М.Е. Перепилица[1], Д.Ю. Казанцев[1], М.В. Аврискин[12]

[1] Тюменский государственный университет, г. Тюмень

[2] Научно-технический университет «Сириус», г. Сочи

УДК 004.91

ПРИЛОЖЕНИЕ С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ КОНТРОЛЯ ВЕРСИЙ GIT ДЛЯ ОРГАНИЗАЦИИ ПРОЦЕССА ОБУЧЕНИЯ

Аннотация. В статье представлен вариант организации учебной работы преподавателей со студентами через приложение для настольных операционных систем с использованием технологии Git.

Ключевые слова: Git, Electron, JavaScript, клиент-сервер, API, кроссплатформенное приложение.

Сейчас существует очень много способов взаимодействия между преподавателями и студентами. Но такие, которые позволили бы сдавать работы удобным, быстрым и эффективным способом, отсутствуют. Именно эту задачу мы решали с помощью приложения, которое работает с системой Git.

Git – одна из наиболее популярных систем контроля версий [4]. Система позволяет сохранять информацию, в нашем случае программный код на разных этапах разработки в локальном репозитории и синхронизировать данные с удаленным репозиторием. Таким образом система подходит для использования в целях сдачи учебных заданий, позволяя студенту отправить свою работу на удаленный сервер, а преподавателю – просмотреть историю изменений в случае длительной поэтапной работы над проектом или лабораторной.

Для того, чтобы получить кроссплатформенное приложение, то есть чтобы была возможность запускать его на macOS, Linux и Windows, было принято решение использовать такой инструмент, как Electron [1] – framework для языка программирования JavaScript. Он позволяет использовать все возможности операционной системы, например, уведомления, а также разрешает написать единый код для всех ОС. Это становится возможным благодаря использованию

web-технологий. То есть написанный код, является веб сайтом, который встраивается и работает, как отдельное настольное приложение.

Наше приложение позволяет создавать преподавателям группы, добавлять студентов, проверять работы. Все взаимодействия представлены на рис. 1. Приложение с помощью REST API запрашивает данные о студентах, преподавателях, группах у нашего сервера, написанного на JavaScript с применением Express Framework [5]. Сервер получает запрашиваемые сведения из базы данных PostgreSQL. А все взаимодействия с удаленными репозиториями происходят с помощью GitHub API.

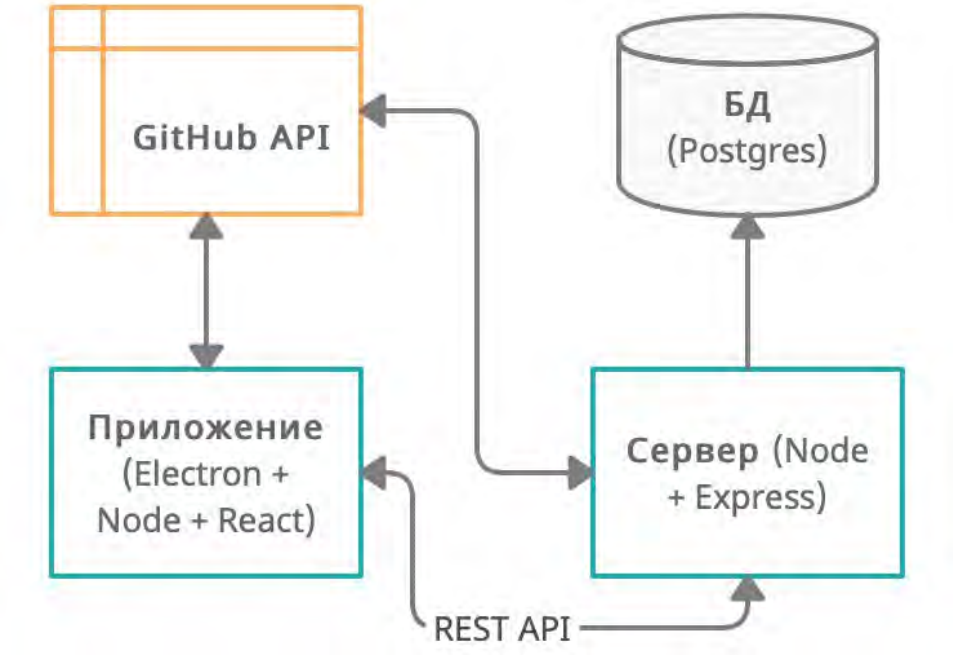


Рис. 1. Схема архитектуры работы приложения

Ключевые же возможности приложения состоят в том, что выдача, сдача и проверка лабораторных работ реализована с помощью Git. Для этого используется сервис GitHub, предоставляющий возможность создания и хостинга Git-репозиториев, а также web-интерфейс и REST API для работы с ними. С помощью запросов к API GitHub мы и получаем возможность создать удобную среду для взаимодействия преподавателей со студентами.

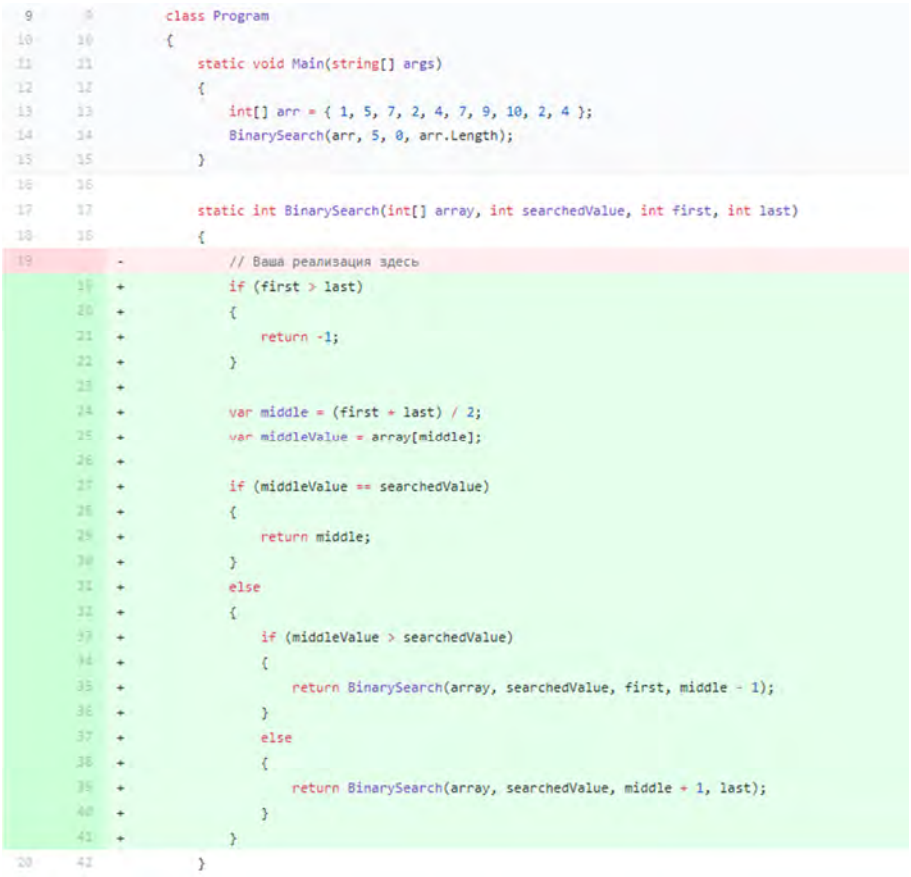
Основная технология, которая осуществляет работу с Git – NodeGit [2], библиотека NodeJS, которая предоставляет удобный интерфейс для взаимодействия с Git.

```
const repo = await nodegit.Repository.init(path.resolve(__dirname, repoDir), 0);
await fs.promises.writeFile(path.join(repo.workdir(), fileName), fileContent);
const index = await repo.refreshIndex();
await index.addByPath(fileName);
await index.write();
```

Рис. 2. Пример кода создания репозитория через NodeGit

Алгоритм учебного процесса со стороны преподавателя выстроен так:

1. Преподаватель регистрируется в системе, администратор присваивает ему роль преподавателя.
2. Преподаватель добавляет лабораторные работы (для начала без привязки к группам), при этом автоматически создаются репозитории на GitHub с загруженными файлами.
3. Преподаватель присваивает добавленные лабораторные нужным учебным группам.
4. Если есть работы, которые можно проверить, то преподаватель нажимает «Начать проверку», и ему на компьютер скачивается репозиторий студента в настроенную папку по умолчанию.
5. Код и изменения преподаватель также может просматривать через веб-интерфейс GitHub.
6. Если задание принято, преподаватель нажимает «Принять».



```
class Program
{
    static void Main(string[] args)
    {
        int[] arr = { 1, 5, 7, 2, 4, 7, 9, 10, 2, 4 };
        BinarySearch(arr, 5, 0, arr.Length);
    }

    static int BinarySearch(int[] array, int searchedValue, int first, int last)
    {
-       // Ваша реализация здесь
+       if (first > last)
+       {
+           return -1;
+       }
+
+       var middle = (first + last) / 2;
+       var middleValue = array[middle];
+
+       if (middleValue == searchedValue)
+       {
+           return middle;
+       }
+       else
+       {
+           if (middleValue > searchedValue)
+           {
+               return BinarySearch(array, searchedValue, first, middle - 1);
+           }
+           else
+           {
+               return BinarySearch(array, searchedValue, middle + 1, last);
+           }
+       }
    }
```

Рис. 3. Пример работы с веб-интерфейсом GitHub

Алгоритм учебного процесса со стороны студента:

1. Студент регистрируется в системе.
2. Преподаватель добавляет его в определенную группу.
3. Студент может посмотреть список лабораторных для выбранной группы и начать выполнение выбранной работы.
4. При начале выполнения студенту скачивается на компьютер репозиторий лабораторной с текстом задания и файлами-шаблонами в ту папку, которую студент выбрал в настройках.
5. Студент выполняет работу, сохраняет промежуточные этапы своей работы в локальном репозитории и синхронизирует с удаленным.
6. После того, как студент закончил работу, он нажимает «Сдать работу», и у преподавателя в списке лабораторных отмечается, что студент отправил работу на проверку.
7. Студент ждет результата проверки и при успехе задание отмечается зеленым цветом в общем списке.

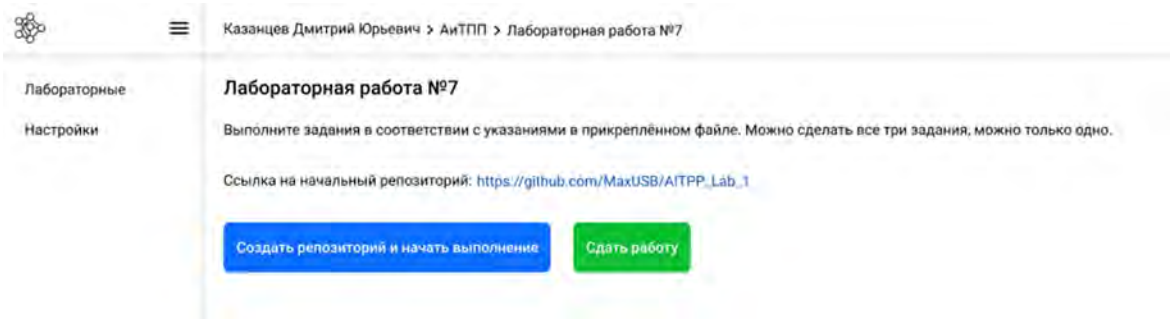


Рис. 4. Пример интерфейса приложения со стороны студента

Таким образом мы получаем инструмент, который позволяет эффективно работать с заданиями экономить время, как преподавателей, так и студентов, а также позволяет получить навыки работы с Git, которые будут полезными для создания большинства профессиональных программных продуктов.

Благодарности

Статья подготовлена в рамках разработки образовательного кейса для НТУ Сириус при финансовой поддержке РФФИ в рамках научного проекта № 19-37-51028.

СПИСОК ЛИТЕРАТУРЫ

1. Документация Electron [Электронный ресурс] // Electron: https://www.electronjs.org/docs (дата обращения: 15.04.21).

2. Документация NodeGit [Электронный ресурс] // NodeGit: https://www.nodegit.org/api (дата обращения: 23.04.21).

3. Документация GitHub API [Электронный ресурс] // GitHub API: https://docs.github.com/en/rest (дата обращения: 16.04.21).

4. Документация Git [Электронный ресурс] // Git: https://git-scm.com/book/ru/v2 (дата обращения: 15.04.21).

5. Документация Express [Электронный ресурс] // Express: https://expressjs.com (дата обращения: 20.04.21).