# Задание № 1. Создание проекта для управления разработкой

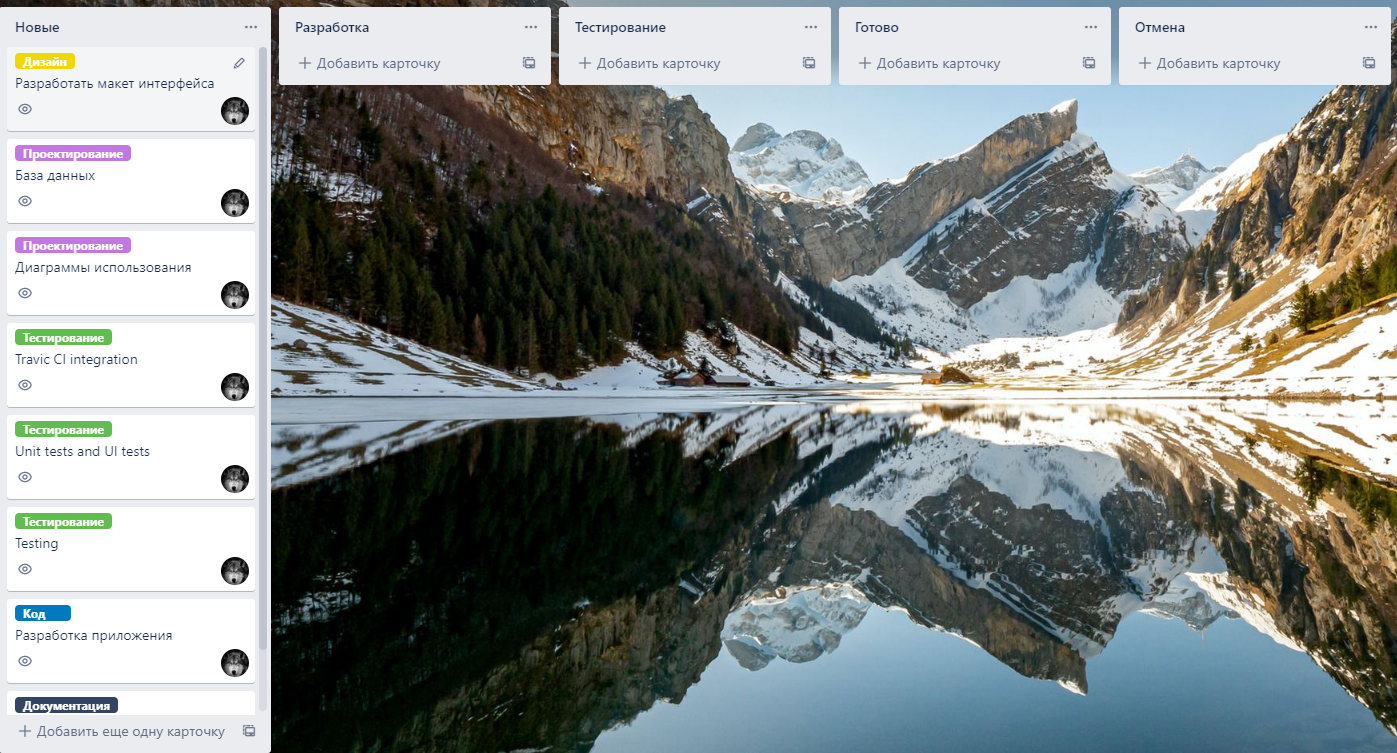
Данный проект выполняется в группе из 2 участников (максимально 3 участников). Создать новый проект в стиле Kanban, используя github projects, [https://trello.com](https://trello.com/) или <http://www.targetprocess.com/>. Добавить в него членов команды, распределив роли следующим образом: *Менеджер проекта, Разработчик, Проектировщик интерфейсов, Тестировщик.*

*Роли в команде были распределены следующим образом:*

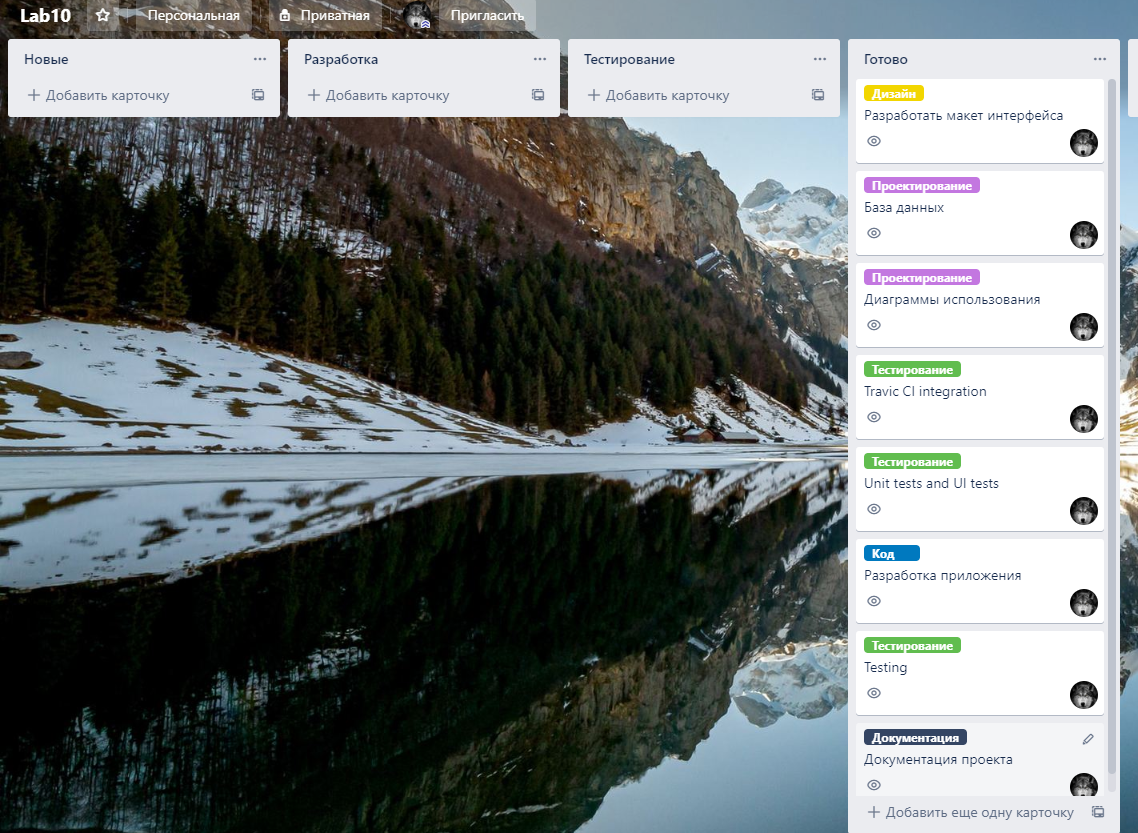
* Менеджер проекта – Шишлянников Иван
* Разработчик – Шишлянников Иван
* Проектировщик интерфейсов – Шишлянников Иван
* Тестировщик – Шишлянников Иван

Для создания проекта в стиле Kanban использовал Trello.

Состояние проекта в начале разработки:



Состояние проекта к концу разработки:



# Задание № 2. Проектирование интерфейса приложения, разработка требований и документирование проекта

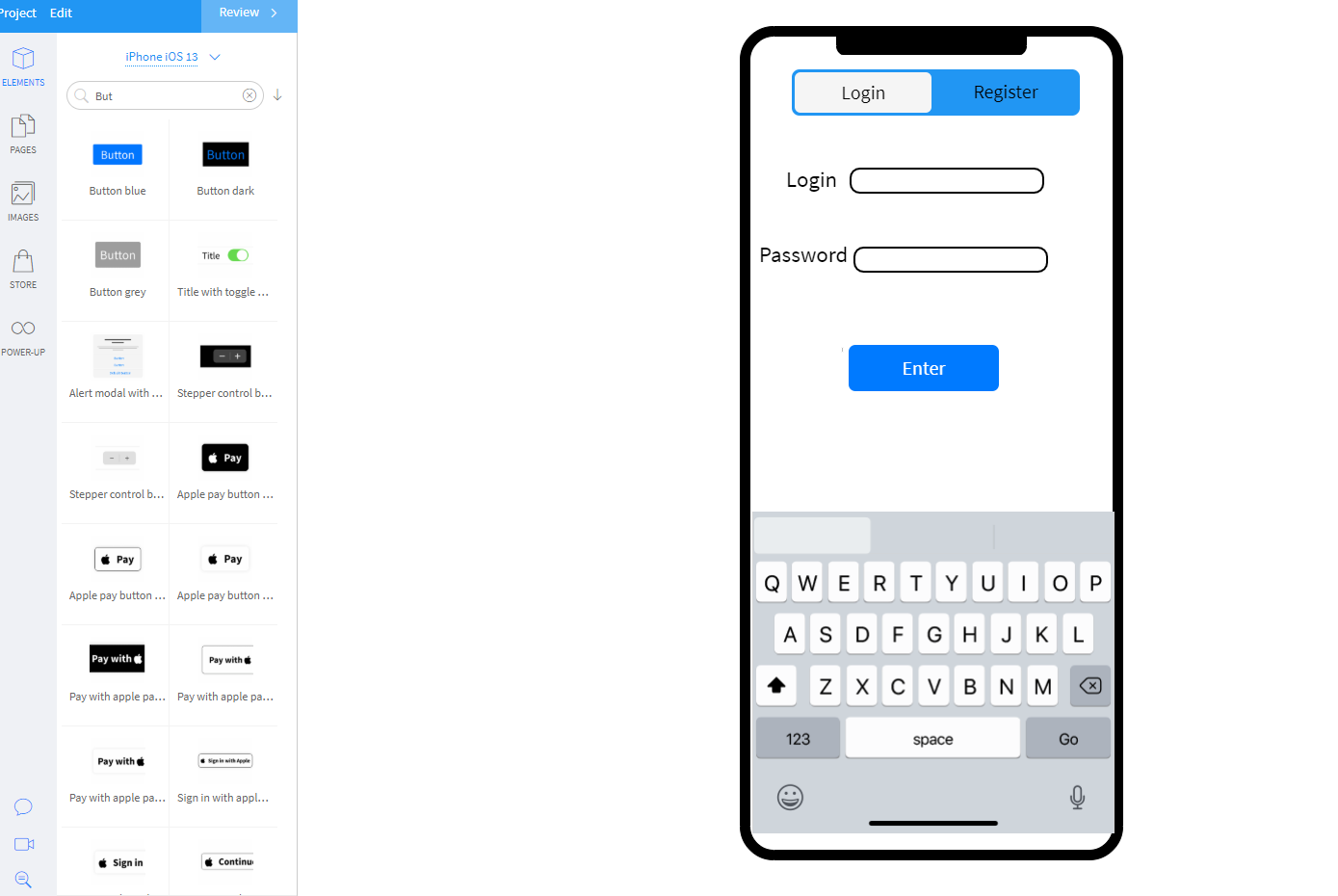
## Упражнение 2.1. Проектирование интерфейса приложения

Разработать mockup-ы (дизайн-макеты) приложения и добавить в проект высокоуровневые требования (features) и user stories, распределив работы между участниками команды. Дизайн-макет приложения может быть разработан с помощью приложения Sketch, Pencil, онлайн-сервиса Figma или других ([Uxpin](http://www.uxpin.com/), [Invision](https://projects.invisionapp.com/), [Creately](http://creately.com/), [Cacoo](http://www.cacoo.com/) и др.). Перед подготовкой макета познакомиться с руководством по проектированию интерфейсов для iOS — [iOS Human Interface Guidelines](https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/).

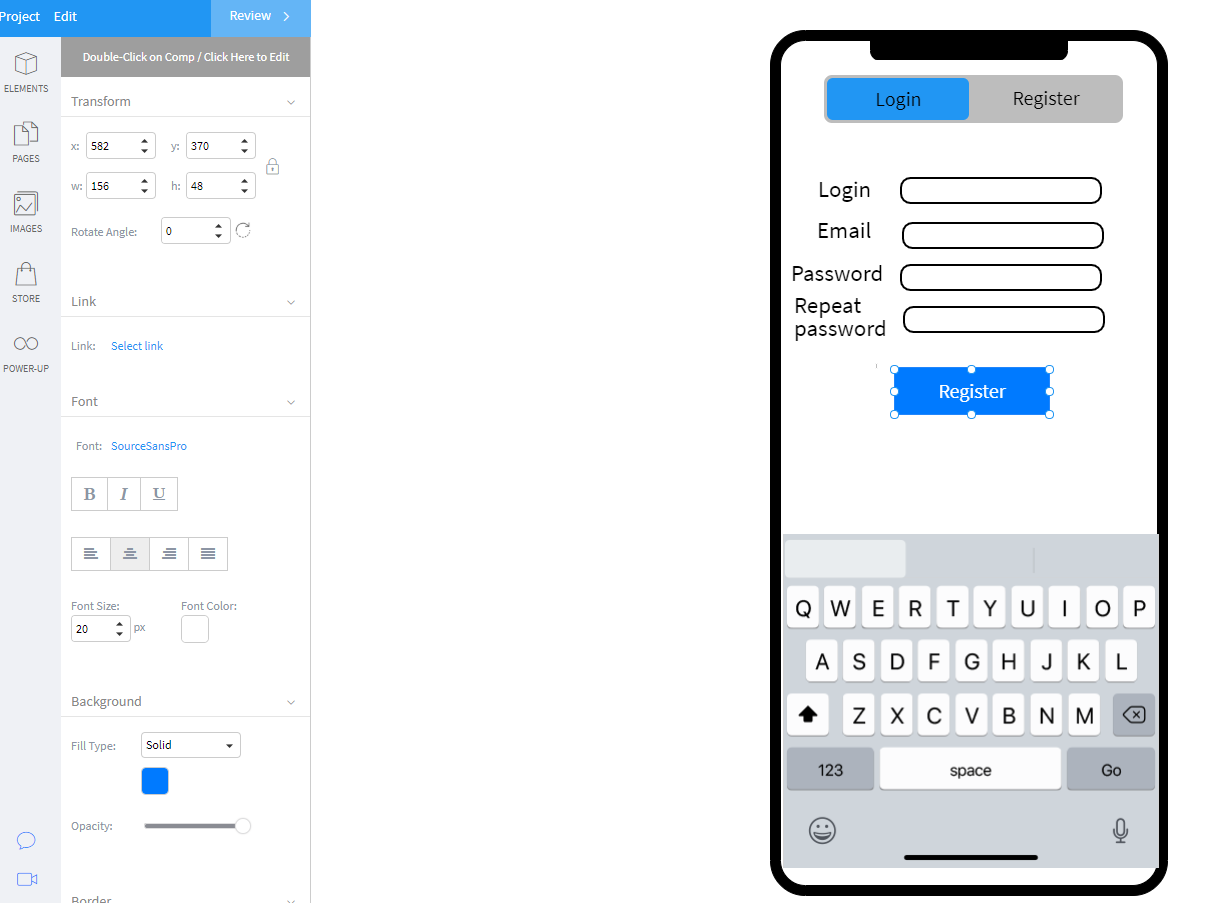
Работы распределить таким образом, что спецификацию готовят менеджер проекта и разработчик, а проектировщик интерфейса — разрабатывает дизайнстраницы (mock-up) приложения. Требуется разработать как минимум 3 вида экрана приложения. Например, если у нас банковское приложение, то разрабатываем экран авторизации, экран (dashboard) после авторизации и еще один экран, например, просмотра списка счетов.

Для разработки макетов использовал сервис wireframepro.

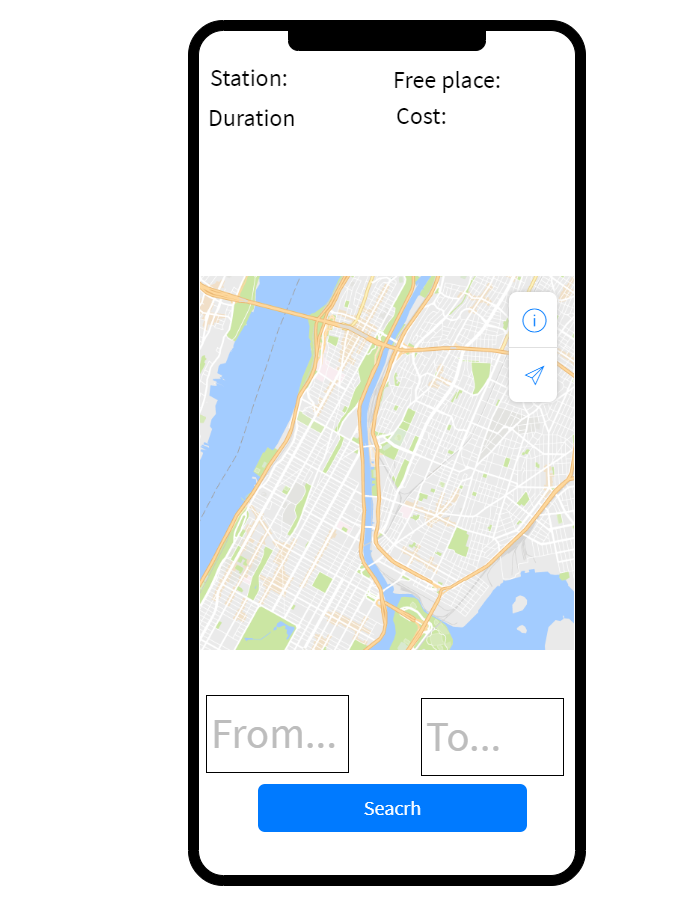
Разработка окна авторизации:



*Разработка окна регистрации:*



*Разработка главного окна:*



## Упражнение 2.2. Разработка требований

**Разработать спецификацию требований согласно структуре ниже:**

1. Назначение приложения/системы

Приложение создано с целью облегчить поиск необходимых маршрутов международных автобусных рейсов. Так же упрощает поиск за счет предоставления информации о погоде в выбранных городах.

1. Общее описание (пользователи, требования к программному и аппаратному обеспечению)

Пользователь после регистрации в системе имеет возможность выбрать пункты отправления и прибытия, после чего ему будет предоставлена подробная информация о цене, длительности, и наличии свободных мест, а также о всех рейсах доступных в данном городе отправки.

При желании пользователь может узнать информацию о погоде в указанном городе.

1. Спецификация требований
   1. Функциональные требования

3.1.1. Предоставление информации об выбранных рейсах.

3.1.2. Предоставление информации о погоде в указанном городе.

* 1. Удобство использования

3.2.1 Пользователь получает всю информацию используя карту.

3.2.2 Пользователь может получить информацию о погоде, нажав на карте на соответствующую иконку города

* 1. Требования надёжности

3.3.1 Карта должна отображать реальные гео и метео-данные в зависимости от выбора пользователя.

3.3.2 Пользователь не может войти в аккаунт другого пользователя, не зная необходимого логина и пароля.

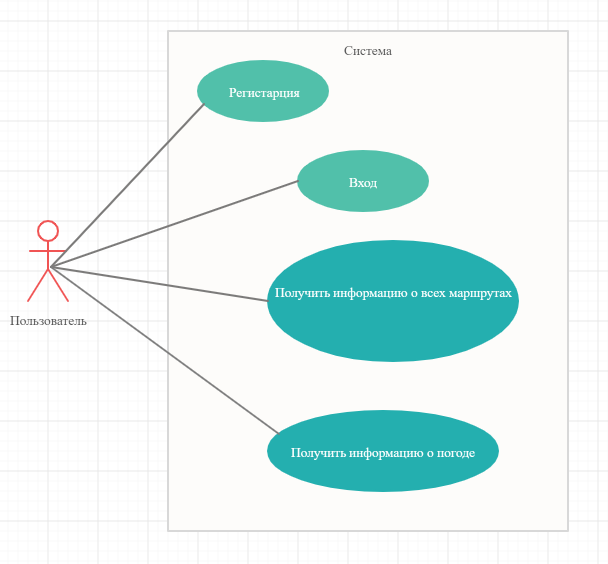
* 1. Требования производительности

3.4.1 Карта должна загружаться плавно и быстро.

3.4.2 Иконки на карте должны появляться в нужной локации с точностью 100 метров.

4 Диаграммы

* 1. Варианты использования со сценариями



4.1.1Сценарий “Регистрация”

* + - 1. Пользователь вводит необходимые поля
      2. Подтверждает свой ввод нажатием на кнопку “Зарегистрировать”
      3. В случае если ввод валидный, и пользователь с введенным логином еще не зарегистрирован, то пользователь переходит в главное меню приложения. Иначе пользователь получает уведомление об ошибке.

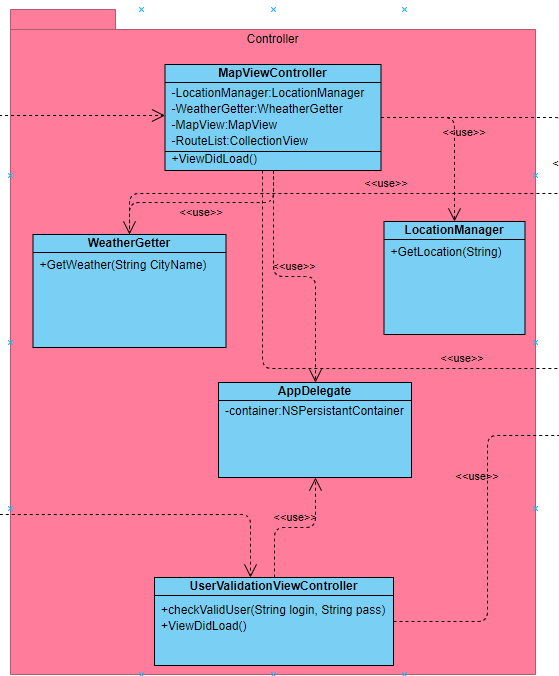
4.1.2 Сценарий “Вход”

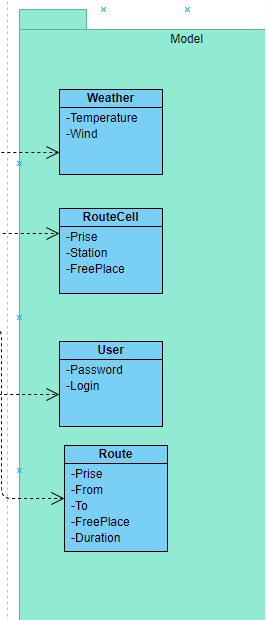
4.1.2.1 Пользователь вводит необходимые поля

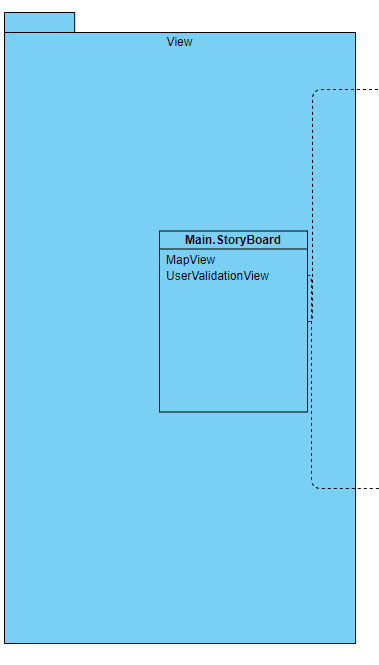
4.1.2.2 Подтверждает свой ввод нажатием на кнопку “Войти”

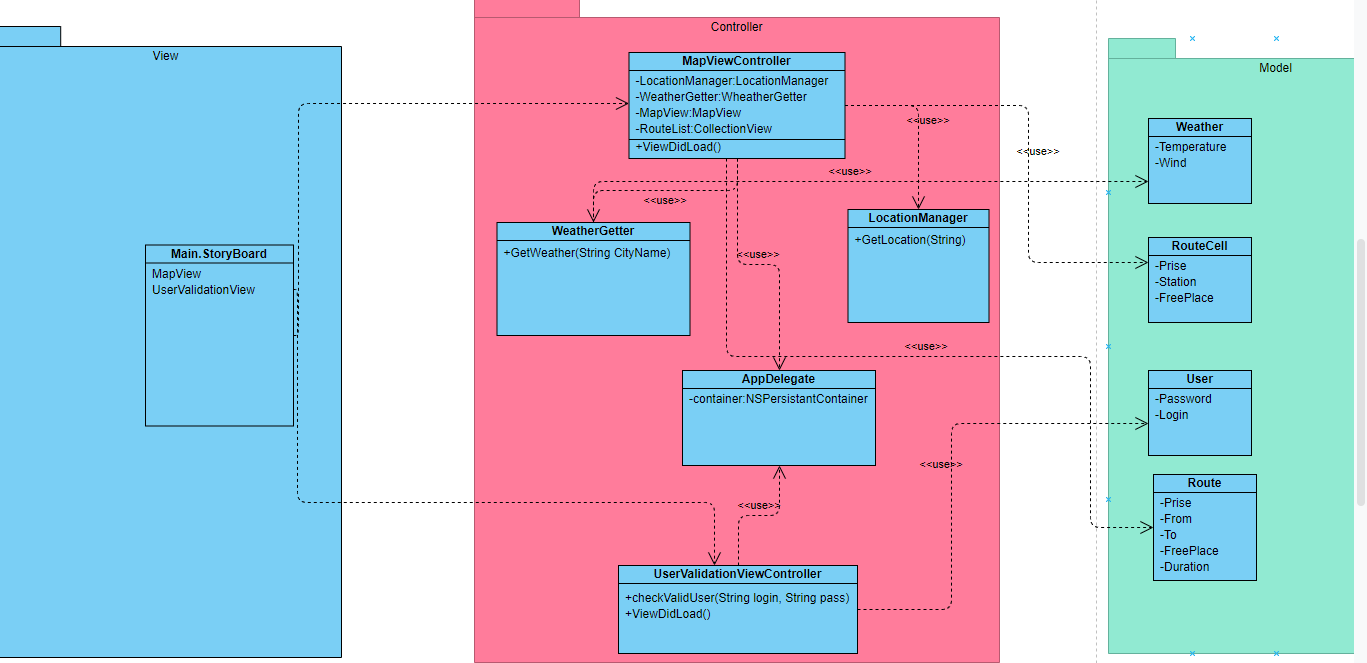
4.1.2.3 В случае если ввод валидный, и пользователь с введенным логином и паролем зарегистрирован, то пользователь переходит в главное меню приложения. Иначе пользователь получает уведомление об ошибке

* + 1. Сценарий “Получить маршруты”
       1. Пользователь города отправки и назначения.
       2. В случае валидного ввода пользователю на карте отображаются точки городов, а также информация о всех рейсах.
    2. Сценарий “Получит информацию о погоде”
       1. Пользователь нажимает на иконку на иконку города на карте.
       2. Во всплывающем окне появляется информация о погоде указанного города.
       3. В случае ошибки загрузки пользователю выведется соответствующее сообщение.
  1. Диаграммы классов

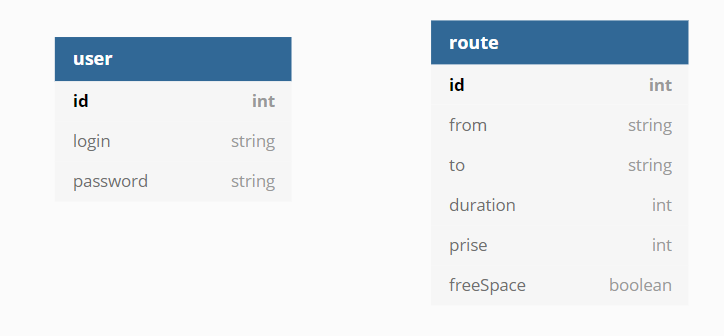






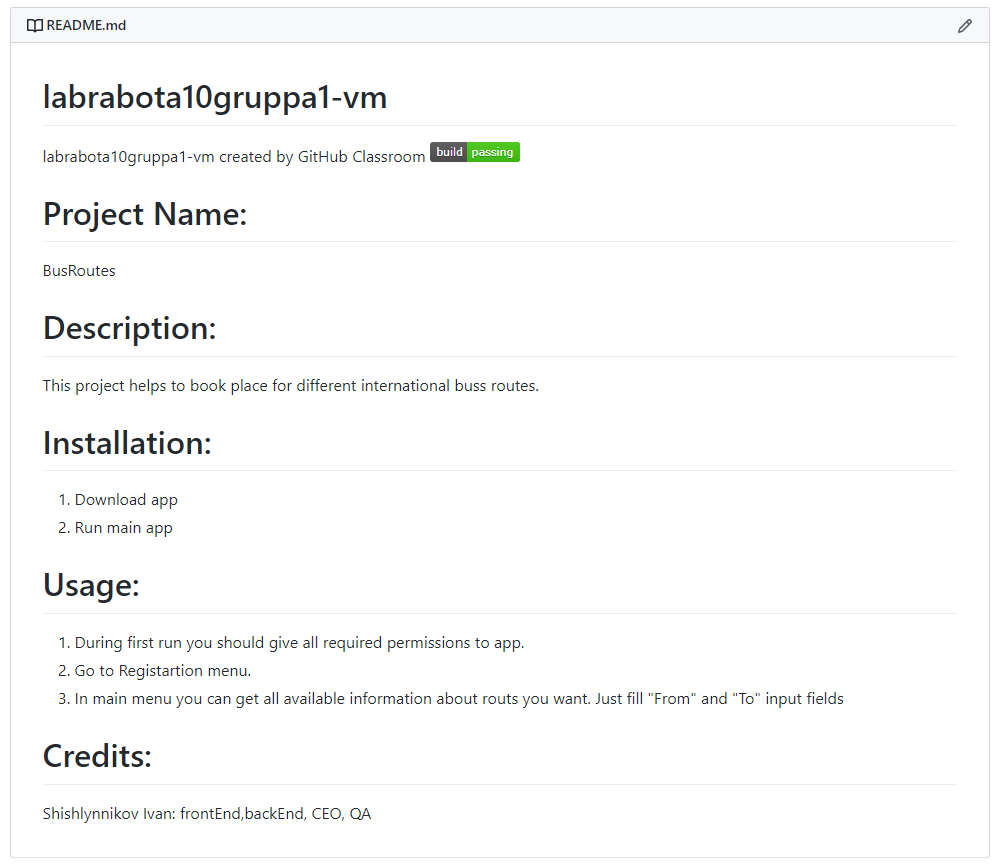


* 1. Физическая модель БД

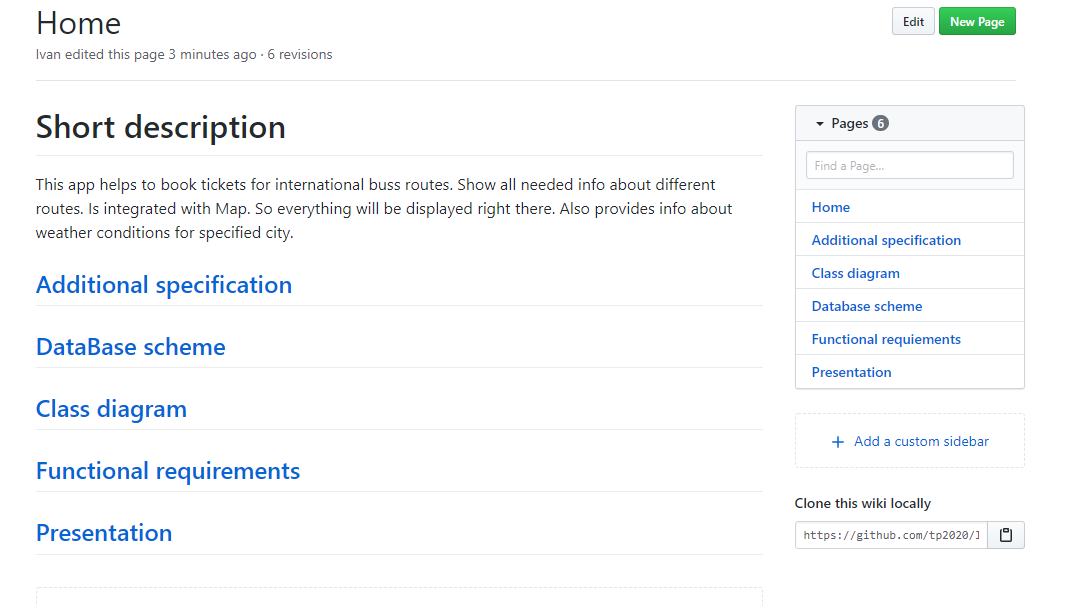


## Упражнение 2.3. Документирование проекта

README проекта :



Главная страница Wiki:

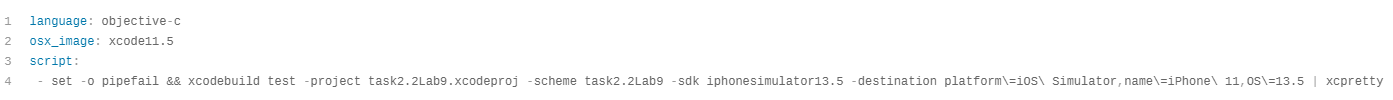


# Задание № 3. Настройка сервиса непрерывной интеграции и разработка Unit- и UI-тестов

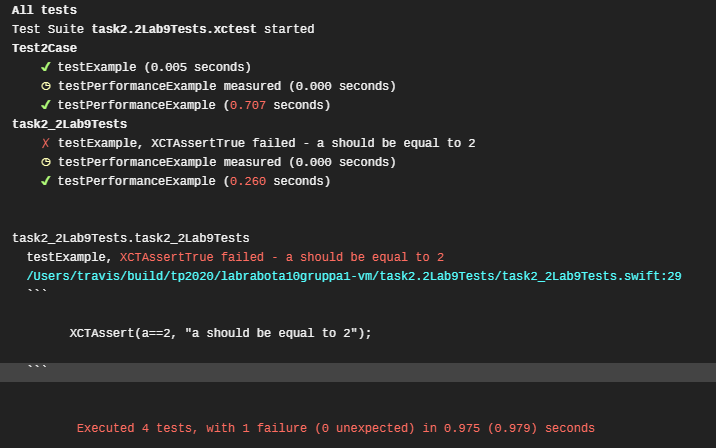
Интеграция с Travis прошла успешно. Большую часть времени занимал запуск симулятора на виртуальной машине Travis (3 минуты за запуск и 3 на выключение).

Так же при попытке запускать тесты ( test-without-building) в облаке, Travis вызывал различные ошибки, которые я не смог решить. Поэтому от данной возможности пришлось отказаться, и сборка проекта всегда происходит перед тестами.

Yml файл:



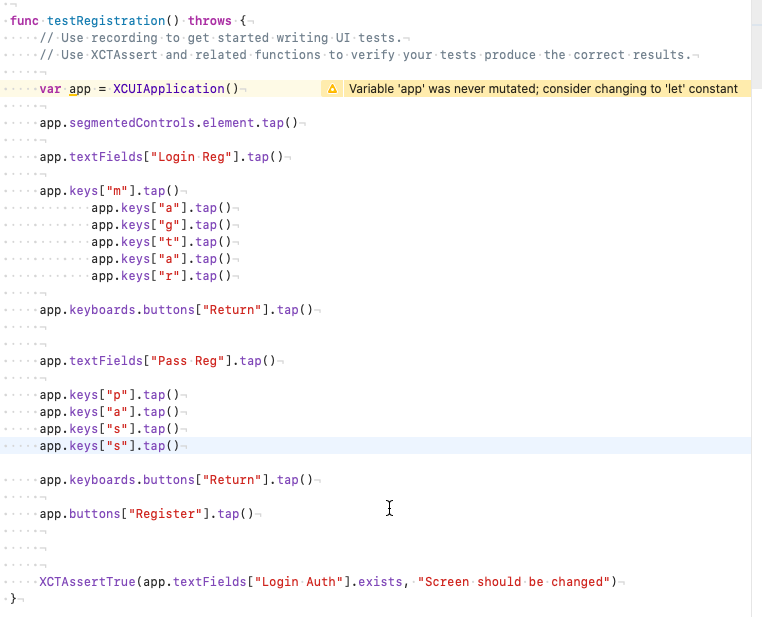
Примеры сборки и тестов:



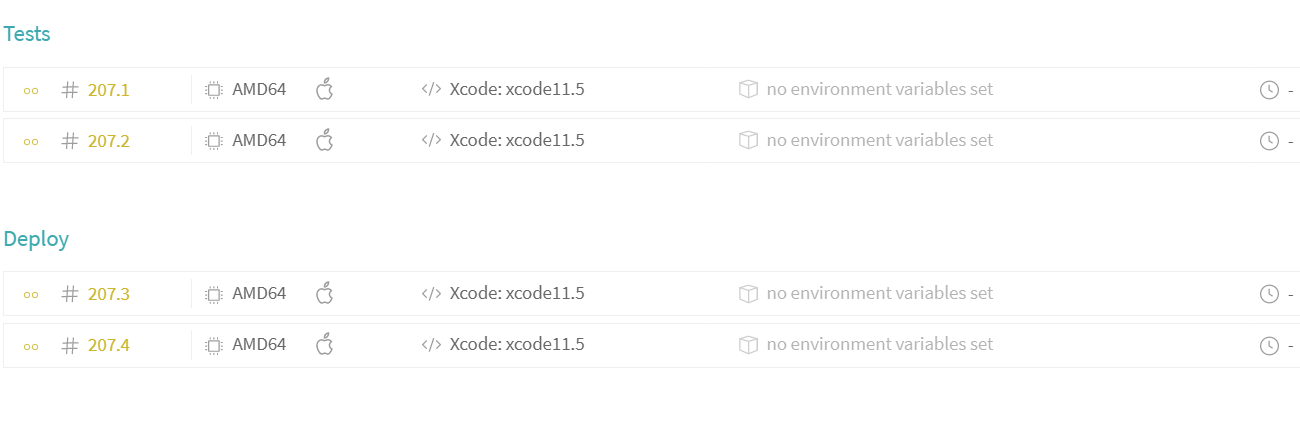
Тестирование работы с БД.



Тестирование UI



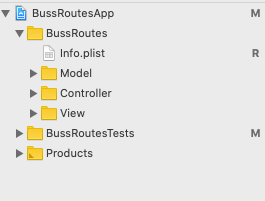
Процесс работы Travis ci:



# Задание № 4. Разработка приложения

Приложение разрабатывалось на основе паттерна MVC

Структура проекта:



В приложении присутствует локализация на русском, английском и японском языках.

Примеры работы:

