|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ |
|  | Руководитель Учебной практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженкова М. А.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

приложения «Steam Achievment view»

|  |
| --- |
| СОГЛАСОВАНО |
| Колледж ВятГУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |
|  |
|  |

2023

1. **Описание проекта:**

Название проекта: Steam Achievements Analyzer

Цель проекта: Создание web-приложения, которое позволит пользователям анализировать и просматривать свои достижения в играх Steam.

1. **Основные функциональные возможности:**

Аутентификация пользователя с использованием Steam API.

Получение списка игр пользователя с использованием Steam API.

Отображение достижений пользователя для выбранной игры.

Поиск игр по названию или жанру.

Отображение детальной информации о достижении.

Возможность сортировки достижений по дате получения, по статусу (завершенные/незавершенные) или по названию игры.

Возможность просмотра профиля пользователя и его статистики в Steam.

1. **Уточнение структуры данных:**

Пользователь:

* Идентификатор пользователя (Steam ID)
* Имя пользователя
* Аватар пользователя
* игры пользователя (список игр)
* средний процент достижений

Игра:

* Идентификатор игры (App ID)
* Название игры
* Статистика пользователя игры
* Достижения пользователя в игре
* Обложка игры

Достижение:

* Идентификатор достижения (Achievement ID)
* Название достижения
* Описание достижения
* Статус достижения (завершено/незавершено)
* Дата получения достижения
* Процент получения среди игроков
* Иконка завершенного достижения
* Иконка не завершенного достижения

1. **Формы представления данных:**

Данные пользователя отображаются на главной странице в виде колонки данных:

* Имя
* Аватар
* Количество игр
* Количество достижений
* Средний процент достижений

Данные об игре представлены в виде карточке. На карточках находятся название игры, обложка игры, статистика пользователя в игре (Время игры, все достижения в игре, полученные достижения, не полученные достижения, процент достижений), и несколько последний достижений (полученных если возможно)

Последние 36 полученных достижений отображаются в квадратной контейнере в виде блока 6 на 6, при наведении на достижение отображается название игры, имя достижения, описание достижения, процент игроков, получивших достижения и время получения.

Достижения в игре или все достижения пользователя отображаются в виде таблицы, если просматриваются все достижения пользователя, то надписи открыто нет.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Открыто | Название | Описание | Процент среди игроков | Дата открытия |
| Иконка | Имя | Сделай что-то | 20% | 2023.9.26 |

Таблица 1 – список достижений

Для сортировки и фильтрации игр присутствует контейнер, в котором есть возможность ввести название игры (при вводе скрываются неподходящие игры в реальном времени). Для сортировки необходимо нажать на кнопку сортировать, после чего выпадет список с выбором сортировки. В контейнере есть стрелки для выбора направления сортировки. Для фильтрации есть кнопки игровое время и завершенность, которые предоставляют выпадающее списки с выбором соответствующей фильтрации

1. **Разработка алгоритма решения задачи:**

Аутентификация пользователя:

* Получение токена доступа и SteamId от пользователя .
* Попытка получение данных о пользователе.

Получение и обработка игр пользователя:

* Использование Steam API для получения списка игр пользователя.
* Использование Steam API для получения всех данных об достижениях в играх
* Обработка достижений в играх, для получения процентов достижений в них, количества достижений, количества полученных достижений.
* Сбор этих данных в один цельный JSON
* Сохранение данных в localstorage

1. **Определение языка, структуры программы и требований к техническим средствам:**

Язык программирования:

Был выбран TypeScript из-за типизации, а также из-за использования Electron Forge.

Фреймворк:

Был выбран React.js из-за его реактивности и возможности обновления компонентов, без необходимости в обновлении всей страницы.

Сборка и упаковка приложения: Electron Forge

Хранение данных:

Используется localstorage из-за небольшого количества хранимых данных и доступности к ним из любого места

Взаимодействие с API Steam:

Используется Steam Web API из-за отсутствия альтернатив

Использование сторонних библиотек:

Используется i18next для локализации, react-loading-overlay-ts для анимации загрузки, react-spinners/BounceLoader для определенного вида загрузки, react-toastify для оповещения пользователя, Express для создания локального сервера (чтобы обходить ограничения CORS политики)

1. **Требования к техническим средствам:**

Клиентская часть:

* Доступ к интернету для загрузки данных из API Steam.