МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Ярославский государственный университет имени П. Г. Демидова»

Кафедра компьютерных сетей

Сдано на кафедру
«_____»_____ 2022 г.
Заведующий кафедрой,
д. ф.-м. н., профессор

_____ С. Д. Глызин

Выпускная квалификационная работа

Разработка сервиса централизованного управления паролями

по направлению
01.03.02 Прикладная математика и информатика

Научный руководитель
к. ф.-м. н., доцент

_____ С. В. Алешин
«_____»_____ 2022 г.

Студент группы ИВТ-41БО

_____ Н. С. Батогов
«_____»_____ 2022 г.

Ярославль, 2022

Реферат

Объем 12 с., 1 гл., 0 рис., 0 табл., 9 источников, 1 прил.

Ключевые слова: **управление паролями, пароль, генерирование пароля, хранение паролей, безопасность компании, кибербезопасность, защита данных, одностраничное приложение.**

Объект исследования - централизованное корпоративное управление паролями.

Цель работы - создание сервиса для удобного, безопасного и централизованного управления паролями в компании.

В результате работы был спроектирован, реализован и протестирован сервис, который позволяет решить данную задачу.

Область применения - небольшая компания, которой нужно удобно организовать пароли для почты, сайтов, серверов, карт и других ресурсов.

Содержание

Введение

В настоящее время разработка веб-приложений является одной из самых перспективных отраслей современного IT. Веб-приложение - это полноценная программа, которую пользователи использует через браузер. Именно поэтому данный вид разработки получил такой большой интерес в последнее время, ведь браузеры есть у каждого. Многие предприниматели переносят свой бизнес в диджитал сферу и выбирают для этого именно веб-приложение. Почти любое веб-приложение можно реализовать в короткие сроки по сравнению с обычным десктопным приложением, а так же любое такое приложение будет кроссплатформенным, ведь оно работает только в браузере клиента. С помощью такого метода программирования можно с легкостью реализовывать даже самые сложные бизнес-процессы и задумки. В рамках данной работы мы будем использовать термин "сервис".

Современный Интернет невозможно представить без паролей, систем шифрования, аутентификации и авторизации пользователей. Одна из самых важных частей каждого человека в сети - это его пароли, но, к сожалению, достаточно много людей не хотят уделять внимание их безопасности. Когда же дело касается безопасности корпоративных паролей,то разговор идет не только о надежной защите данных конкретного пользователя, но и о сохраннении имиджа и репутации всей компании. Одновременно с этим, многие компании в своей работе используют незащищенные и неудобные инструменты управления и хранения паролей. Не стоит забывать, что от правильной тактики хранения паролей зависит и уровень защиты от кибератак. Для компании критически важно, чтобы у администратора системы хранения паролей было понимание того, у кого есть тот или иной доступ к информации. Многие используют слишком примитивные способы хранения паролей, например текстовые файлы на рабочем столе компьютера, письма или документы в облаке. Для того, чтобы избежать таких способов и грамотно управлять корпоративными паролями требуется специальный инструмент, заточенный под конкретные нужды компании. Такие инструменты называют менеджерами паролей. В большинстве случаев, компании предпочитают рациональный, надежный и удобный подход к внутреннему приложению собственной команды. Следовательно, наилучшим способом реализации сервиса для централизированного управления паролями можно считать именно веб-приложение. Объектом исследования является централизованное корпоративное управление паролями.

Предметом исследования является сервис для безопасного, централизованного и удобного управления паролями.

Целью выпускной квалификационной работы является создание сервиса для удобного, безопасного и централизованного управления паролями в ком-

пании. Данный сервис позволит небольшим компаниям грамотно работать со своими паролями, а так же не переживать за безопасность их хранения.

Для достижения этой цели необходимо решить следующие задачи:

- произвести аудит существующих сервисов для хранения паролей(менеджеров паролей)
- создать клиентскую часть(интерфейс) сервиса
- создать серверную часть сервиса

???

1. Разработка сервиса

1.1. Общая архитектура сервиса

Приложение написано в стиле SPA(Single page application). SPA - это одностраничное приложение, которое работает в браузере и не требует перезагрузки за счет динамического обновления контента с помощью технологии AJAX(Asynchronous Javascript and XML). При использовании AJAX мы получаем много приемуществ, например:

1) активное взаимодействие пользователя с веб-страницей(интерактивность)
2) плавная работа с приложением
3) удобное использование всей страницы
4) уменьшение трафика пользователя
5) снижение нагрузки на сервер
6) положительное влияние на пользовательский опыт

На рис. 1 можно посмотреть отличия классического сайта от приложения в стиле SPA. В нашем сервисе данные передаются через формат JSON.

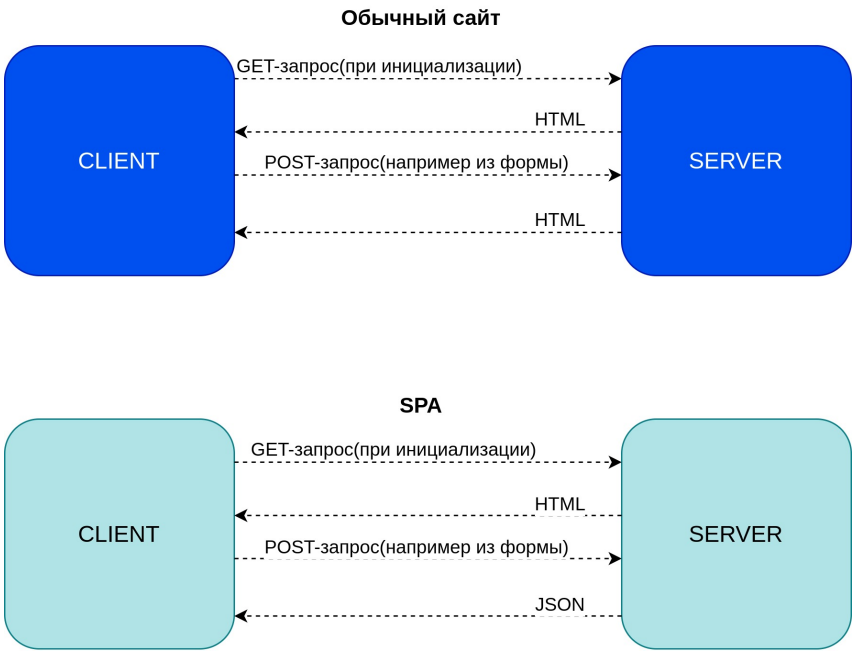

Рис. 1 — Отличия сайта от SPA

Для написания в данном стиле программисту желательно использовать специальную библиотеку для написания клиентского кода. Самые полюбившиеся сообществом JavaScript-библиотеки для такой задачи на рынке сейчас представлены на рис. 2.

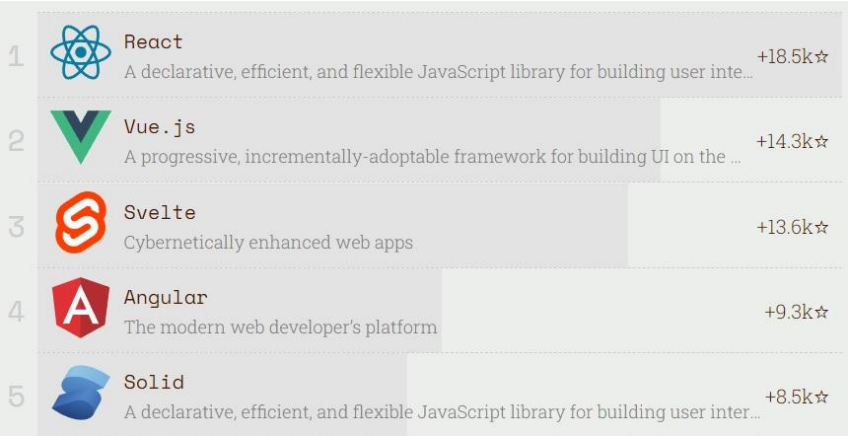

Рис. 2 — библиотеки JavaScript с наибольшим количеством звезд GitHub

Таким образом, все вышеперечисленные достоинства одностраничного приложения делают его оптимальным выбором для реализации сервиса.

1.2. Серверная часть

В качестве языка программирования мы выбрали TypeScript - это язык программирования для веб-разработки, основанный на JavaScript, который позволяет писать статически типизированный код и следовать лучшим ООП практикам.

Для написания backend-части сервиса использовался Nest.js - это мощный и гибкий фреймворк для написания эффективного серверного кода. Он построен на Express.js - самом популярном решении для создания серверного кода для платформы Node.js. Использование фреймворков помогает нам писать меньше шаблонного кода и сосредоточиться на решении конкретных бизнес-задачах. Преимущества решения:

1) обеспечивает заранее масштабируемую и чистую архитектуру, требует от программиста писать хороший код
2) поддерживает множество инструментов “из коробки”
3) присутствуют мощные инструменты командного интерфейса для быстрой разработки модулей
4) хорошая реализация инъекции зависимостей(Dependency Injection) из коробки
5) поддержка сообщества

Создатели данного фреймворка вдохновлялись Angular и многие вещи написаны с использованием декораторов. Декораторы - это способ “оборачивания” функций, они позволяют модифицировать поведение классов.

Для хранения данных в сервисе использовалась СУБД PostgreSQL. Для связи базы данных и объектов в приложении использовалась технология ORM(Object-Relational Mapping - объектно-реляционное отображение) и ее конкретная реализация - Sequelize ORM.

Для решения аутентификации, авторизации и контроля доступа использовался продукт Keycloak. Преимущества решения:

1) единый вход в приложение(SSO)
2) LDAP/Active Directory интеграция и быстрый перенос пользователей
3) хороший и документированный API

Документация конечных точек API для серверной части задокументирована с помощью OpenAPI 3.0 Swagger.

1.3. Клиентская часть

Для клиентской части приложения использовался язык программирования JavaScript и фреймворк Vue.js 2.

Для интерфейсов средней сложности это идеальный выбор, потому что данное решение легко масштабировать, у него прекрасная документация, а так же он очень производительный.

В роли менеджера состояния у нас выступает библиотека и одновременно паттерн управления состоянием Vuex 3. Это самое популярное решение для Vue, потому что данные хранятся с правилами, которые гарантируют изменение только предсказуемым образом. По мере роста приложения возникает потребность передавать данные между компонентами и это начинает приносить неудобства в использовании данных. Vuex призван решить эту проблему. В нем данные хранятся централизовано.

Управление маршрутами происходит с помощью Vue Router - официального роутера для Vue.js.

В качестве UI библиотеки для написания компонентов интерфейса мы использовали библиотеку Vuetify 2.6. Данное решение хорошо документировано и позволяет писать масштабируемые компоненты.

Заключение

В заключении подводятся итоги выполненной работы, рассказывается о том, что удалось и что не удалось сделать, описываются перспективы продолжения исследований.

Перспективы исследования. В дальнейшем планируется создание дополнительного функционала, например создаение публичных паролей с разовыми токенами для авторизации, чтобы доступ в некоторым паролям имели не только сотрудники компании. Так же планируется улучшить эффективность серверной и клиентской части приложения, для этого будут использоваться специальные инструменты замера метрик скорости работы приложения.

Список литературы

[1] TeX в ЯрГУ [Электронный ресурс]. URL: http://www.tex.uniyar.ac.ru (дата доступа: 20.05.2017).

[2] Oetiker T., Partl H., Hyna I., Schlegl E. The Not So Short Introduction to LaTeX 2_ε [Электронный ресурс]. URL: https://tobi.oetiker.ch/lshort/lshort.pdf (дата доступа: 01.06.2017).

[3] Котельников И. А., Чеботаев П. З. LaTeX 2_ε по-русски. 3-е изд., перераб. и доп. Новосибирск : Сибирский хронограф, 2004. 496 с.

[4] Tantau T. PGF — Create PostScript and PDF graphics in TeX [Электронный ресурс]. URL: https://www.ctan.org/pkg/pgf (дата доступа: 17.05.2017).

[5] Кирютенко Ю. А. TikZ & PGF. Создание графики в LaTeX 2_ε-документах. Ростов-на-Дону, 2014. 277 с. URL: https://open-edu.sfedu.ru/files/pgf-ru-all-method.pdf

[6] Cook S. A. The complexity of theorem-proving procedures // Proceedings of the third annual ACM symposium on Theory of computing. ACM, 1971. P. 151–158.

[7] Пупырев С. Н., Тихонов А. В. Визуализация динамических графов для анализа сложных сетей // Модел. и анализ информ. систем. 2010. Т. 17, № 1. С. 117–135.

[8] Кузьмин И. Г. Некоторые проблемы государственных финансов в современной России // Российские предприятия в системе рыночных отношений : материалы научно-практич. конф. Ярославль, 17–18 окт. 2000 г. / отв. ред. Л.Б. Парфенова. Ярославль, 2000. С. 86–90.

[9] ГОСТ Р 517721-2001. Аппаратура радиоэлектронная бытовая. Входные и выходные параметры и типы соединений. Технические требования. Введ. 2002-01-01. М. : Изд-во стандартов, 2001. IV, 27 с.

Приложение А

Исходный код программы на TypeScript

```
async findOneWithBasicInfo(
    entry_id: number,
    currentUser,
    req,
  ): Promise<Record<string, any>> {
    const entry = await this.entrysRepository.findByPk(entry_id, {
      raw: true,
      attributes: [
        [Sequelize.literal('CONCAT(\'e\', "Entry".entry_id)'), 'uniq_id
        ['entry_id', 'id'],
        'name',
        'comment',
        'parent_id',
        'type',
        [Sequelize.literal('array_agg(tagss.name)'), 'tags'],
      ],
      group: ['Entry.entry_id', 'Entry.name'],
      include: [
        {
          model: Tag,
          as: 'tagss',
          attributes: [],
          through: {
            attributes: [],
          },
        },
      ],
    });

    if (!entry) {
      throw new NotFoundException('Password are not exist');
    }

    const parentFolder = await this.foldersRepository.findByPk(entry.pa
    const { access } = await this.accessService.isAccess(
```

```
      entry.parent_id,
      parentFolder.owner_id,
      currentUser,
      req,
    );

    if (!access) {
      throw new ForbiddenException('Do not have access to entry');
    }

    try {
      if (entry.tags[0] === null) {
        entry.tags = [];
      }
      return {
        ...entry,
        is_subscribe: await this.subscribesService.isInSubscribes(
          currentUser?.id,
          'e',
          entry_id,
        ),
        is_favorite: await this.favouritesService.isInFavourites(
          currentUser?.id,
          'e',
          entry_id,
        ),
      };
    } catch (e) {
      throw new NotFoundException('Password are not exist');
    }
  }
```