Основные функциональности

**1.1. Регистрация и аутентификация пользователей**

Аутентификация пользователей должна осуществляться через логин/пароль или OAuth 2.0.

**1.2. Профиль пользователя**

Возможность редактирования профиля пользователя, включая имя, фотографию, биографию и навыки.

Возможность просмотра профиля других пользователей.

**1.2.1. Обязательные поля профиля**

Имя и фамилия

Электронная почта

Пароль

Фото профиля

Краткое описание о себе

**1.2.2. Дополнительные поля профиля**

Страна и город проживания

Ссылки на профили в социальных сетях

Портфолио (ссылки на работы или проекты)

Опыт работы

Навыки и компетенции

Интересы и хобби

**1.2.3. Возможности редактирования профиля**

Пользователи должны иметь возможность редактировать свой профиль.

Некоторые поля, такие как имя, фамилия и электронная почта, могут быть недоступны для редактирования.

**1.2.4. Поиск пользователей**

Добавить возможность поиска пользователей по названию компании, опыту работы и навыкам.

Результаты поиска должны отображаться на отдельной странице, где можно просмотреть профиль каждого пользователя.

**1.2.5. Подписки на пользователей**

Добавить возможность подписки на пользователей, чтобы пользователи могли следить за обновлениями профиля других пользователей.

Новые обновления профиля должны отображаться на специальной странице.

**1.2.6. Рейтинг пользователей**

Реализовать возможность оценки пользователей другими пользователями, чтобы облегчить поиск работы или партнеров.

Добавить возможность просмотра рейтинга каждого пользователя на странице его профиля.

**1.2.7. Чаты с пользователями**

Добавить возможность общения между пользователями через чаты, которые должны быть доступны с страницы профиля каждого пользователя.

**1.3. Статьи**

Возможность создания, редактирования и удаления статей пользователей.

Статьи должны иметь заголовок, описание, теги и содержимое.

Возможность поиска статей по заголовку, тегам и содержимому.

Возможность комментирования статей другими пользователями.

**1.4. Сообщества**

Возможность создания, редактирования и удаления сообществ.

Сообщества должны иметь название, описание и теги.

Возможность присоединения к сообществам другими пользователями.

Возможность поиска сообществ по названию и тегам.

Возможность просмотра статей и чатов, связанных с сообществом.

**1.5. Чаты**

Возможность создания, редактирования и удаления чатов пользователей.

Чаты должны иметь название и список участников.

Возможность отправки текстовых сообщений в чате.

**Технические требования**

**2.1. Backend**

Бэкенд должен быть написан на языке программирования Python с использованием фреймворка Django.

Для базы данных использовать PostgreSQL.

**2.2. Frontend**

Фронтенд должен быть написан на языке программирования JavaScript с использованием фреймворка React.

Для управления состоянием приложения использовать Redux.

**2.3. Аутентификация и авторизация**

Для аутентификации и авторизации пользователей использовать библиотеку Django REST framework.

Использовать JWT-токены для аутентификации и авторизации пользователей.

**2.4. API**

Создать RESTful API для взаимодействия между фронтендом и бэкендом.

Документировать API с помощью Swagger или OpenAPI.

**Дизайн**

**3.1. Общий стиль**

Использовать современный дизайн, удобный для пользователей.

Стиль должен быть согласован с темой сообщества онлайн работников.

**3.2. Адаптивный дизайн**

Обеспечить адаптивный дизайн, чтобы сайт корректно отображался на различных устройствах (компьютерах, планшетах, мобильных телефонах).

**3.3. UI элементы**

Использовать стандартные UI элементы, чтобы пользователи быстро могли разобраться в интерфейсе.

Обеспечить хорошую читаемость контента, выбрав соответствующие шрифты и цвета.

Безопасность

**4.1. Защита от XSS и CSRF атак**

Обеспечить защиту от атак типа XSS (межсайтовый скриптинг) и CSRF (межсайтовая подделка запроса).

Использовать механизмы защиты, предоставляемые Django и React.

**4.2. Защита паролей**

Хранить пароли пользователей в зашифрованном виде, используя хеширование и соль.

Ограничить число неудачных попыток ввода пароля, чтобы предотвратить атаки перебором.

**Тестирование**

**5.1. Юнит-тестирование**

Написать юнит-тесты для всех моделей, представлений и сериализаторов.

Использовать фреймворк pytest для тестирования.

**5.2. Интеграционное тестирование**

Написать интеграционные тесты для проверки взаимодействия различных компонентов.

Использовать фреймворк Selenium для автоматизированного тестирования веб-интерфейса.

**Развёртывание**

**6.1. Хостинг**

Разместить приложение на облачном сервисе, таком как Heroku, AWS или Google Cloud.

Обеспечить автоматическое развертывание приложения с помощью Continuous Integration/Continuous Deployment (CI/CD).

**6.2. HTTPS**

Обеспечить защищенное соединение с помощью протокола HTTPS.

**Расширяемость**

**7.1. API**

Обеспечить API для интеграции с другими сервисами и приложениями.

Использовать формат данных JSON.

**7.2. Плагины**

Реализовать возможность создания плагинов для расширения функциональности приложения.