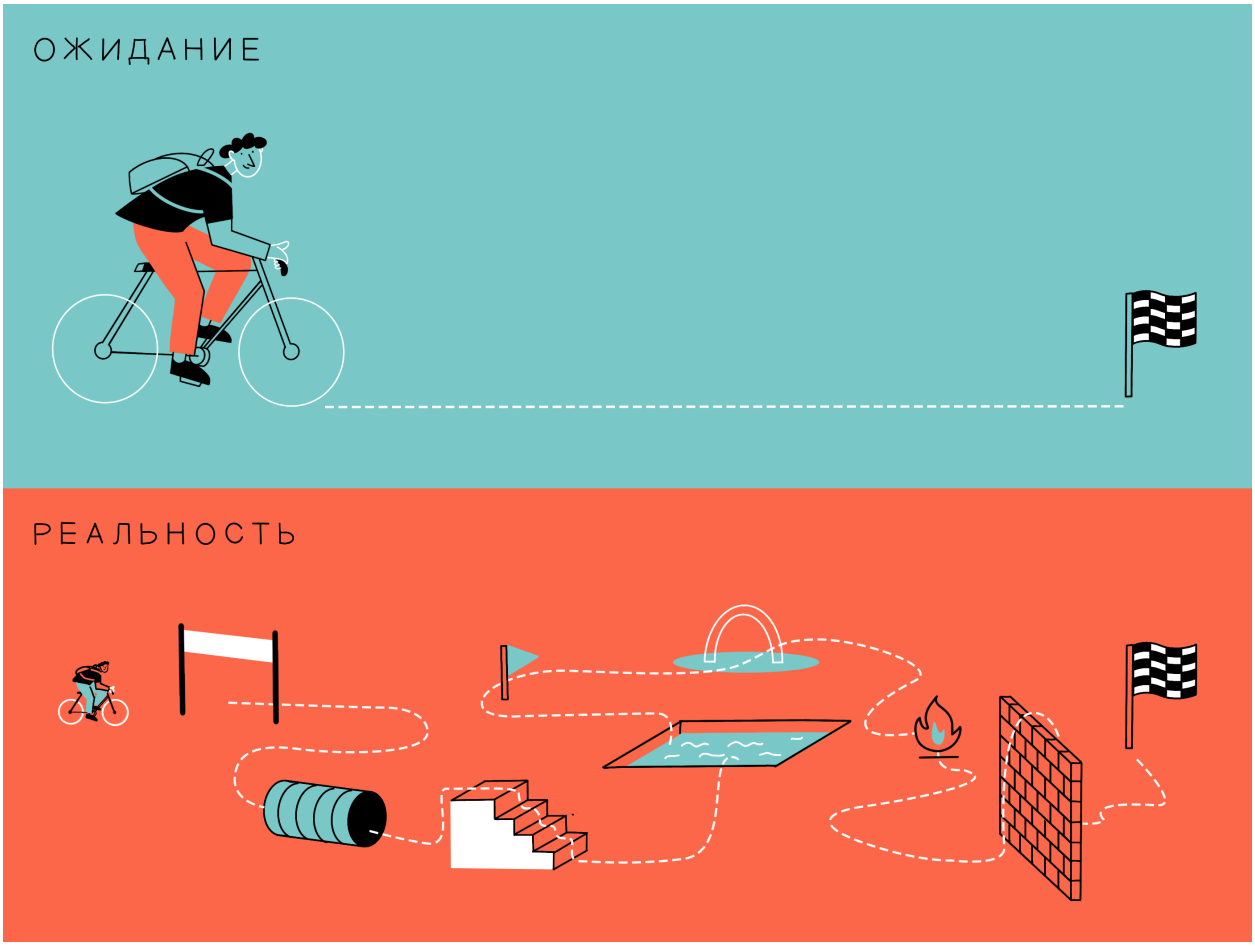
План работы

Код — это воплощение идеи. Спланируйте работу, и лишь после этого начинайте писать код. Голова, бумага и карандаш — вам в помощь; редактор кода понадобится гораздо позже.

Как бы ни был прекрасен план — реальность всё равно будет соответствовать иллюстрации. Но без планирования будет ещё хуже.



Проектирование моделей и связей между ними

Сколько моделей должно быть в проекте, какими полями они связаны, какие поля обязательны, где будут нужны множественные связи, где может потребоваться каскадное удаление?

Решите, где может потребоваться индексирование, например — для названий ингредиентов.

Набросайте схемы моделей и связей. Скомкайте, сожгите, набросайте снова.

Проектирование приложений в составе проекта

Прочитайте техническое задание, выделите сходные задачи, объедините их в приложения (пока что в голове или на листе бумаги).

Проведите мысленный эксперимент: а можно ли приложение скопировать в другой проект, заработает ли оно там в том виде, в котором вы его запланировали. Получилось? Значит, идея верна.

Спланируйте в общем виде функции: будет ли правильно хранить всю логику во view-функциях или есть смысл вынести какую-то часть в отдельные файлы, чтобы не загромождать *views.py*

Проектирование запросов (URL)

Придумайте структуру адресов: она должна быть удобной для пользователя и соответствовать структуре вашего проекта.

В адресах не транслитерируйте русские слова: *recipe* выглядит лучше, чем *retsept*. Избегайте большой вложенности: *recipe/<id>* лучше, чем *<username>/recipe/<id>/detail*

Создание проекта, виртуальное окружение, база данных, Git

Склонируйте репозиторий `foodgram-project` с исходными материалами проекта, разверните в нём виртуальное окружение, установите новый Django-проект.

Разверните и подключите базу данных PostgreSQL (если компьютер не очень быстрый — разрабатывайте проект на SQLite, а PostgreSQL подключите позже, при подготовке к деплою).

Создайте суперпользователя и несколько обычных пользователей, они пригодятся для проверки работы проекта.

Создайте файл зависимостей *requirements.txt*, актуализируйте его каждый раз, когда устанавливаете новые пакеты.

Приложения, модели

Опишите первое приложение (*app*), его модели и связи. После этого займитесь следующим приложением. Такая очерёдность позволит вам скорректировать план, если в нём что-то не так.

Логика, view-функции, path(), подключение шаблонов

Вот теперь начинается код.

Именно в этот момент всё пойдёт не по плану. Это нормально. Но если все предыдущие этапы выполнены — работа будет несколько легче, чем если бы этой подготовки не было.

Самое время запустить в консоли интерактивный режим Python, выполнить `import this` и немного помедитировать: круг замкнулся.

И после этого — писать код.

Первое ревью проекта

После того, как проект заработает и вы проверите его по чек-листу — наступит момент первого этапа проверки. Прочтите урок «Как сдавать проект», там всё описано.

Подготовка к продакшн, настройка статики, Docker

Проверьте актуальность файла *requirements.txt*, базу замените на PostgreSQL (если вы разрабатывали проект на SQLite).

Напишите докер-файл с Django-проектом и Gunicorn. В конце сборки контейнера не забудьте добавить команды для миграции и сборки статики.

Напишите docker-compose, в нём опишите запуск контейнеров из официальных образов PostgreSQL и nginx и своего контейнера с проектом. Опишите конфигурацию контейнеров.

Запушьте контейнер с проектом на Docker Hub.

Разверните контейнеры на своём сервере в Яндекс.Облаке

Проверка по чек-листу

Чтобы ничего не забыть — прочтите чек-лист и убедитесь, что всё работает, как задумано. И после этого отправляйте проект на второй этап проверки.

Критерии оценки работы

Никаких жёстких рамок по структуре и содержанию кода задано не было, есть несколько технических условий общего плана, которые должны быть соблюдены и которые мы будем проверять

И проект, разумеется, должен работать — весь, без исключений.

Функциональность проекта

Все сервисы и страницы доступны для пользователей в соответствии с их правами.

Рецепты на всех страницах сортируются по дате публикации (новые — выше).

На всех страницах с рецептами корректно работает фильтрация по тегам и пагинатор (в том числе пагинатор работает при фильтрации по тегам).

Обрабатывается ошибка 404.

Инфраструктура

Проект работает с СУБД PostgreSQL.

Проект запущен на сервере в Яндекс.Облаке в трёх контейнерах: nginx, PostgreSQL и Django+Gunicorn. Контейнер с проектом обновляется на Docker Hub.

В nginx настроена раздача статики, остальные запросы переадресуются в Gunicorn.

Данные сохраняются в volumes.

Оформление кода

Код соответствует PEP8