**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Web-технологии»**

Тема:Модуль пользователя приложения «Социальная сеть»

|  |  |  |
| --- | --- | --- |
| Студентка гр. 1381 |  | Васильева О. М. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2023

## Цель и задачи работы.

Целью работы является изучение основ языка TypeScript и особенностей применения фреймворка Angular для разработки web-приложений, ведения журналов ошибок, реализации взаимодействия приложений с использованием web-сокетов, организации модульного тестирования web-приложений с использованием Jest.

Для достижения поставленной цели требуется решить следующие задачи:

– разработка интерфейса web-приложения;

– создание web-сервера на основе express, настройка маршрутов, подго-

товка и обработка REST-запросов с учетом CORS (серверная часть);

– создание каркаса web-приложения с использованием Angular;

– определение перечня компонентов и сервисов web-приложения;

– создание шаблонов компонентов;

– регистрация и подключение в web-приложение журнала ошибок.

– создание web-сокета для отправки сообщений всем клиентам;

– создание и запуск Jest тестов для приложения.

**Основные теоретические сведения.**

Angular – фреймворк для разработки клиентских частей web-приложений, основанный на языке TypeScript. Фреймворк построен на использовании компонентного подхода, где каждый компонент может отображаться пользователю в соответствии с его индивидуальным шаблоном.

Фреймворк в том числе предлагает инструменты для управления из командной строки и организации тестирования.

CORS (Cross-Origin Resource Sharing) – это система, позволяющая отвечать на запросы из другого домена, отличного от домена происхождения запрашиваемого ресурса. Пример запроса CORS: приложение запущено на http://localhost:8080, а запросы отправляются на другой порт – http://localhost:3000.

Журналы ошибок позволяют контролировать появление ошибок как на этапе разработки, так и при работе пользователей. В качестве журналов ошибок предлагается использовать Rollbar (https://rollbar.com/) или Sentry (https://sentry.io/).

WebSocket – протокол связи, который может передавать и принимать одновременно сообщения поверх TCP-соединения, предназначен для обмена сообщениями между браузером и web-сервером, но может быть использован для любого клиентского или серверного приложения. Для создания web-сокетов предлагается использовать модуль Socket.IO (https://socket.io/).

Jest (https://jestjs.io/ru/) – это фреймворк для написания тестов серверной части web-приложений.

## Общая формулировка задание.

Необходимо создать web-приложение, обеспечивающее использование пользователем социальной сети. Пользователь может зарегистрироваться в социальной сети. Может добавить или удалить свою фотографию, может управлять своими друзьями в социальной сети, может добавить сообщение (новость) на свою страницу, может просматривать список новостей своих друзей.

Основные требования:

1. Приложение получает исходные данные из модуля администрирования приложения «Социальная сеть» в виде JSON-файла и работает одновременно с модулем администрирования приложения «Социальная сеть».

2. В качестве сервера используется Node.JS с модулем express.

3. Предусмотрены:

– HTML-страница для регистрации пользователя;

– HTML-страница для просмотра ленты новостей (пользователя и его

друзей);

– HTML-страница для добавления сообщения (новости).

4. Если пользователь является администратором, то у него есть возможность перехода в модуль администрирования приложения «Социальная сеть».

5. Переписка и страница новостей обновляются сразу после появления сообщений и новостей от пользователей без необходимости обновлять страницу целиком.

6. Разработаны тесты для серверной части web-приложения с использованием Jest.

7. Все элементы управления реализованы с использованием компонентов Angular. Взаимодействие между компонентами реализовано с использованием сервисов Angular.

8. Для реализации эффектов на HTML-страницах используются директивы Angular.

9. Для всех страниц web-приложения разработан макет интерфейса с использованием Figma (https://www.figma.com/).

Преимуществом будет использование звукового сопровождения событий: получений сообщений, появление новостей.

Преимуществом будет использование компонентов Angular Material (https://material.angular.io/).

## Выполнение работы.

1. Разработка шаблонов с помощью Figma.

Было создано три основных шаблона для сайта (главная регистрации, страница входа, главная страница). См. Рис. 1-2.

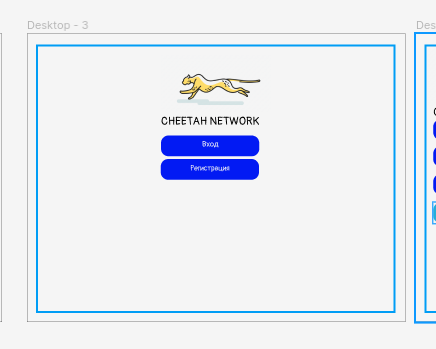


Рис. 1 — Шаблон страницы с использованием Figma

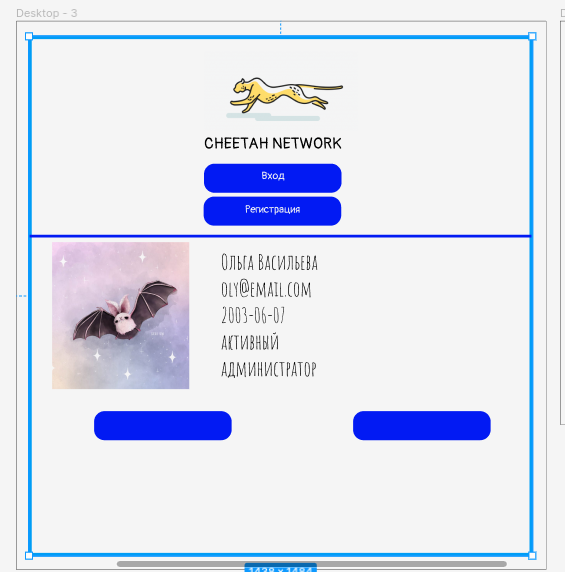


Рис. 2- Шаблон страницы с использованием Figma

2. Шапка регистрации/входа (app.component.html).

В данном файле прописана основная панель с кнопками вход и регистрация, которые отображаются на каждой странице сайта. Также на странице расположен название сайта и логотип.

3. Папка app.

В данной папке расположены основные компоненты сайта — такие папки как: login, register, main, news (каждая папка имеет одинаковую структуру- css файл, html файл и ts файл).

Cтраница login.component.html содержит два поля — почта и пароль, кнопка входа (см. рис. 3).

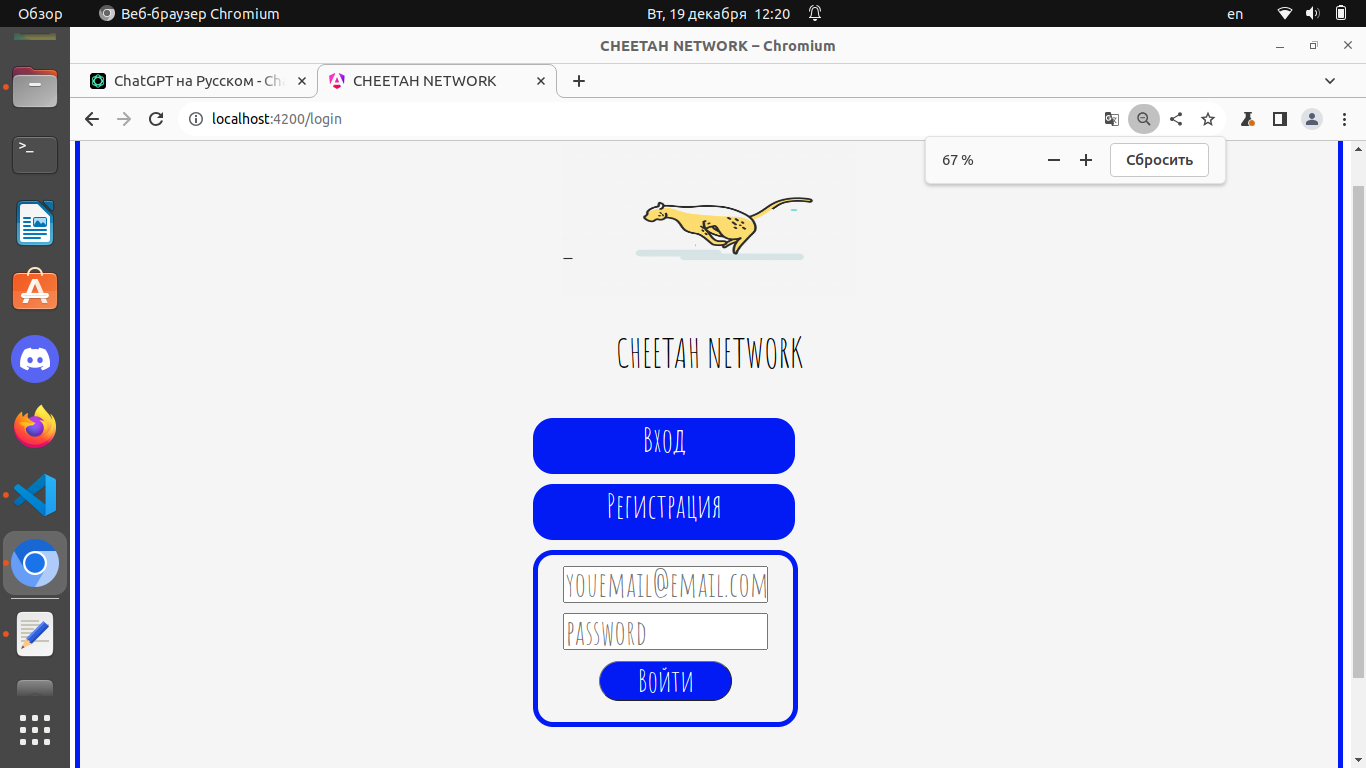
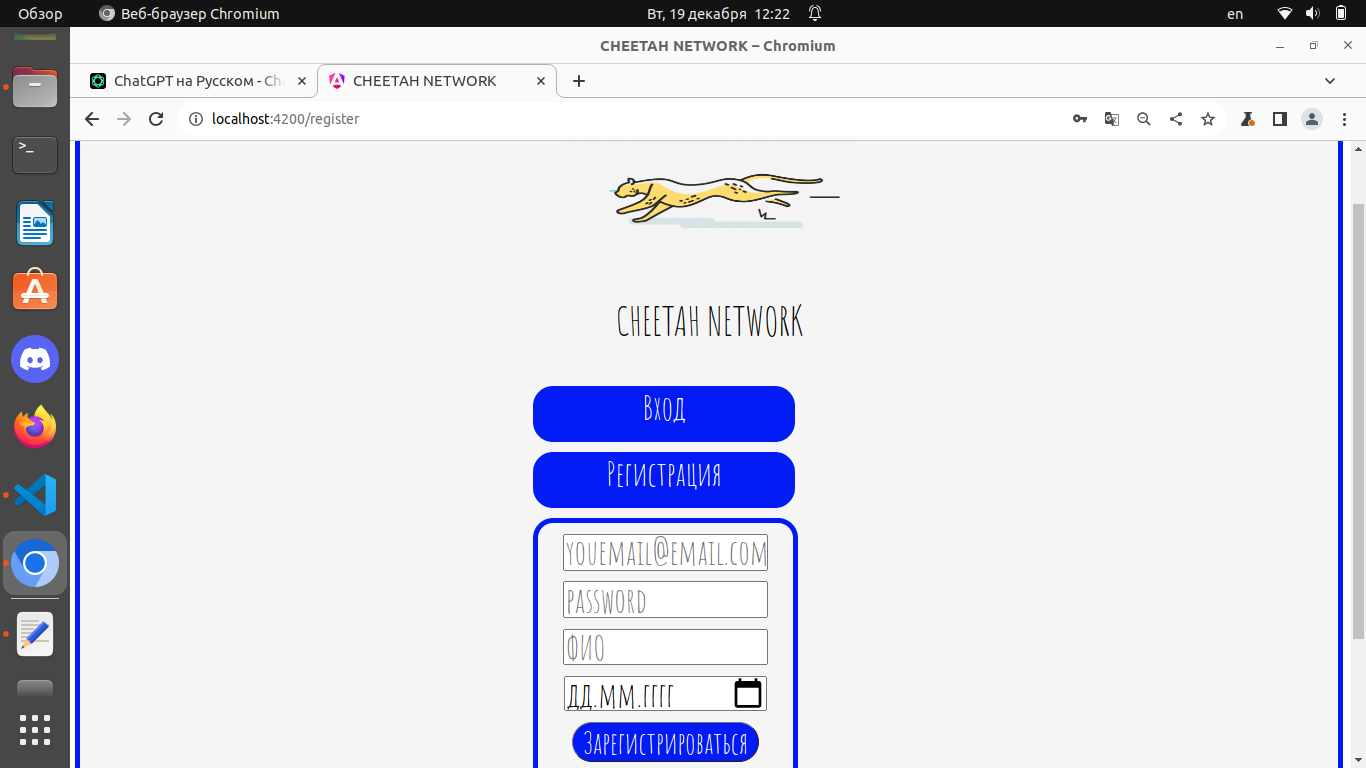


Рис. 3 — Страница входа

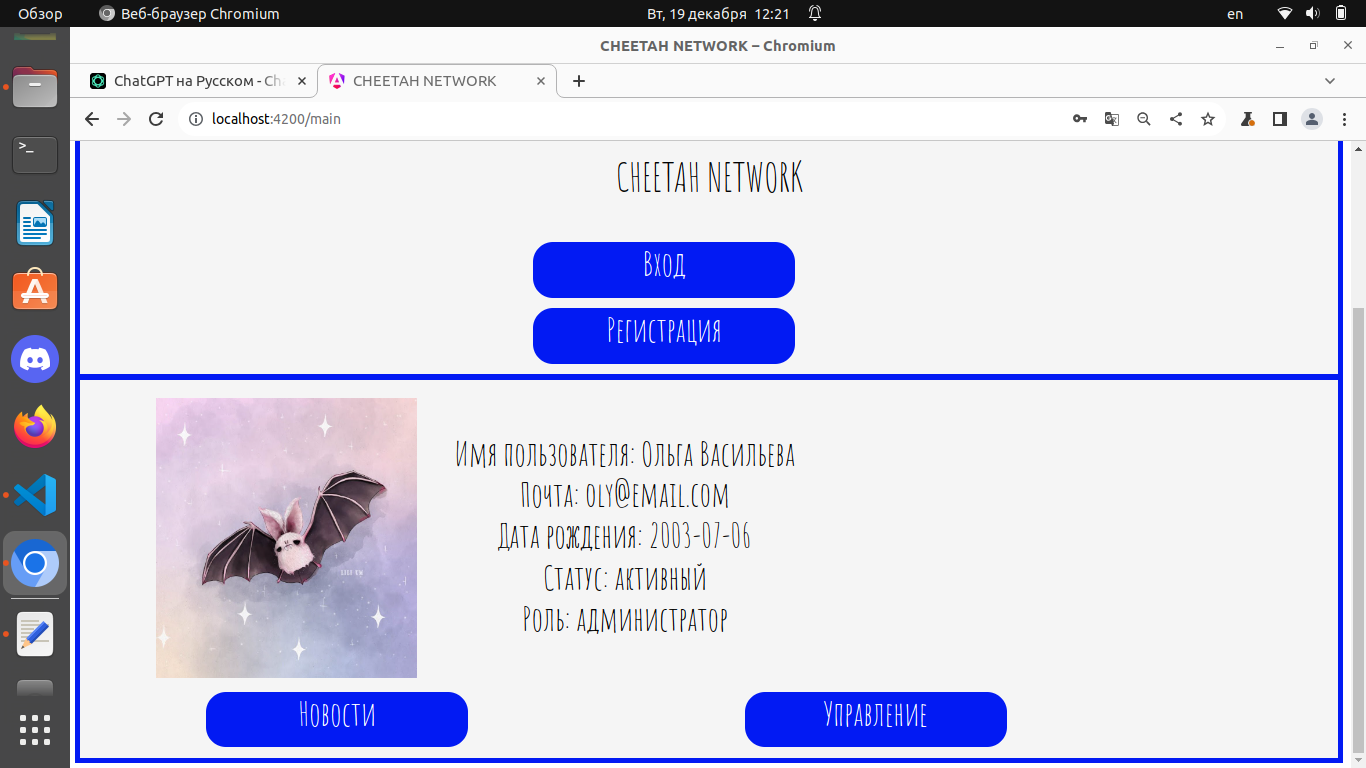
Файл login.component.ts описывает класс LoginComponent с его конструктором, который создает объект класса LoginService, и функцию при нажатии на кнопку входа (в функции происходит вызов функции класса LoginService, в которую передается пароль, логин и тексты,вызов alert в случае ошибки или успешного дальнейшего выполнения, ссылка на главную страницу).

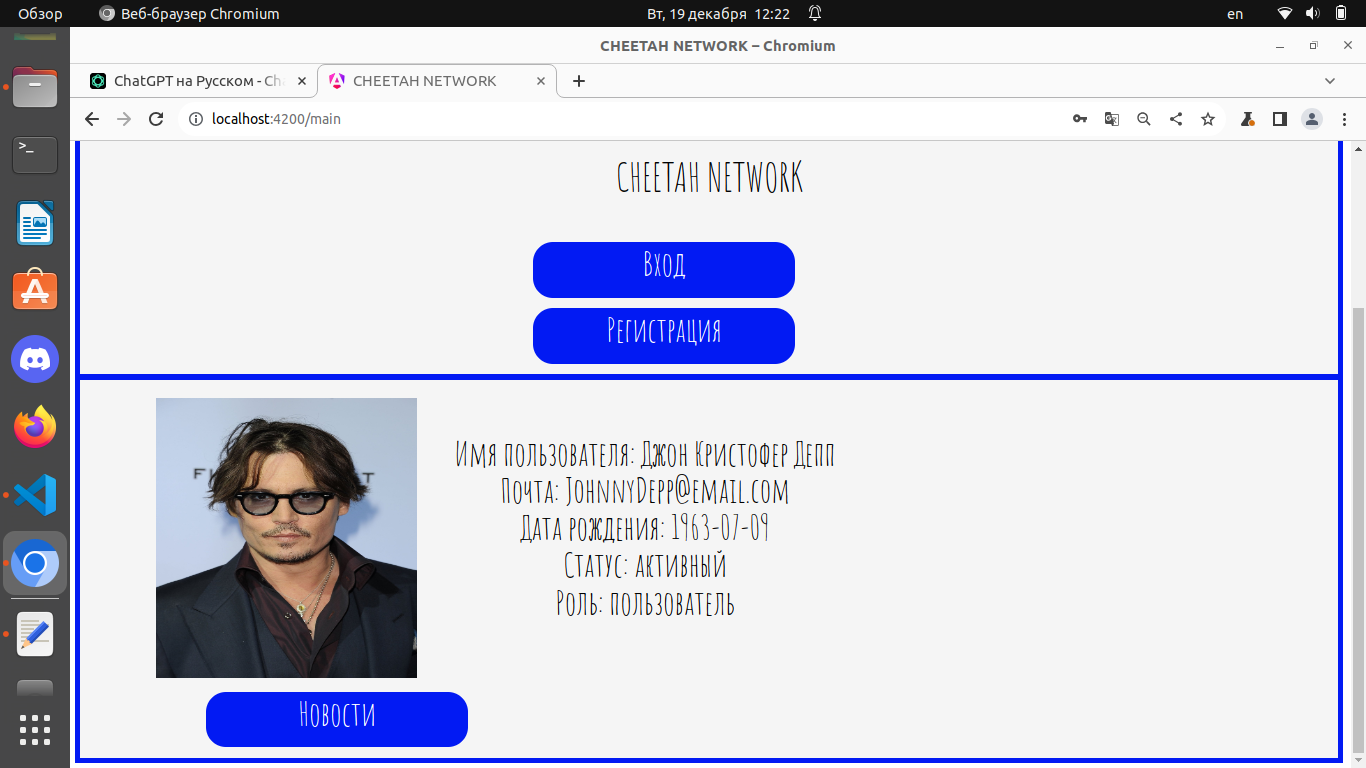
Файл register.component.ts имеет аналогичную структуру, что и login.component.ts за исключение того для объекта класса LoginService вызывается функция, которой передается данные созданного аккаунта, а дальше аналогичные параметры.

Cтраница register.component.html содержит поля для регистрации — почт,пароль, имя, дата рождения - кнопка входа (см. рис. 4).

Рис. 4 — Страница регистрации

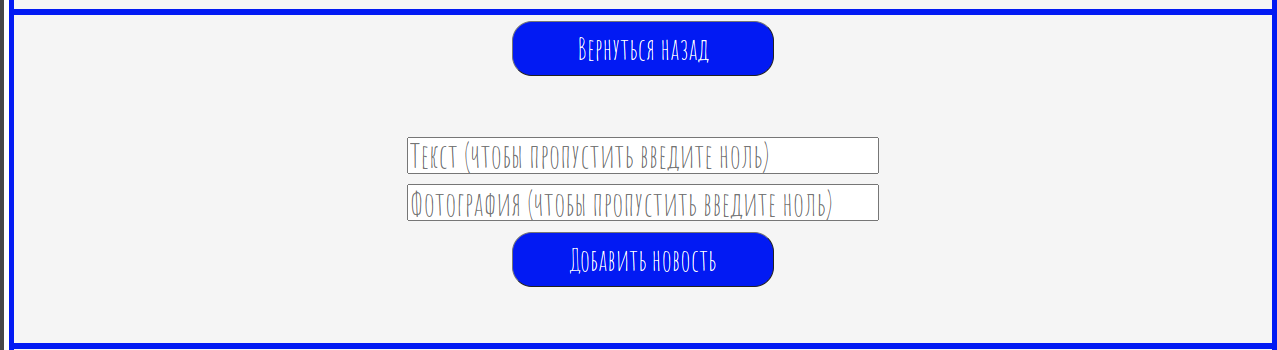
Страница main.component.html содержит информацию о пользователе (фото, почта, имя, дата рождения, статус и роль), ниже расположены кнопки — новости и управление (данная кнопка появляется только если пользователь является администратором). см. рис. 5-6.

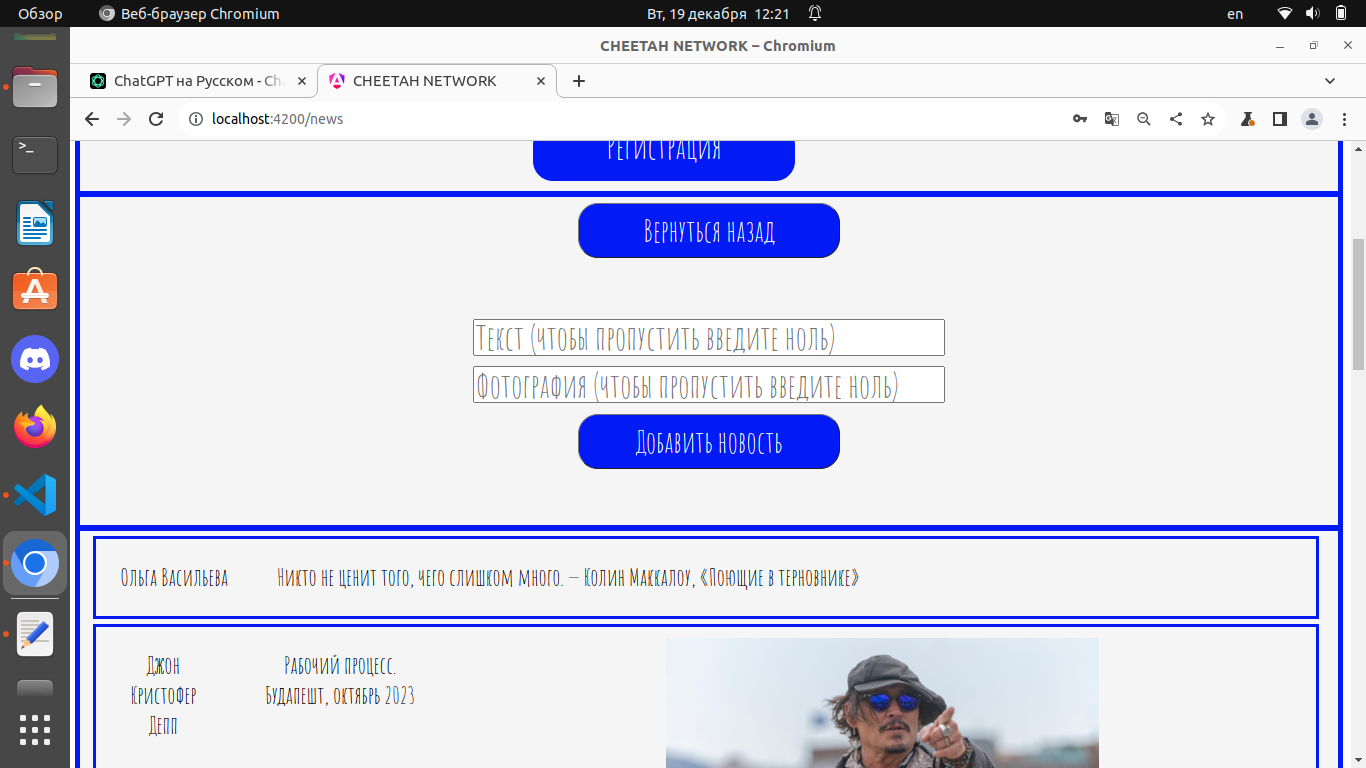
Рис. 4 — Главная страница для администратора

Рис. 5 — Главная страница для простого пользователя

Файл main.component.ts описывает класс MainComponent с его конструктором, который создает объект класса MainService и также определяется роль пользователя. Функция класса вызывает два метода для объекта класса MainService — взятие фотографии пользователя и взятие информации.

Страница news.component.html выводит поле для добавление новости и список новостей друзей пользователя и самого пользователя, также кнопка вернуться назад (см. рис. 6-7).

Рис. 6 — Поля для добавления новости

Рис. 7 — Страница новостей пользователя

Файл news.component.ts описывает класс NewsComponent с его конструктором, который создает объект класса NewsService и объект класса MainService. Также в классе имеется три функции (каждая из них вызывает методы классов созданных объектов): первая берет новости, вторая добавляет, а третья возвращается на главный экран.

4. Основные сервисы в папке app.

Данная папка также содержит сервисы: start.service.ts, login.service.ts, news.service.ts.

В файле start.service.ts описан класс MainService, в котором описаны методы для взятия фото и информации пользователя для отображения на главную страницу. Метод getPhoto использует метод post объекта HttpClient для отправки POST-запроса на URL-адрес '<http://localhost:4300/user_photo>' с объектом {id} в качестве тела запроса. Он подписывается на ответ и возвращает полученное фото в виде строки через функцию getting. Метод getInfo также использует метод post объекта HttpClient для отправки POST-запроса на URL-адрес '<http://localhost:4300/user_info>' с объектом {id} в качестве тела запроса. Он подписывается на ответ и возвращает информацию о пользователе (имя, электронная почта, дата, статус, роль) через функцию gettingInfo. Интерфейсы Photo и Info используются для типизации данных, которые ожидаются получить в ответ на запросы.

В файле login.service.ts в классе LoginService определены два метода: login и register. Метод login принимает email и password пользователя, а также две функции обратного вызова - on\_login и on\_error. Он отправляет POST-запрос на сервер по указанному адресу "<http://localhost:4300/login>" с параметрами email и password. В ответе ожидается объект типа id\_Response, где содержится поле id, error и is\_admin. Если поле error не содержит ошибки (пустое или равно нулю), то в сессионное хранилище записываются значения id и is\_admin, затем вызывается функция on\_login(). Если поле error содержит ошибку, то вызывается функция on\_error(), которой передается сообщение об ошибке. Метод register похож на метод login. Он также принимает email, nameN, date и password пользователя, а также две функции обратного вызова - on\_register и on\_error. Он отправляет POST-запрос на сервер по указанному адресу "<http://localhost:4300/register>" с параметрами email, password, nameN и date. В ответе ожидается объект типа id\_Response, где содержится поле id и error. Если поле error содержит ошибку, то вызывается функция on\_error(), которой передается сообщение об ошибке. Если ошибки нет, то вызывается функция on\_register(). Интерфейс id\_Response задает структуру данных, которую ожидается получить в ответе от сервера. Состоит из трех полей: id (число), error (строка) и is\_admin (логическое значение).

В файле news.service.ts внутри класса NewsService определены два метода: get\_news и add\_news. Метод get\_news получает новости с заданным идентификатором (id) от сервера. Он отправляет POST-запрос на URL-адрес "<http://localhost:4300/news>" с параметром id, и в случае успешного ответа вызывает колбэк-функцию getting\_news, передавая полученные новости в качестве аргумента. Если ответ содержит ошибку, вызывается колбэк-функция on\_error, передавая сообщение об ошибке. Метод add\_news добавляет новость на сервер. Он отправляет POST-запрос на URL-адрес "<http://localhost:4300/add_news>" с параметрами id, news\_text и news\_photo, и в случае успешного ответа вызывает колбэк-функцию on\_add. Если ответ содержит ошибку, вызывается колбэк-функция on\_error, передавая сообщение об ошибке. Также в классе NewsService определен метод onGetNews, который устанавливает подписку на событие 'news' от сервера. С помощью метода socket.on() из библиотеки Socket.IO устанавливается обработчик события 'news', который вызывает переданную ему функцию socketfunc(). Это позволяет клиенту реагировать на новости, полученные от сервера. И наконец, определен интерфейс News, который описывает структуру объекта новости с полями user\_name, news\_text и news\_photo.

5. Файлы server.js и routes.js.

Первый файл является сервером, а второй файл описывает основные маршруты сайта, также описывает функцию сохранения, получения id по адресу почты и проверку на корректность ввода постового адреса при регистрации.

6. Файл test.js.

В данном файле тестируем две функции: получения id по адресу почты и проверку на корректность ввода постового адреса при регистрации. См. Рис. 8

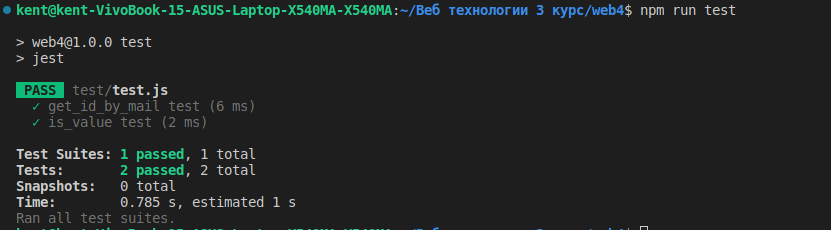


Рис. 8 — Результат тестирования

## Вывод.

В ходе лабораторной работы были изучены основы языка TypeScript и особенности применения фреймворка Angular для разработки web-приложений, ведения журналов ошибок, реализации взаимодействия приложений с использованием web-сокетов, организации модульного тестирования web-приложений с использованием Jest.