Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Автоматизированные системы обработки информации и управления»



**Отчет**

**Лабораторная работа № 7**

**По курсу «Разработка интернет приложений»**

**ИСПОЛНИТЕЛЬ:**

Группа ИУ5-55Б

Болдин Д.А.

"16" декабря 2020 г.

**ПРЕПОДАВАТЕЛЬ:**

Гапанюк Ю.Е.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2020 г.

Москва 2020

1. **Задание**

Необходимо разработать систему для проведения викторин по российским городам.

Работа была разделена на несколько частей:

* Проработка дизайна
* Разработка фронтенда
* Разработка бекенда
* Деплой и дальнейший мониторинг

1. **Разработка фронтенда**

На стороне фронтенда в проекте используется FLUX архитектура и PWA

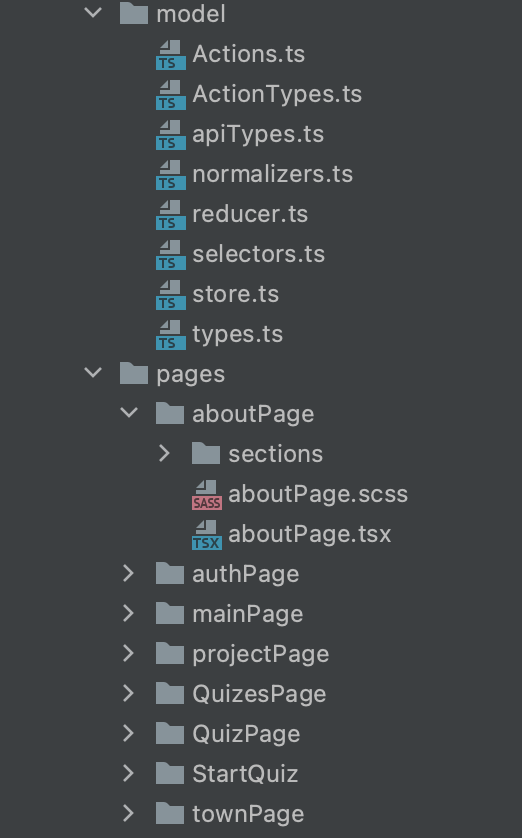
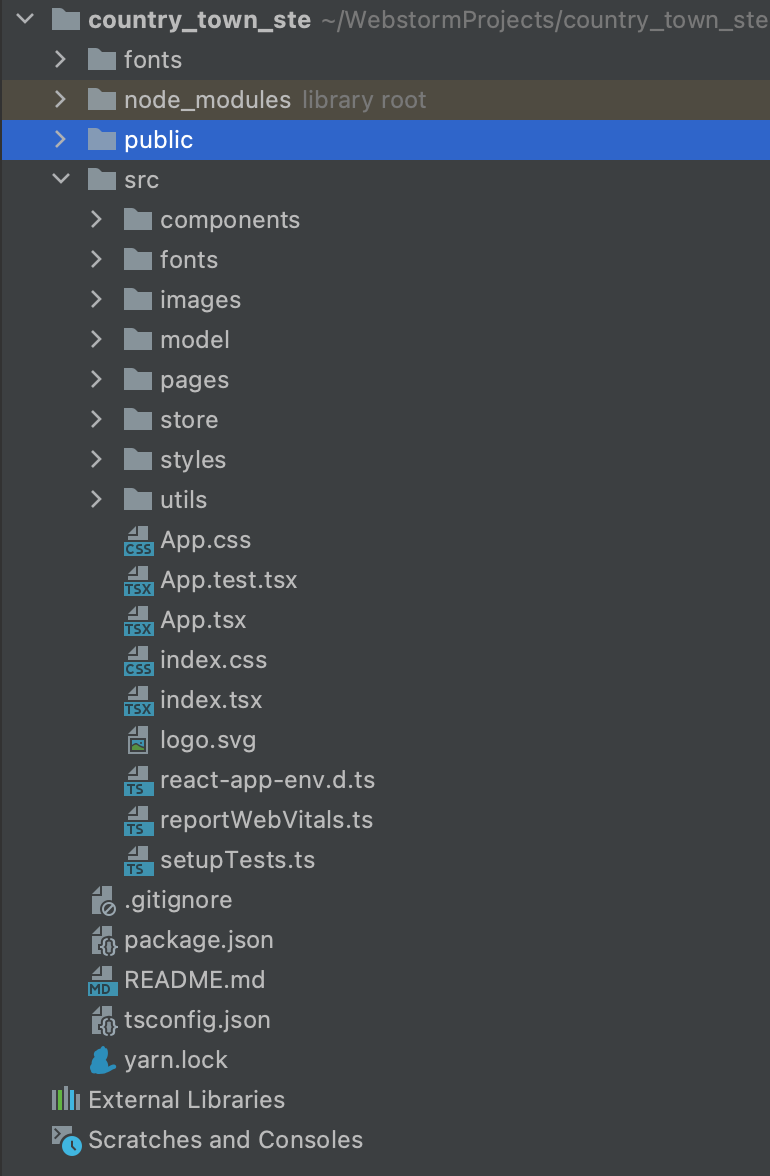
Этапы разработки фронтенда

* Сетап проекта.
* Проектирование макетов страниц в figma
* Проектирование базовых элементов интерфейса
* Проектирование хранилища веб-приложения.
* Проектирование роутинга в приложении.
* Верстка страниц и компонентов.
* Применение стилей для страниц.

Технологии, используемые в фронтенде:

* TypeScript — расширенная версия JavaScript со статической типизацией
* React - Создание пользовательских интерфейсов
* Redux - Хранилище
* HTML - Разметка
* SCSS - Стили

Структура проекта:



1. **Разработка бекенда**

На стороне бэкенда в проекте используется MVP архитектура.

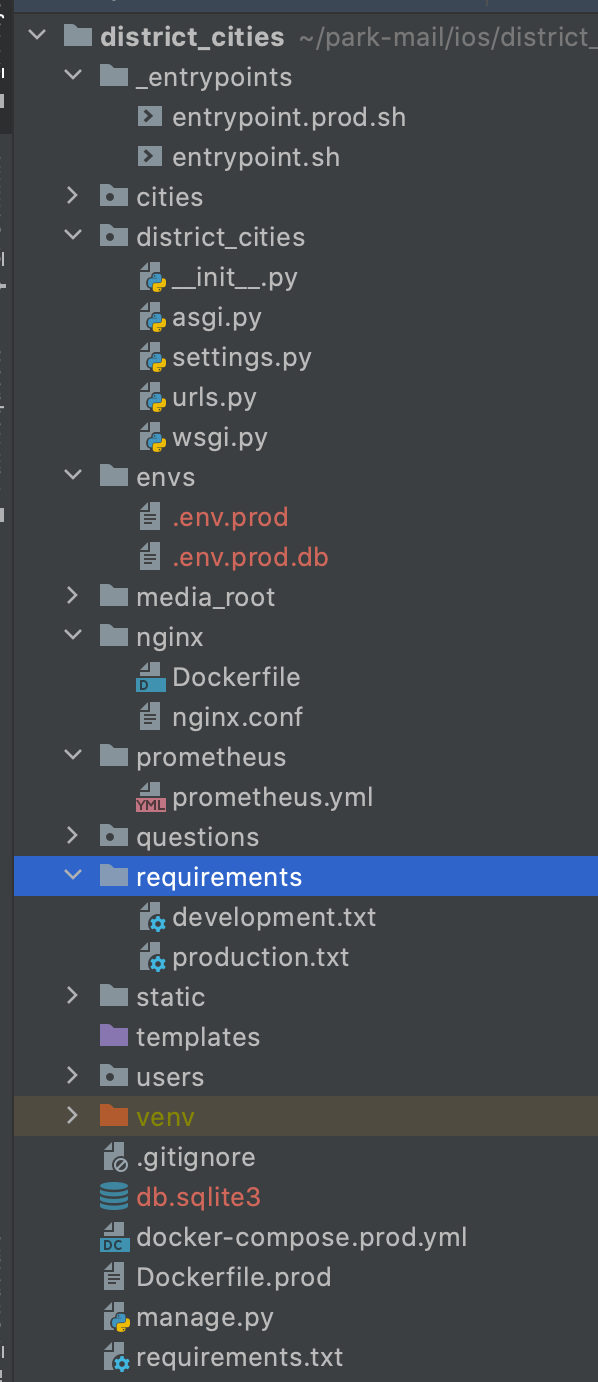
Этапы разработки бекенда:

* Проектирование базы данных
* Написание моделей в Django
* Написание сериалайзеров для моделей с помощью Django Rest Framework
* Создание портала для администратора
* Написание view, а также добавление необходимой бизнес-логики в модели

Технологии, используемые в бекенде:

* Django
* Django Rest Framework(DRF)
* PIL(для работы с картинками)
* Django-json-widget(виджет для редактирования json в admin панели)
* Django-prometheus(мониторинг)
* Postgresql(база данных)

Структура проекта:



1. **Деплой и мониторинг**

После написания бекенда и фронтенда необходимо произвести деплой и настроить удобный мониторинг.

Во время деплоя были использованы следующие технологии

* Nginx для отдачи фронтенда и проксирования запроса на бекенд
* Node\_exporter для отдачи информации о состоянии сервера
* UWSGI Gunicorn позволяет запустить несколько процессов с Django для ускорения работы сайта
* Prometheus собирает и хранит все данные о мониторинге из Django и Node\_exporter
* Grafana используется для составления дашбордов о работе сайта на основе данных из Prometheus
* Docker-compose позволяет удобно запускать все сервисы приложения одной командой

После деплоя и настройки приложения на сервере мы выпустили сертификат для подключения по https и настроили его автоматический перевыпуск с помощью certbot.

Конфигурация docker-compose файла:

version: '3.3'

services:

prometheus:

image: prom/prometheus:v2.14.0

volumes:

- ./prometheus/:/etc/prometheus/

restart: always

grafana:

image: grafana/grafana:6.5.2

ports:

- 3060:3000

restart: always

nginx:

build: ./nginx

volumes:

- static\_volume:/usr/src/district\_cities/static

- static\_volume:/usr/src/district\_cities/media\_root

ports:

- 127.0.0.1:9090:80

depends\_on:

- web

restart: on-failure

web:

build:

context: .

dockerfile: ./Dockerfile.prod

command: gunicorn district\_cities.wsgi --bind 0.0.0.0:8000 --worker-connections=1000 --workers=3

volumes:

- static\_volume:/usr/src/district\_cities/static

- static\_volume:/usr/src/district\_cities/media\_root

ports:

- 127.0.0.1:8080:8000

env\_file:

- envs/.env.prod

depends\_on:

- db

restart: on-failure

db:

image: postgres:12.0-alpine

volumes:

- postgres\_data:/var/lib/postgresql/data/

env\_file:

- envs/.env.prod.db

restart: on-failure

node\_exporter:

image: prom/node-exporter

ports:

- 127.0.0.1:9100:9100

restart: always

volumes:

postgres\_data:

static\_volume:

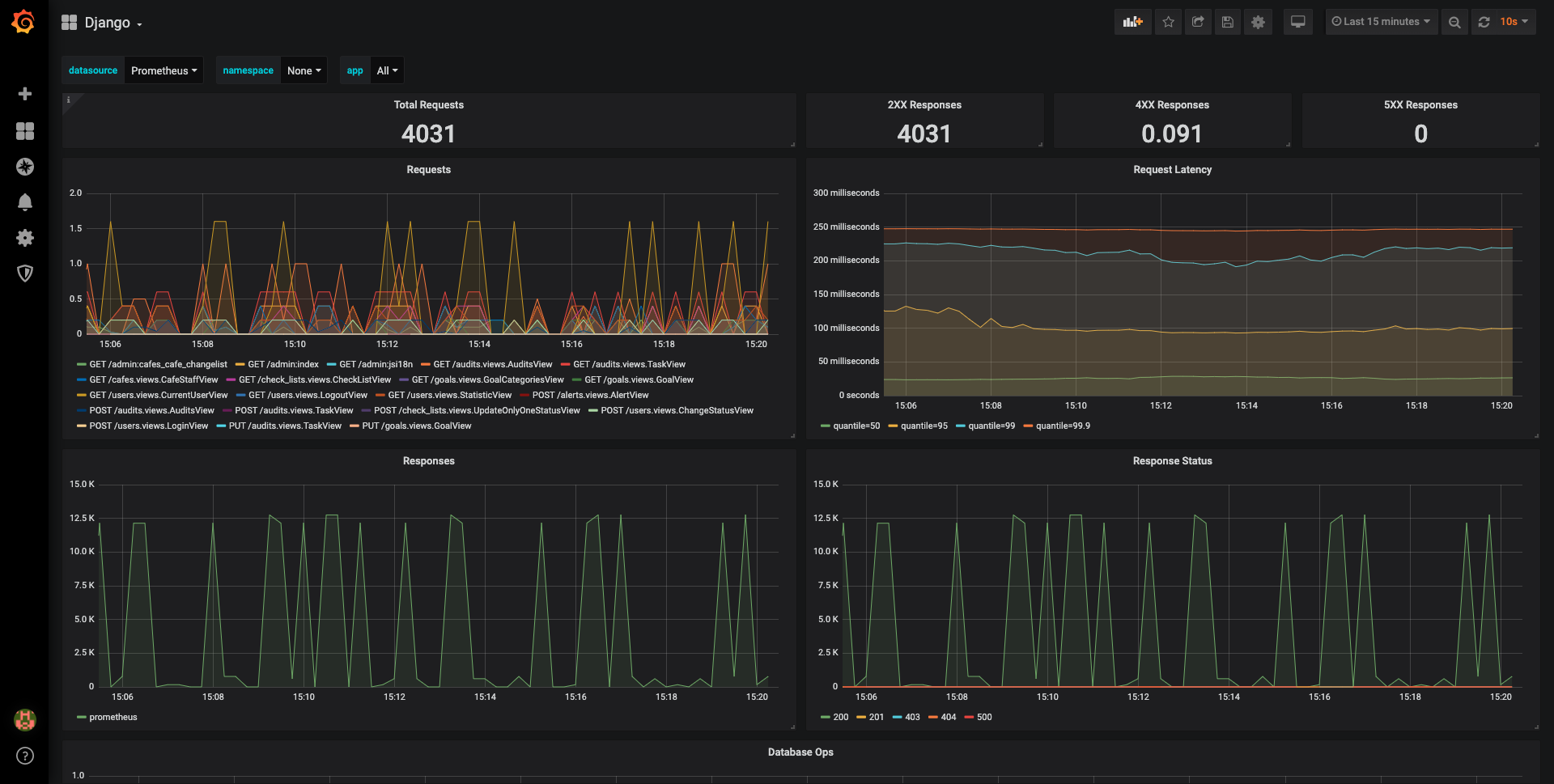
frontend\_volume:

Скриншоты дашборда мониторинга системы:

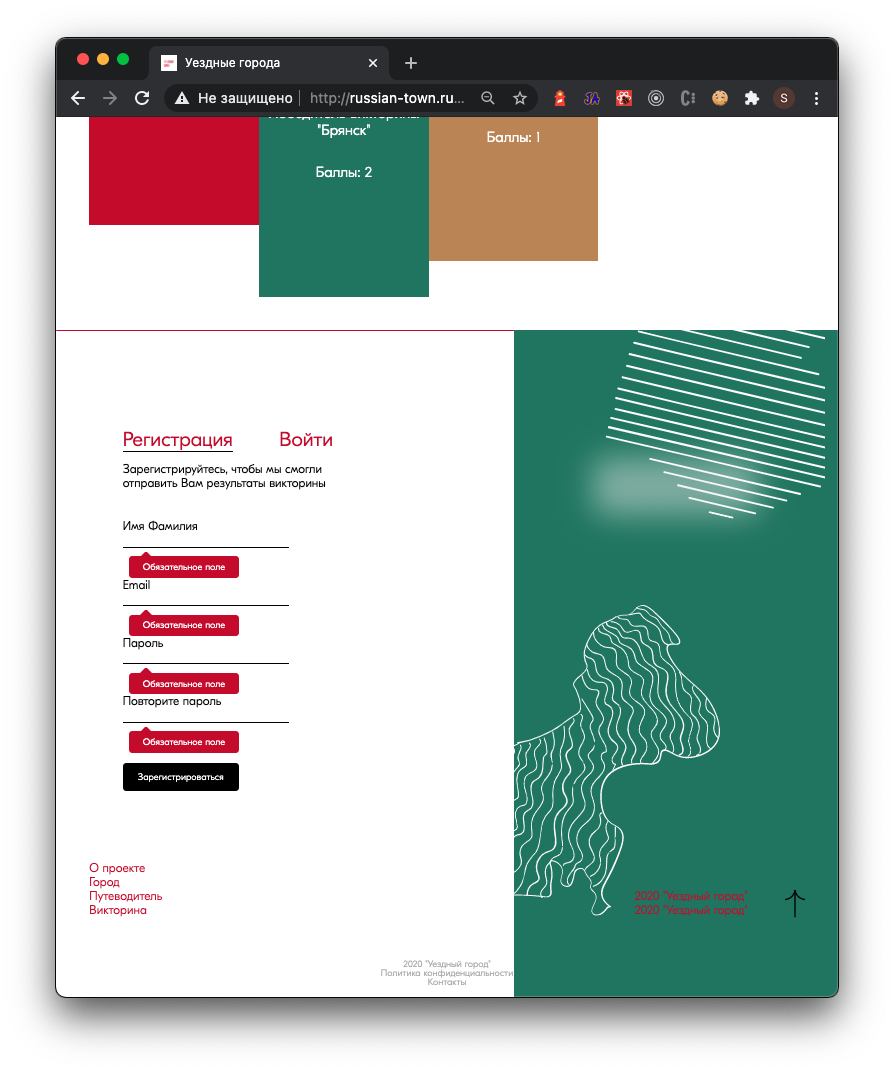
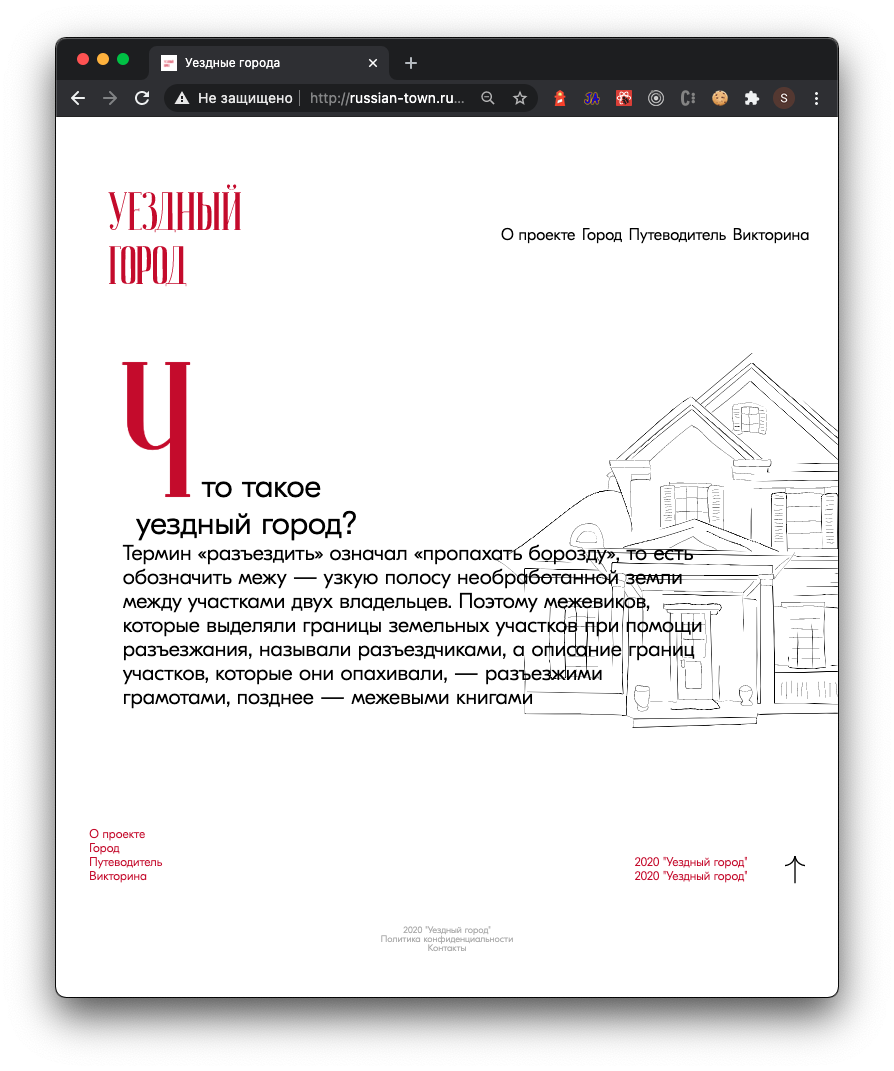
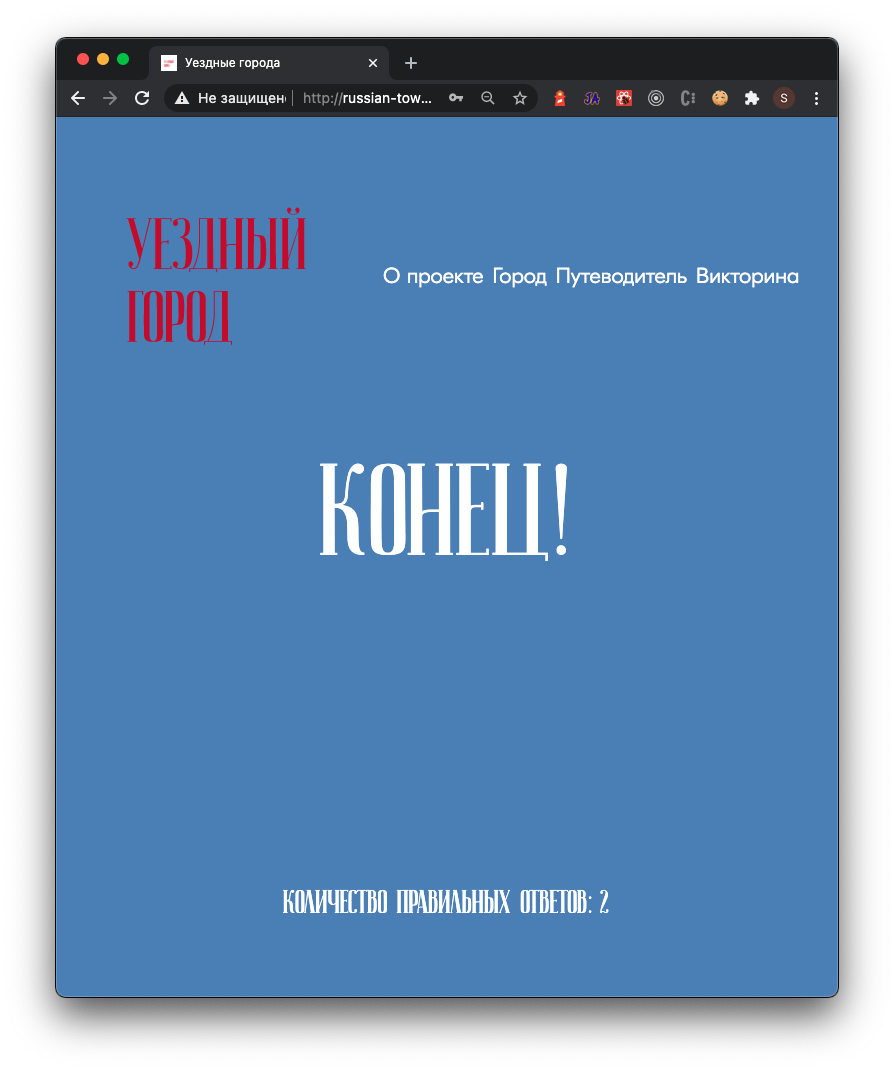
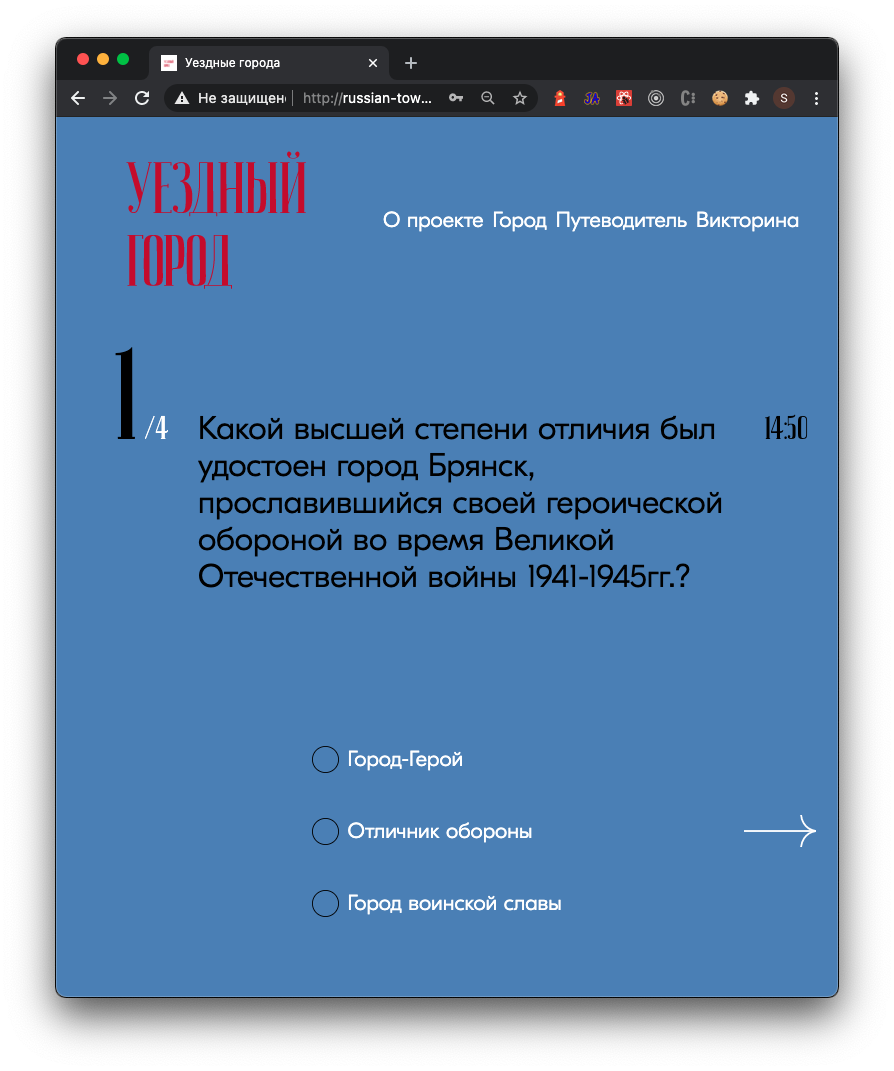
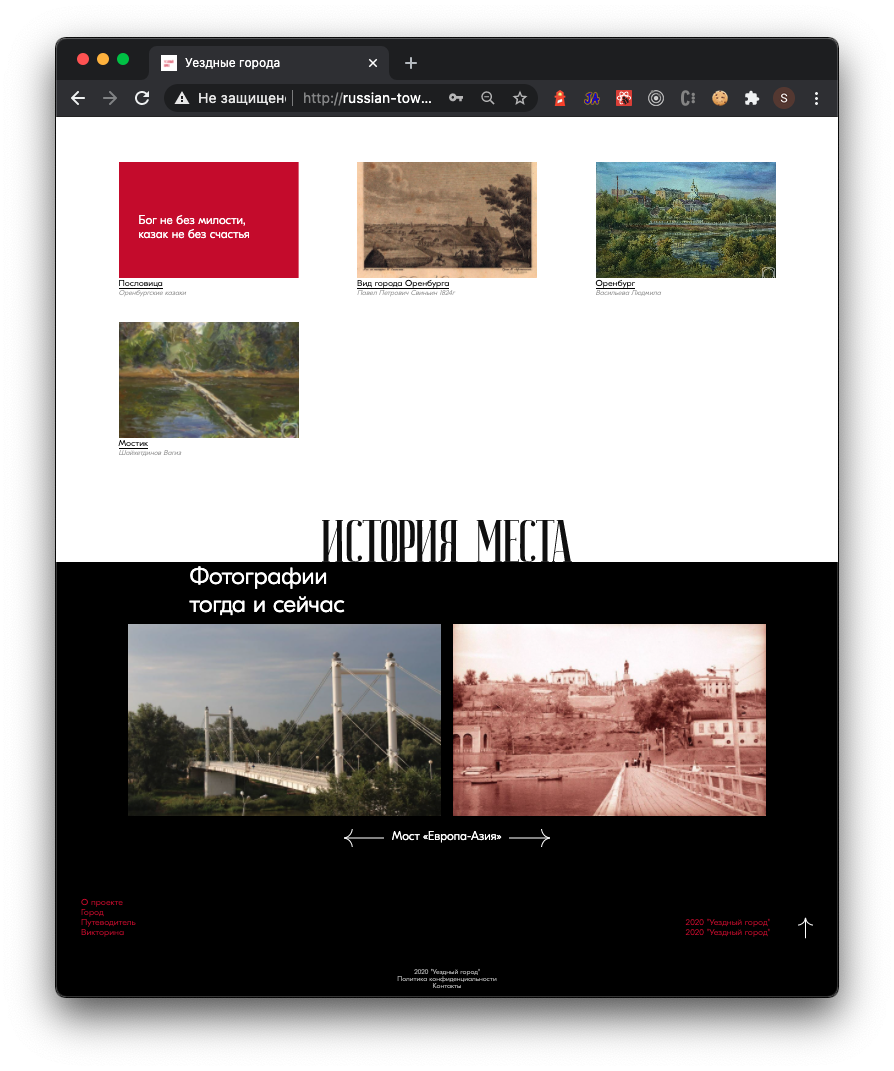
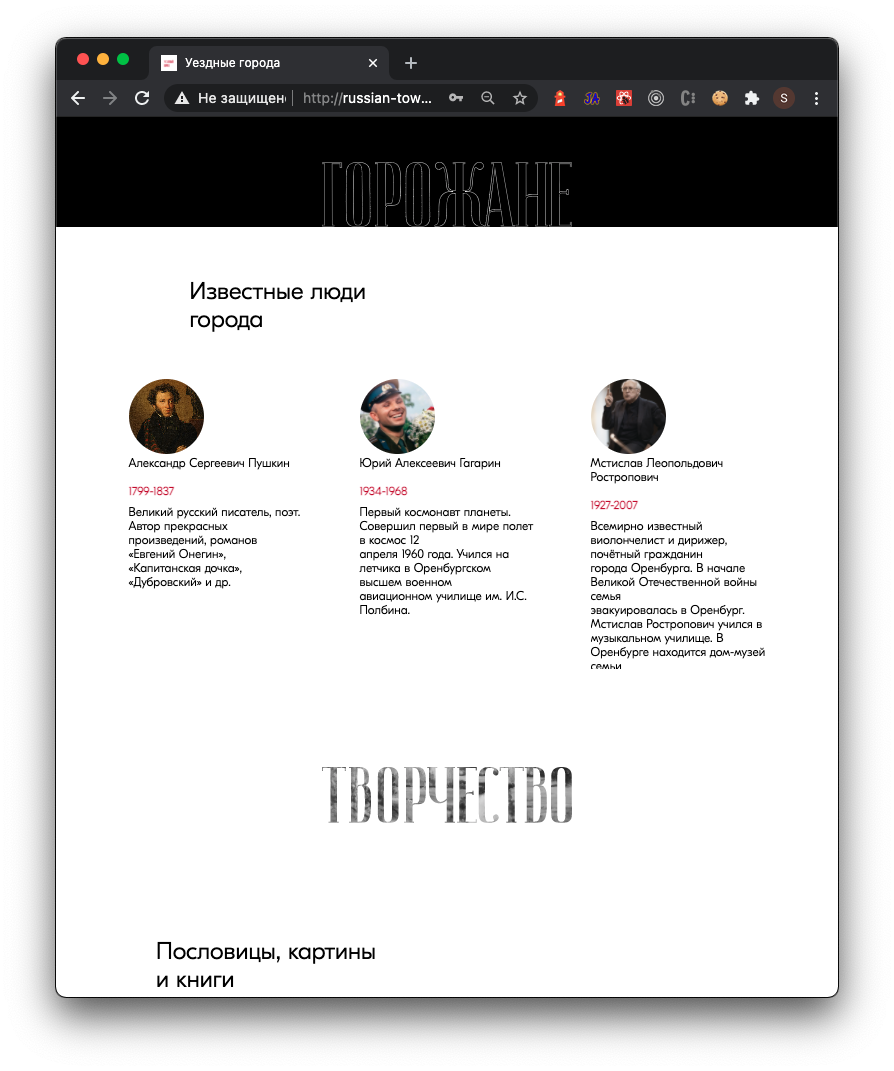
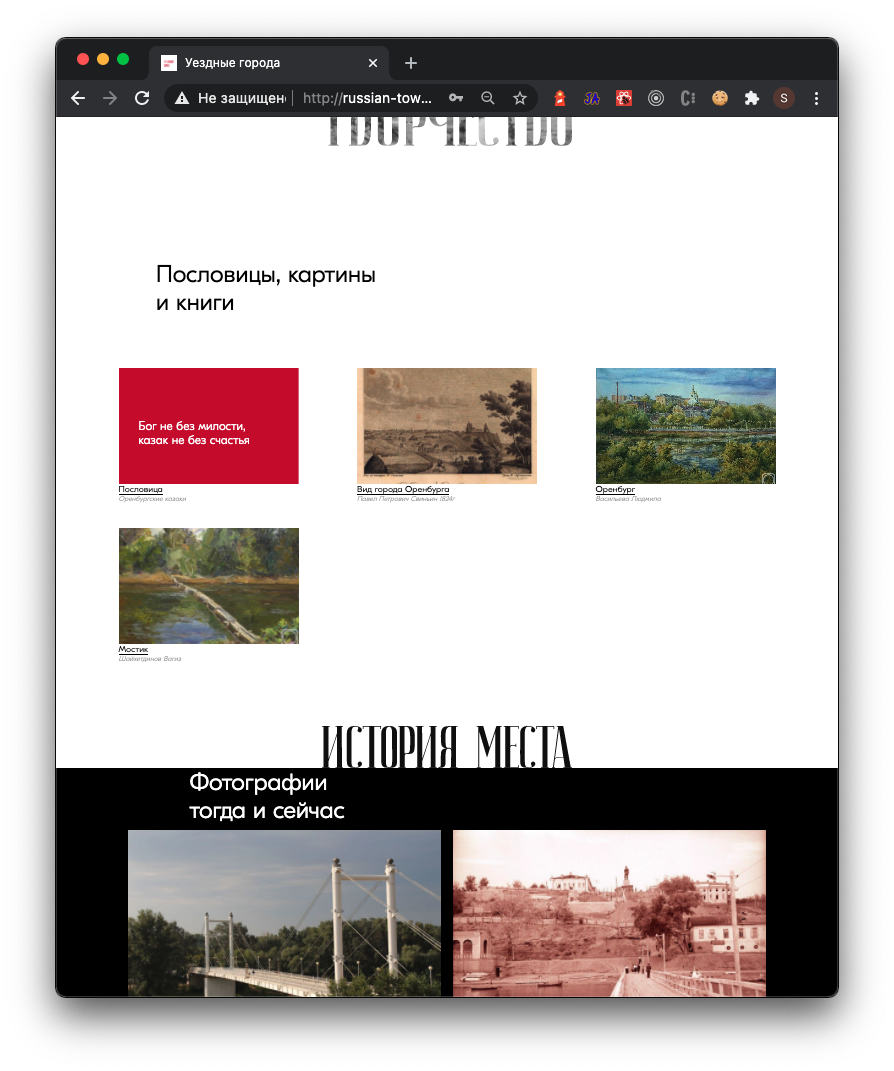
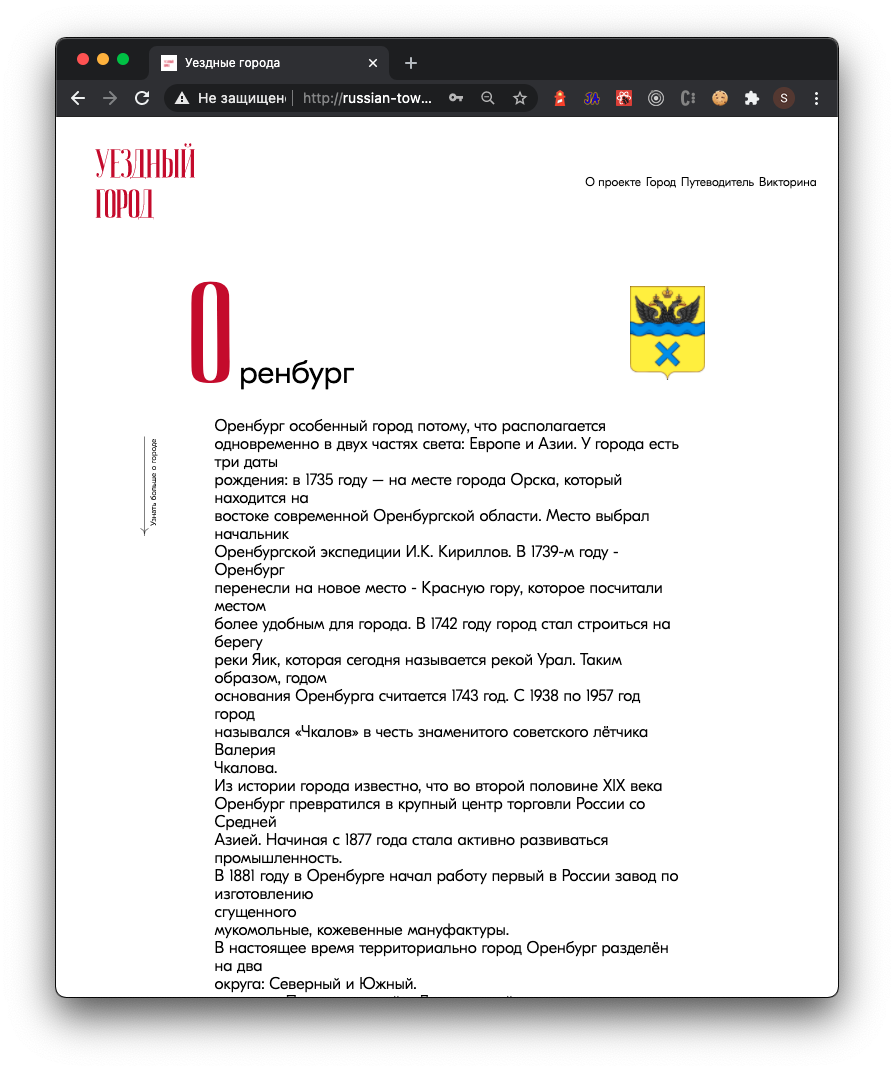
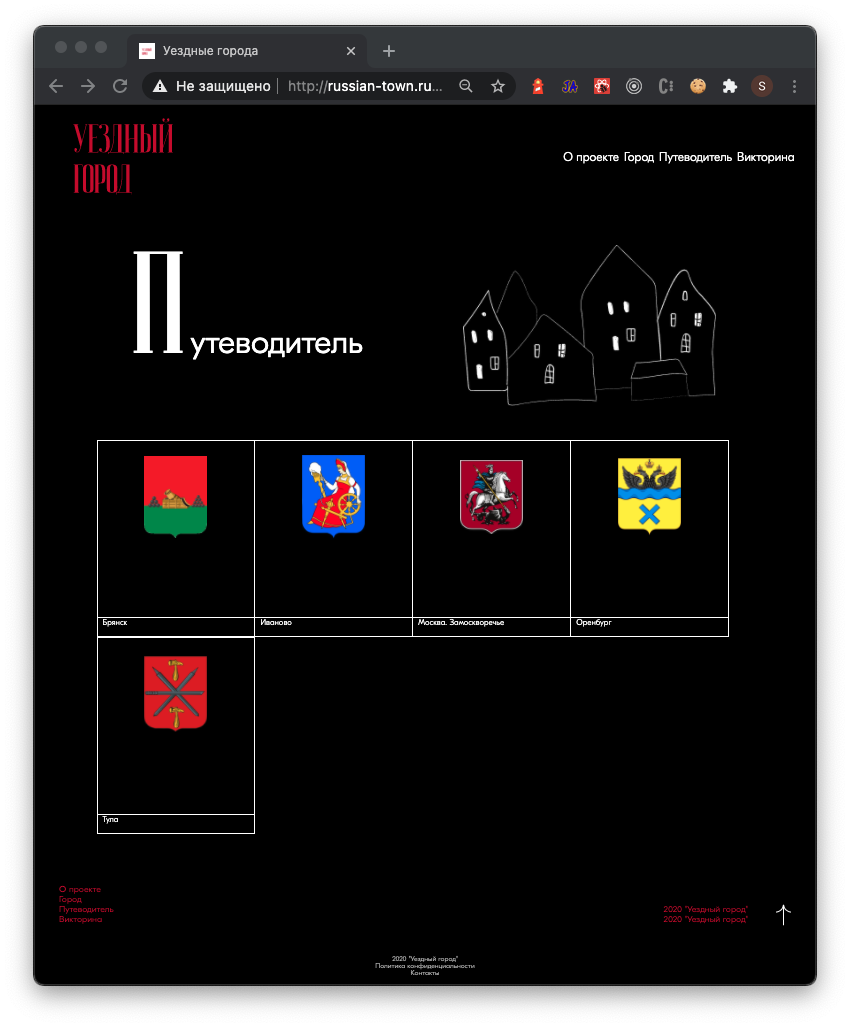
Состояние сервера(данные node)



Состояние Django(данные django)



**Скриншоты приложения**



**Ссылка на проект**

<http://russian-town.ru>