Санкт-Петербургский государственный университет

Кафедра Системного программирования

Группа 21.Б10-мм

Разработка инструмента для интеграции Timetable в приложения-календари

Минязев Роман Русланович

Отчёт по учебной практике
в форме «Решение»

Научный руководитель:
старший преподаватель кафедры системного программирования,
Кириленко Яков Александрович

Санкт-Петербург
2023

Оглавление

Введение

Для получения учебного расписания Санкт-Петербургского государственного университета(СПБГУ), преподаватели и студенты пользуются сервисом электронного расписания Timetable [6]. Однако, расписание содержит множество событий, которые часто меняются, то есть на составление, проверку и обновление расписания вручную тратится большое количество времени.

Преподаватели СПБГУ, а в частности преподаватели матмеха, имеют большое количество событий, не связанных напрямую с расписанием Timetable. Для снижения количества времени на составление общего расписания и проверки его актуальности, требуется возможность видеть все свои события в одном приложении.

Решением этой проблемы может стать интеграция расписания в приложения-календари, которыми пользуются преподаватели и студенты СПБГУ. Пользователь получит возможность импорта/подписки на календарь со своим расписанием и сможет отслеживать его в своем любимом приложении(Google Calendar, Yandex Calendar и т.д.).

Для упрощения создания подобных инструментов для работы с расписанием, можно построить Rest API, который расширял бы функциональность Timetable API и давал возможность не тратить время на обработку данных с Timetable. Также, решением может стать создание базы данных и предоставление разработчикам исходных файлов для ее создания и заполнения или готовых .csv файлов, которые можно обрабатывать с помощью различных библиотек.

1. Постановка задачи

Целью работы является реализовать инструмент для интеграции Timetable в приложения-календари.

Для её выполнения были поставлены следующие задачи:

1. Провести обзор существующих решений и разбор функциональности Timetable API.
2. Спроектировать базу данных для представления сущностей из Timetable.
3. Написать скрипты для автоматического заполнения базы данных.
4. На основе базы данных и других решений реализовать интеграцию Timetable в разные приложения-календари.
5. Провести оценку удобства использования инструмента.

2. Обзор

В обзоре рассматривались как веб, так и Android приложения для работы с Timetable. Поиск производился при помощи поисковых систем Yandex и Google.

2.1. Telegram Bot

Для получения расписания из Timetable релизован telegram бот [10]. Несмотря на его удобство в плане получения расписания в Telegram, он не решает некоторые поставленные цели. Одна из ключевых целей работы — избавить пользователя от необходимости использования Timetable отдельно от своего общего расписания, т.е. существует потребность видеть все свои события в одном приложении. Также, просмотр большого количества событий(для разных групп и преподавателей) будет неудобным из-за формата получения расписания, в то время как в календарях можно выделить под каждую группу или преподавателя персональный календарь, пометить их разными цветами и т.д. При этом существует проблема массового получения расписания(нет возможности выгрузить расписание на семестр, год или пять лет если это необходимо).

2.2. Расписание СПБГУ на Android

Приложение [13] представляет из себя урезанный Timetable, но в виде Android приложения, то есть обладает теми же недостатками, но при этом накладывает ограничение на количество одновременно просматриваемых преподавателей или групп. Кроме того, при каждом входе в приложение нужно заново выбирать факультет, направление и т.д., что является плохим вариантом с точки зрения удобства использования.

2.3. Timetable API

Timetable API не задокументирован(возможно, документация где-то существует, но ссылка https://timetable.spbu.ru/api/docs на документацию не рабочая). Причем для выгрузки событий не было указано формата ссылок.

Timetable API на данный момент предоставляет следующую функциональность:

GET запрос	Описание
https://timetable.spbu.ru/api/v1/addresses	Получение адресов
https://timetable.spbu.ru/api/v1/addresses/oid/classrooms	Получение аудиторий по oid адреса
https://timetable.spbu.ru/api/v1/educators/search/query	Поиск преподавателя по фамилии
https://timetable.spbu.ru/api/v1/study/divisions	Получение всех направлений обучения
https://timetable.spbu.ru/api/v1/study/divisions/alias/programs/levels	Получение учебных программ по alias направления
https://timetable.spbu.ru/api/v1/programs/id/groups	Получение групп по id учебной программы
https://timetable.spbu.ru/api/v1/educators/id/events/YYYY-mm-dd/YYYY-mm-dd	Получение расписания преподавателя по timetable id, левой и правой временных границах для выгрузки событий
https://timetable.spbu.ru/api/v1/groups/id/events/YYYY-mm-dd/YYYY-mm-dd	Получение расписания группы по timetable id, левой и правой временных границах для выгрузки событий

2.4. Остальные решения

Существует еще несколько приложений для работы с расписанием, но все они также не решают целей данной курсовой работы:

1. Веб-приложение для получения и распечатки раписания СПБГУ [12].

2. Анализ расписания СПБГУ [11].

3. Еще один телеграм бот [4].

3. Реализация

3.1. Используемые инструменты

Для реализации инструмента были использованы языки программирования Python и JavaScript, веб-фреймворк Quart, облачная платформа Google Apps Scripts и СУБД PostgreSQL.

3.2. Выбор СУБД

Среди рассматриваемых СУБД были SQLite и PostgreSQL. Для возможности расширения проекта с использованием мультипроцессинга было решено использовать PostgreSQL.

В качестве адаптеров для PostgreSQL в Python рассматривались psycorg и asyncpg. Так как последний является более производительным на больших данных [8], было решено остановиться на нем. Для миграций базы данных используется модуль Alembic.

3.3. Построение базы данных

Для построения базы данных необходимо выделить основные сущности с Timetable:

Сущность	Описание
Division	Факультет или иное направление подготовки
Program	Программа обучения
Group	Конкретная группа для каждой программы
Educator	Преподаватель
Event	Пара, пересдача, комиссия

Так как каждому событию может соответствовать несколько преподавателей и несколько групп, существуют таблицы для связи многие ко многим для отношений Educator и Event и для отношений Group и Event. Полученная ER диаграмма изображена на рис. 1.

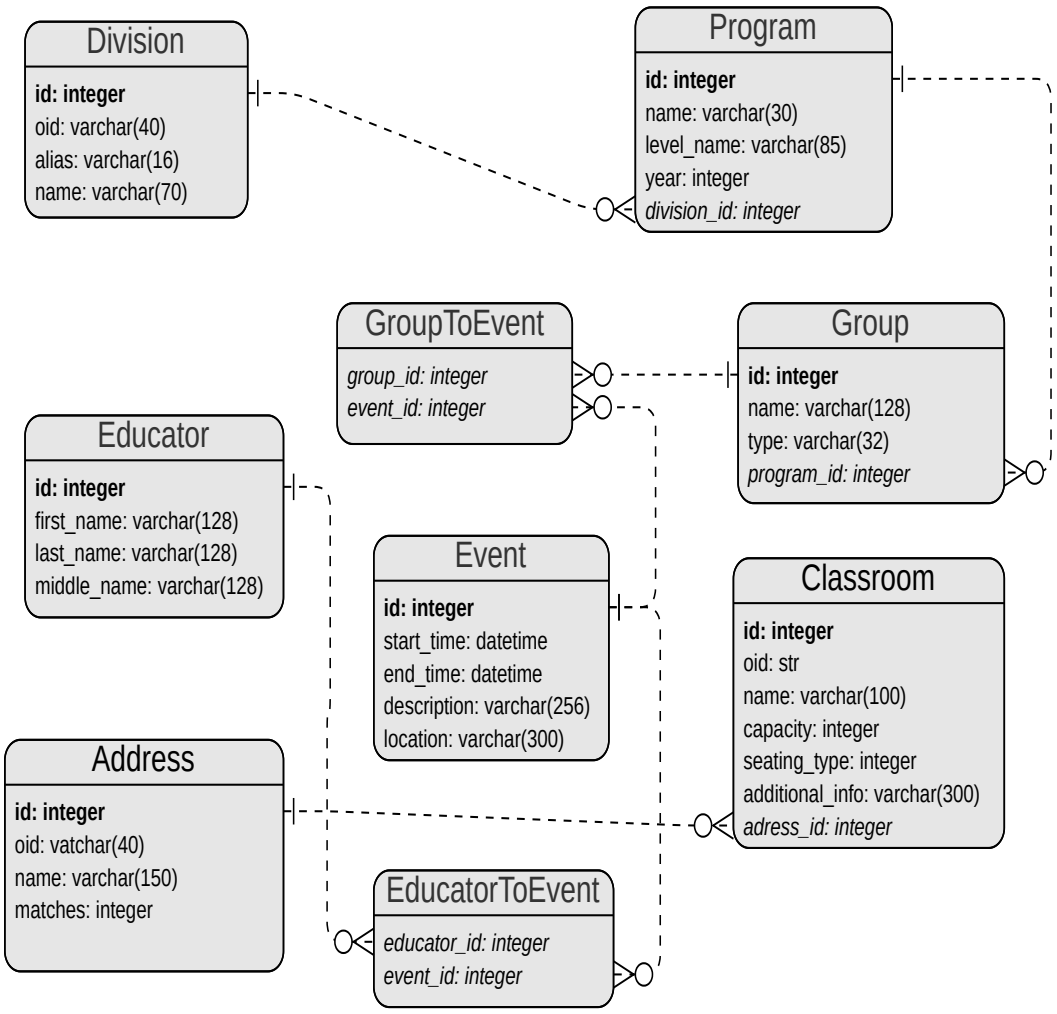

Рис. 1: ER Диаграмма

3.4. Получение файлов календаря

Для возможности импорта и подписки на календарь, необходимо на стороне сервера генерировать файлы календаря(.ics) и предоставить возможность их получения GET запросом. Ссылка для GET запроса : http://194.87.237.205/calendar.

Если не указаны левая и правая граница даты, то расписание будет автоматически генерироваться на месяц вперед.

Для удобства составления ссылок, используемых для получения календарей, написан генератор ссылок. [1]

Параметр	Описание
type	educators, groups
id	timetable id группы или преподавателя
left_date	Левая граница даты для выгрузки расписания
right_date	Правая граница даты для выгрузки расписания

3.5. Веб-фреймворк

Так как разработка ведётся на Python, то в качестве веб-фреймворка был выбран Flask из-за своей простоты(проект одностраничный), но в последствии проект был переведен на Quart, поскольку он является асинхронным фреймворком и поддерживает ASGI.

3.6. Выгрузка данных

Существует несколько подходов к выгрузке данных с Timetable.

3.6.1. Scraping

Одним из подходов для получения данных с Timetable является их извлечение со страниц. Такой метод не накладывает сильных ограничений на количество запросов, что дает возможность довольно эффективно получать данные с Timetable. Однако, он подвержен частым ошибкам при изменении структуры, так как в некоторых случаях данные собираются совершенно произвольно. Как пример, страница с поиском преподавателей СПБГУ [5]. Единственный фактор, по которому можно отличить должность преподавателя от его ФИО или кафедры — тип столбца. Минимальная замена структуры будет влечь за собой множество ошибок. Данные не хранятся в структурированном виде, поэтому повышается риск ошибок не только при их получении(например, на Timetable может быть не всегда указана левая или правая временная граница события, могут быть не указаны группы или места проведения), но и при фильтрации и работе с ними(например, если сменится формат даты).

Для реализации [2] этого подхода использовался модуль Beautiful

Soup. С помощью этого модуля по html-разметке строится дерево разбора, по которому можно производить поиск, вырезать ненужные html элементы и т.д. В написанном скрипте выгружается информация о преподавателях, группах, факультетах и направлениях, после чего на основании этого выгружается информация о событиях.

Прототип доступен по ссылке:
https://github.com/MinyazevR/timetable

3.6.2. Timetable API

Другой способ выгрузки данных — использование timetable API [7]. Использование API помогает избежать ошибок, связанных с получением и обработкой данных. Однако, Timetable API имеет определенные ограничения. А именно, 1500 запросов в день и 30 запросов в секунду. Очевидно, что для заполнения базы данных, а тем более ее заполнения несколько раз в день, не хватит такого количества запросов. Для обхода ограничения в 1500 запросов в день используются прокси сервера. Для того, чтобы вместиться ограничение в 30 запросов в секунду, каждые 30 запросов обрабатываются с определенными таймерами для сна. Ответ от Timetable API приходит в формате JSON, который парсится с помощью встроенных модулей языка Python. Асинхронная обработка производилась с помощью модулей asyncio и aiohttp.

Реализация доступна по ссылке: https://github.com/MinyazevR/timetable-database

Оба этих подхода были реализованы и из-за надежности и корректности получения данных, был выбран подход с API. Логирование происходит при помощи модуля logging. Для удобного автоматического мониторинга приложения и диагностики ошибок используется Sentry [9].

3.7. Google Calendar

Google Calendar поддерживает импорт календарей и возможность подписки на календари по ссылке, однако он редко выгружает события из сторонних календарей, что критично для получения актуального

раписания, поэтому было решено получать права на изменение календарей и работать с ними при помощи Google Calendar API.

3.7.1. Решения к разрешению доступа Google Calendar

Для получения доступа к календарям пользователей, необходимо на стороне сервера настроить процесс авторизации, что влечет за собой определенные риски. Необходимо продумать процесс хранения access и refresh token. Есть множество подходов для хранения access token: local storage, cookie и т.д.. Однако, все они, так или иначе, имеют свои недостатки. В связи с этим, было решено использовать Google Apps Script. Google Apps Script — облачная платформа для взаимодействия с сервисами Google. В случае использования Google Apps Script, процесс авторизации настроен автоматически и получение токена доступа сводится к использованию одной библиотечной функции. Однако, это не защищает пользователя от потенциальной возможности кражи или порчи его календарей автором скрипта. В этом случае пользователю предлагается создать еще один Google аккаунт, после чего предоставить доступ к изменению календарей с этого аккаунта и подписаться на этот календарь с основого аккаунта. Это будет работать, так как Google Calendar эффективно выгружает события из других Google календарей. Также, при использовании GAS для каждого пользователя существует возможность изменения параметров(таких как частота выгрузки, выгружаемые id и прочее) без необходимости написания вебсервиса.

3.7.2. Реализация для Google Calendar

Реализация для Google Calendar [3] является автономной, так как не зависит от сервера и базы данных, а обращается напрямую к Timetable API. Пользователь должен в скрипте указать тип расписания: преподаватель или группа, Timetable id, а также предоставить id своего календаря. При помощи Google Calendar API из календаря удаляются события для нужного промежутка времени, после чего на основе данных

с Timetable API генерируется payload для POST запроса и в календарь добавляются актуальные события. Написана отдельная функция, которая создает триггер на запуск скрипта каждые 6 часов. Однако, пользователь может настроить частоту выполнения скрипта самостоятельно с помощью панели в Google Apps Script. Дневной лимит на запуск триггерных функций — 90 минут, чего более чем достаточно на выгрузку расписания с timetable.

3.7.3. Улучшение usability для Google Calendar

При выходе новой версии скрипта у пользователя в календаре создается событие с просьбой об обновлении скрипта и соответсвующим описанием изменений. Пользователю необходимо заново скопировать обновленный файл. Такое решение может быть достаточно неудобным, в частности, при обновлении скрипта с мобильных устройств. Для удобства обновления скрипта есть несколько решений:

1. Использовать eval на .gs скрипт, который лежит на github и периодически обновляется. Такое решение будет менее безопасным для пользователя чем текущее, так как скрипт обновляется автоматически, в то время как при копировании есть возможность изучить скрипт перед тем как его запускать или давать разрешения. В качестве решения этой проблемы можно просить пользователей делать fork репозитория, после чего делать eval. Периодически синхронизировать fork может показаться более удобным вариантом, но далеко не все преподаватели СПБГУ есть на github, поэтому текущее решение является более общим.

2. Делать развертывание скрипта в библиотеку. Для перехода на новую версию будет достаточно сменить версию библиотеки с помощью веб-интерфейса в Google Apps Script.

3.8. Yandex Calendar

Yandex Calendar также поддерживает импорт календарей и возможность подписки на календари по ссылке. Частота выгрузки для Yandex Calendar 1-2 раза в день.

3.8.1. Реализация для Yandex Calendar

Таким образом, пользователю необходимо подписаться на календарь по ссылке, формат которой был указан выше. Получая GET запрос с параметрами, сервер, обращаясь к базе данных, получает нужную выборку, генерирует и возращает .ics файл.

3.8.2. Решения для увеличения частоты выгрузки Yandex Calendar

Возможные решения:

1. Настроить на сервере процесс авторизации для пользователей Яндекс календаря, учитывая специфику Яндекс OAuth и все возможные риски, связанные с получением доступа к календарям пользователей. А также предоставить пользователям возможность менять выгружаемые календари, частоту выгрузки и прочее.

2. Найти решение, аналогичное Google Apps Script и Google Calendar. Возможно, Yandex Cloud Functions или какие-то сервисы Yandex предоставляют упрощенную и безопасную работу с токенами доступа.

3.9. Yahoo Calendar и Outlook

Yahoo Calendar поддерживает возможность подписки на календари по ссылке и постоянно обновляет их автоматически, но также присутствует возжность обновления вручную. Outlook обновляет события каждые 6 часов. То есть для пользователей Yahoo Calendar и Outlook самым удобным и рабочим решением будет подписка на календарь по ссылке.

3.10. Android и IOS

На данный момент нет никакой реализации для Android и IOS, однако имеется возможность получить расписание на мобильные устройства при помощи Google, Yandex Calendar или Outlook. Инструкция:

- Для пользователей Google Calendar: добавьте в настройках календаря свой Google account.
- Для пользователей Yandex Calendar: перейдите по ссылке и проследуйте инструкции https://yandex.ru/support/calendar/common/sync/sync-mobile.html.

3.11. Расширение функциональности для календарей

Примеры:

1. Текущая реализация выгружает расписание Timetable таким, какое оно есть. Однако, оно содержит много лишней информации. В частности, для студентов:
    - Пары по английскому языку, которые для каждого преподавателя создаются как отдельные события в расписании группы.
    - Элективы, которые не являются элективами по выбору для студента(и на которые он не хочет ходить).
    - Русский язык как иностранный.
2. Добавить для каждого пользователя возможность выгрузки расписаний нескольких групп/преподавателей в одном скрипте. На данный момент существует такая возможность, но придется сделать несколько копий скрипта.

3.12. Практическая польза проекта на конкретном примере

Используя построенную базу данных, можно за короткое время реализовать поиск свободных аудиторий СПБГУ. Получив от пользователя адрес здания и время, на которое нужна аудитория, можно простым SQL запросом получить нужную выборку(с помощью отношений Event, Classroom, Address), а также улучшить удобство использования с помощью опциональных параметров(например, желаемый этаж) или на основе дополнительного описания(наличие проектора) с помощью фильтрации.

4. Апробация

Так как данный инструмент рассчитан на большое количество пользователей, то важным фактором является удобство его использования.

Удобство использования будет проверяться достаточно стандартным образом, а именно с помощью System Usability Scale. Эта шкала была выбрана, так как на нее ссылается более 1300 статей и публикаций, а также из-за своей простоты и надежности.

Критерии для отбора испытуемых:

1. Быть пользователем данного инструмента.
2. Быть студентом / преподавателем СПБГУ.

Заключение

Текущие результаты:

- Реализовано несколько подходов к выгрузке данных с Timetable.
- Написаны скрипты для заполнения базы данных.
- Реализовано одностраничное веб-приложение для получение файлов календаря GET запросом и генератор ссылок для календарей.
- Реализована интеграция Timetable в Google Calendar при помощи платформы Google Apps Script(также есть реализация GAS для Yandex, но она является избыточной).
- Реализована интеграция Timetable в Yandex Calendar, Yahoo!, Outlook при помощи GET запросов к серверу и встроенной функциональности этих календарей.

Планируется сделать:

- Улучшение usability для Google Calendar.
- Увеличение частоты выгрузки для Yandex Calendar.
- Расширение функциональных возможностей.
- Рассмотреть возможность расширения проекта с базой данных в мультипроцессинг для улучшения временных показателей.

Выгрузка при помощи Timetable API:
https://github.com/MinyazevR/timetable-database.
Реализация при помощи GAS:
https://github.com/MinyazevR/spbu-timetable-gas-integration
Ссылка для получения календарей GET запросом:
http://194.87.237.205/calendar
Выгрузка при помощи скрейпинга(рабочий прототип):
https://github.com/MinyazevR/timetable.

Список литературы

[1] MinazevR. Link generator. — URL: http://194.87.237.205/link (online; accessed: 2023-09-16).

[2] MinyazevR. Scraper for timetable. — URL: https://github.com/MinyazevR/timetable (online; accessed: 2023-08-14).

[3] MinyazevR. Timetable GAS integration. — URL: https://github.com/MinyazevR/spbu-timetable-gas-integration (online; accessed: 2023-08-14).

[4] Platonov Ivan. SPBGU Timetable. — URL: https://github.2u2.cc/EeOneDown/spbu4u (online; accessed: 2023-08-14).

[5] SPBGU. Search for SPBGU teachers. — URL: https://timetable.spbu.ru/EducatorEvents/Index (online; accessed: 2023-08-14).

[6] SPBGU. Timetable. — URL: https://timetable.spbu.ru/ (online; accessed: 2023-08-14).

[7] SPBGU. Timetable API. — URL: https://timetable.spbu.ru/api/v1 (online; accessed: 2023-08-14).

[8] Selivanov Yury. asyncpg. — URL: https://github.com/MagicStack/asyncpg (online; accessed: 2023-08-14).

[9] Sentry. Sentry. — URL: https://sentry.io/welcome/ (online; accessed: 2023-08-14).

[10] Vadim Sazanov. Timetable_SPBU_bot. — URL: https://github.com/vad-ii-k/Timetable_SPBU_bot (online; accessed: 2023-08-14).

[11] https://github.com/orgs/spbu-schedule/people. SPBGU Timetable. — URL: https://github.com/spbu-schedule (online; accessed: 2023-08-14).

[12] https://github.com/spbu-timetable/timetable/graphs/contributors. Timetable. — URL: https://github.com/spbu-timetable/timetable (online; accessed: 2023-08-14).

[13] stardust.skg. SBGU Timetable. — URL: https://play.google.com/store/apps/details?id=ru.vbushev.spbsu.timetable (online; accessed: 2023-08-14).