобы перейти на следующую вкладку, нужно выбрать город из списка и нажать кнопку далее.

*Парсить[8] – автоматически анализировать текст, написанный в определенном формате, и превращать его в определенные объекты.*

На следующей странице есть 1 ввод – название поездки, также поля выбора даты начала и конца поездки (эти виджеты я написал также в UI-ките), а также ввод количества человек (детей и взрослых). Переход на следующую страницу осуществляется, если все поля заполнены и количество взрослых больше ноля.

Следующая страница – главная функция приложения. Это выбор активностей. На момент написания этой страницы уже был практически полностью готов сервер. Поиск активностей тут работает примерно так-же, как и поиск города, только формат запроса на сервер и ответа с него другой. Запрос также оправляется на сервер каждый раз, когда к тексту добавляется символ (а не просто при изменении текста, это сделано для экономии ресурсов сервера). Здесь также есть несколько кнопок, которые автоматически найдут нужное – например, кнопка с иконкой еды найдет вам рестораны и кафе. Можно добавить несколько активностей, а можно и не одной – продолжить можно в любом случае.

После этого пользователь увидит сообщение о том, что его поездка будет сохранена при нажатии на кнопку “Завершить” и что поездку можно будет отредактировать, нажав на неё. После нажатия кнопки, на сервер отправится запрос со всеми данными, которые были введены. Если приложение получит ответ о том, что план успешно сохранен, откроется главная страница. Иначе, если ответа не будет либо на сервере произойдет какая-либо ошибка, приложение сообщит об этом и попросит повторить попытку.

Теперь перейдем к редактированию поездок. Эти вкладки доступны, когда человек нажимает на кнопку “Просмотр и редактирование” в главном меню. На первой вкладке, которую видит пользователь написана вся информация о поездке, также внизу есть дополнительная функция – счетчик расходов. При нажатии на него откроется новая вкладка – расходы. Сюда можно внести все, что человек потратил в поездке.

Над блоком расходов на странице просмотра есть список активностей. При нажатии на одну из активностей откроется окно просмотра, где будет указан адрес места, название, время работы, какая-либо дополнительная информация (телефоны для связи).

**Детальное описание работы (сервер)**

При написании серверной части я начал с простого HTTP-сервера, который просто возвращал “Hello, world!” при попытке запроса на него. Далее я написал методы для обращения к базе данных – различные функции с SQL запросами[9]. Для каждого метода я написал отдельную функцию, чтобы код хорошо читался. Главной функцией является “guess_method”, что означает “угадать метод”. Она нужна для того, чтобы определить, какой тип запроса поступает на сервер, и вызвать соответствующую функцию. В каждом методе происходит авторизация пользователя – чтобы никто другой не получил доступ к информации.

*SQL запрос[9] – набор команд и инструкций для базы данных. Используются, если нужно записать или получить какие-либо данные.*

Сервер может ответить несколькими способами:

1. Если запрос успешный, он возвращает запрошенные данные в формате *JSON[10]*.
2. Если запрос не успешный, возвращается текстовый ответ в формате “код ошибки сообщение ошибки”, например “404 not found” или “500 server error”.

*JSON[10] (javascript object notation) – текстовый формат, предназначенный для передачи нетекстовых данных через Интернет.*

## Практическая и социальная значимость

Я думаю, что моё приложение может помочь многим людям не столкнуться с вопросами “Куда пойдем дальше? Чем займемся? Как успеть всё, что хотелось?” на отдыхе. По моей статистике, собранной среди друзей и одноклассников, более 70% людей планируют заранее только 1-2 пункта плана (например, аквапарк и музей), а около 10% людей вообще не думают, чем хотят заняться.

Преимущества моего приложения перед существующими:

1. Удобство и красота.
2. Многофункциональность.
3. Простота интерфейса.

Мой проект – это эффективное средство для быстрого создания плана отдыха, который предоставляет кроме основной функции ещё несколько дополнительных, но очень нужных.

## Заключение

Подводя итоги работы, я могу утверждать, что поставленная цель по созданию мобильного приложения для планирования поездок достигнута. Созданное программное обеспечение выполняет все поставленные задачи. По моему мнению, оно будет очень полезно многим людям.

В дальнейшем, для развития проекта, я планирую:

1. Расширить команду.
2. Сделать доступ по подписке для получения выгоды.
3. Запустить несколько соцсетей для того, чтобы люди узнали о проекте.

## Список используемых ресурсов

Visual Studio Code – среда написания кода.

Golang – средство выполнения серверного кода.

Flutter – фреймворк для создания мобильных приложений.

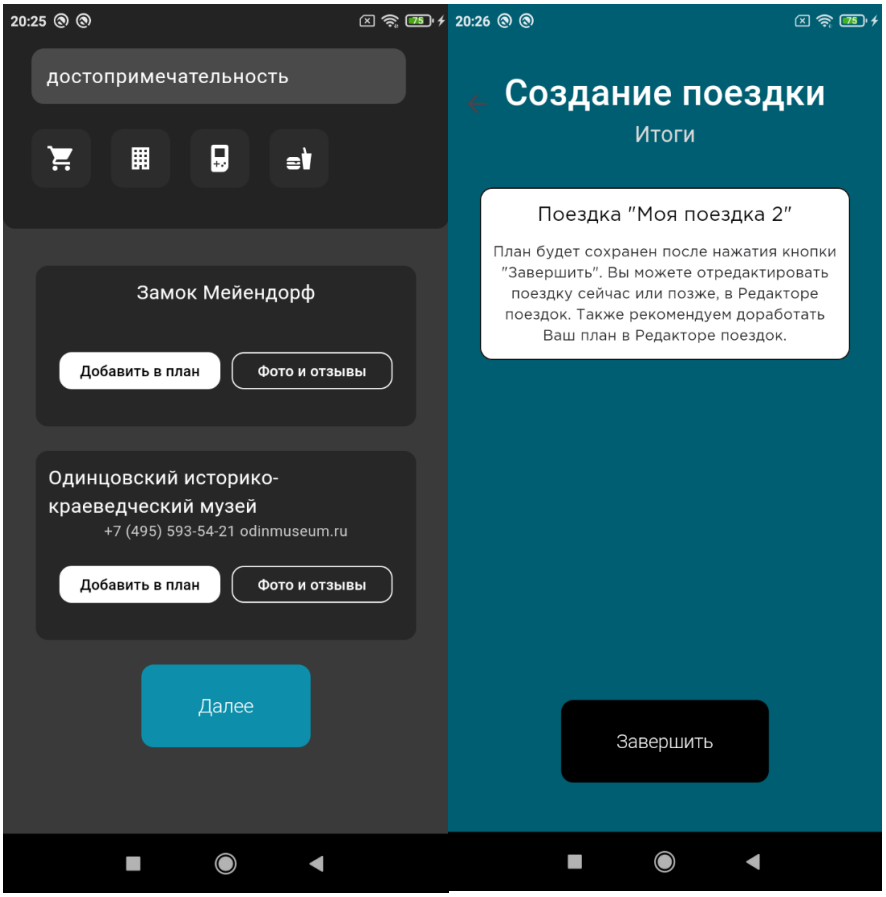
vk.com – социальная сеть, из базы данных которой берутся названия городов.

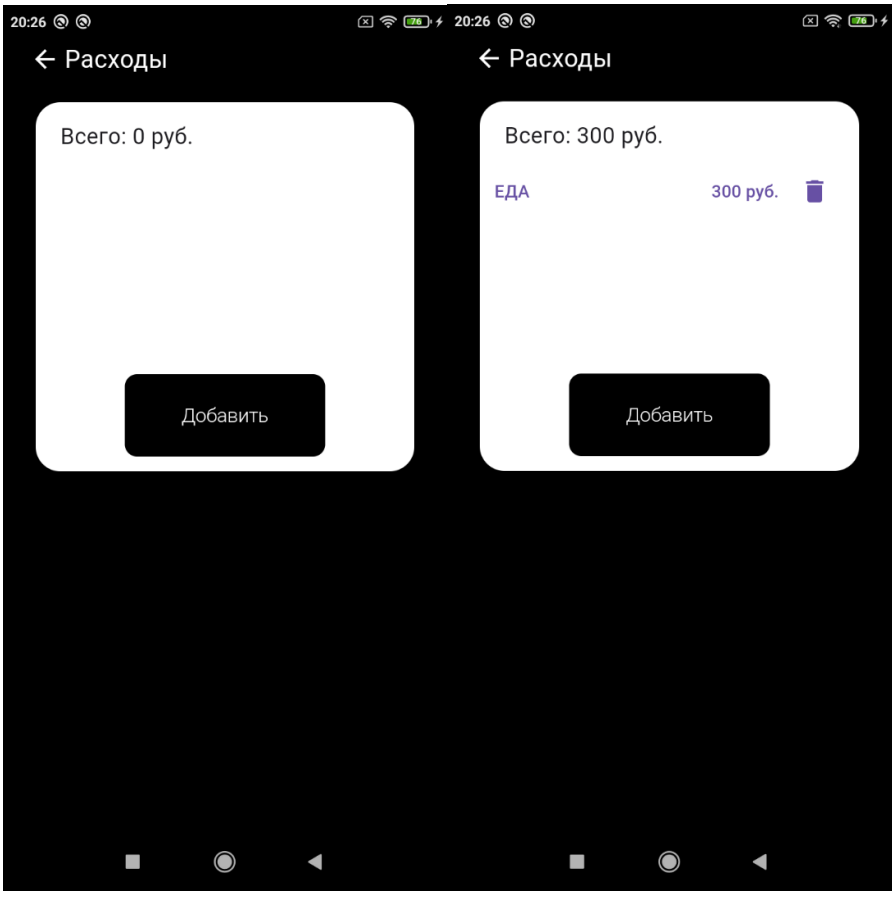
## Список используемой литературы

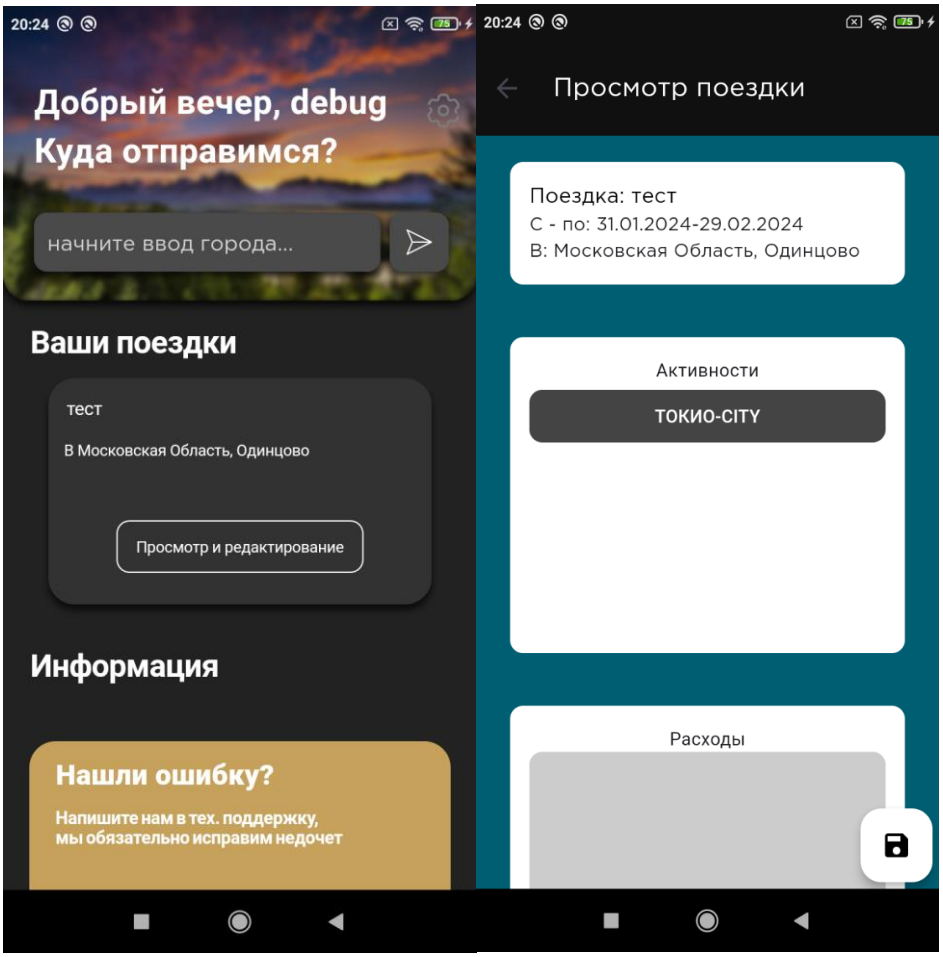
Wikipedia.org – энциклопедия, откуда брались определения в тексте.

## Приложения

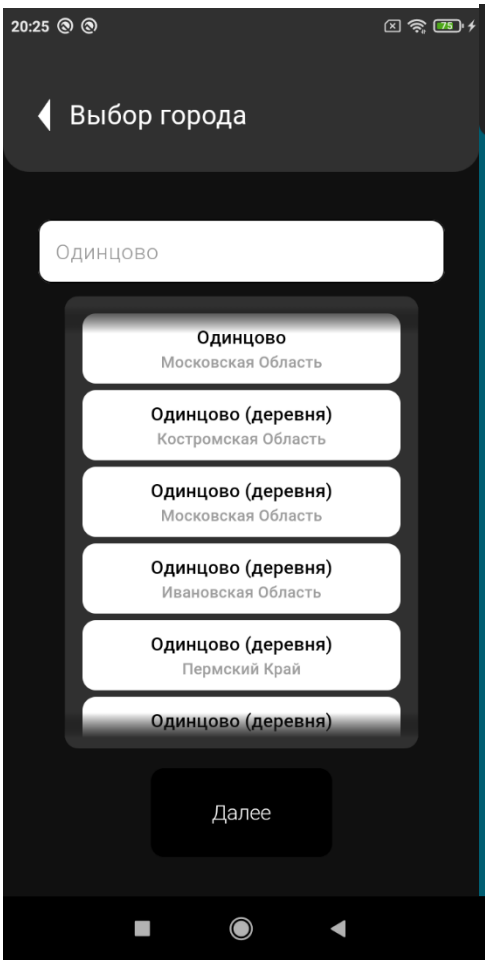
Приложение 1. Мобильное приложение

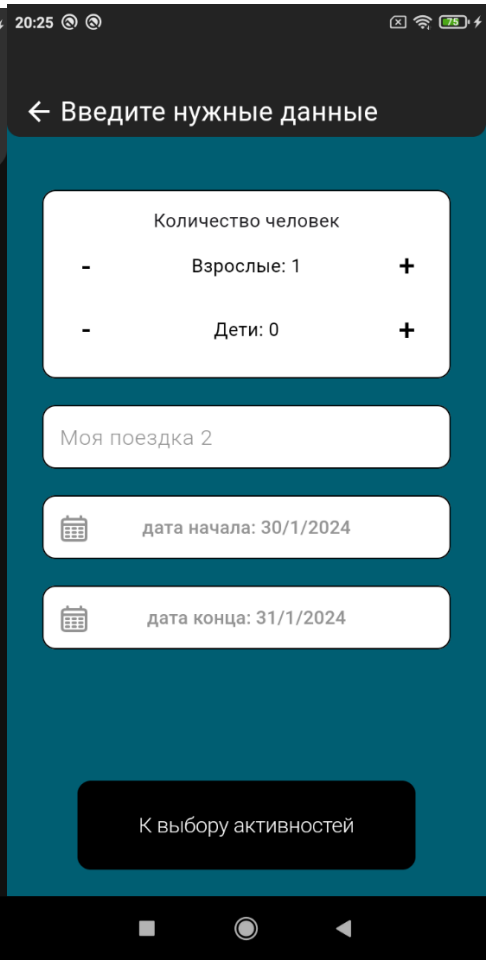



20:24
Добрый вечер, debug
Куда отправимся?
начните ввод города...
Ваши поездки
тест
В Московская Область, Одинцово
Просмотр и редактирование
Информация
Нашли ошибку?
Напишите нам в тех. поддержку,
мы обязательно исправим недочет


20:24
Просмотр поездки
Поездка: тест
С - по: 31.01.2024-29.02.2024
В: Московская Область, Одинцово
Активности
ТОКИО-CITY
Расходы


20:25
Выбор города
Одинцово
Одинцово
Московская Область
Одинцово (деревня)
Костромская Область
Одинцово (деревня)
Московская Область
Одинцово (деревня)
Ивановская Область
Одинцово (деревня)
Пермский Край
Одинцово (деревня)
Далее


20:25
Введите нужные данные
Количество человек
- Взрослые: 1 +
- Дети: 0 +
Моя поездка 2
дата начала: 30/1/2024
дата конца: 31/1/2024
К выбору активностей