**Степанов В. В., Журавлевич М. Е., Петухов Сережа**

**Документация**

**К микро сервисному проекту, разработанному на основе языков:**

**Java, JS, Ruby, Go и т. д.**

**Абакан 2023 г.**

Содержание

[Введение 3](#_Toc134119258)

[1. ЧАСТНОЕ ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА ПОДСИСТЕМУ “РАСПИСАНИЕ” 4](#_Toc134119259)

[1. ВВЕДЕНИЕ 4](#_Toc134119260)

[2. НАЗНАЧЕНИЕ РАЗРАБОТКИ 4](#_Toc134119261)

[3.1 REST API GATEWAY 4](#_Toc134119262)

[3.2. REST API Audience 4](#_Toc134119263)

[3.3 REST API Teachers 5](#_Toc134119264)

[3.4. WEB Client “Расписание” 5](#_Toc134119265)

[3.5 Mobile Client “Расписание” 6](#_Toc134119266)

[3.6 Desktop Client “Диспетчерская” 6](#_Toc134119267)

[4. Диаграммы 8](#_Toc134119268)

Введение

Расписание — вид календаря (то есть, упорядоченность по времени), для которого указана информация о предстоящих (планируемых или потом произошедших) событиях.

Ни для кого, ни секрет, что в учебных заведениях используют расписание. Расписания в школах, в колледжах, в университетах, меняется формулировка, но основа у них одна, расписание каких-либо занятий, которые предоставляют преподаватели, аудитория и время, которое формулируется из нескольких пунктов. Но по стандартному расписанию невозможно быстро оценить в какое время будет преподаватель в учебном заведении, или когда аудитория будет свободна. Так же на такое расписание оказывается большой человеческий фактор при заполнении и человек спокойно может ошибиться при заполнении.

По небольшому техническому заданию от заказчика требовалось соблюдать следующие пункты:

* приложение должно соблюдать микро сервисную архитектуру, для переиспользования одних и тех же сервисов внутри организации, как пример, говорились аудитории и преподаватели;
* клиенты должны иметь приятный глазу дизайн;
* для общения между сервисами использовать REST API

Для такого приложения присутствует несколько ролей, которые будут делиться на клиенты, тем самым пользователь даже, подсмотрев где-нибудь пароль должности выше не сможет в своем клиенте подключиться к функционалу главного человека.

1. ЧАСТНОЕ ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА ПОДСИСТЕМУ “РАСПИСАНИЕ”
2. ВВЕДЕНИЕ

Наименование разработки: “Программное обеспечение подсистемы управления расписанием” (далее подсистема)

Подсистема является дополнением общей системы учебного заведения.

1. НАЗНАЧЕНИЕ РАЗРАБОТКИ

Подсистема предназначена для полной автоматизации работы студентов, преподавателей и частичной диспетчерской, а главное сделать работу комфортнее.

1. ТРЕБОВАНИЯ К ПРОГРАММНЫМ ПРОДУКТАМ:

**3.1 REST API GATEWAY**

- способность запускаться на UNIX-системах и Windows 7, 8, 10;

- поддержка асинхронных HTTP – запросов;

- наличие Gateway - токена для защиты от посторонних приложений и людей;

- реализация HATEOAS;

- репликация данных из других сервисов;

- реализация под возможность MAJOR-обновлений;

- наличие базы данных;

- сервис должен автоматически заполнять атрибуты

- при обращении к доменному имени должен возвращаться список всех возможных ссылок.

**3.2. REST API Audience**

- способность запускаться на UNIX-системах и Windows 7, 8, 10;

- поддержка асинхронных HTTP – запросов;

- реализация HATEOAS;

- реализация под возможность MAJOR-обновлений;

- наличие базы данных;

- сервис должен подписывать на события, к примеру, обновления данных в базе данных;

- при обращении к доменному имени должен возвращаться список всех возможных ссылок.

**3.3 REST API Teachers**

- способность запускаться на UNIX-системах и Windows 7, 8, 10;

- поддержка асинхронных HTTP – запросов;

- реализация HATEOAS;

- реализация под возможность MAJOR-обновлений;

- наличие базы данных;

- сервис должен подписывать на события, к примеру, обновления данных в базе данных;

- при обращении к доменному имени должен возвращаться список всех возможных ссылок.

**3.4. WEB Client “Расписание”**

- должен запускаться на UNIX или Windows 7, 8, 10.

- должен иметь приятный интерфейс;

- должен отображать расписание в структурированном формате;

- должен иметь удобную навигацию по другим группам;

- должен иметь быструю навигацию по преподавателям или аудиториям;

- должен иметь фильтрацию по времени, неделям, наименованиям, и другим параметрам;

- должен иметь выделением цветом по категориям;

- должна быть краткая информация по расписанию графика пар;

- должен кешировать последнее место навигации;

- должна быть обращения к API через ссылки, который предоставляет HATEOAS реализация;

- веб клиент должен предоставлять ссылку на скачивание мобильного клиента

**3.5 Mobile Client “Расписание”**

- должен запускаться на **Android** – устройстве версией не старее 7.0

- должен иметь приятный интерфейс;

- должен отображать расписание в структурированном формате;

- должен иметь удобную навигацию по другим группам;

- должен иметь быструю навигацию по преподавателям или аудиториям;

- должен иметь фильтрацию по времени, неделям, наименованиям, и другим параметрам;

- должен иметь выделением цветом по категориям пар;

- должна быть краткая информация по расписанию графика пар;

- должен производить репликацию данных;

- должна быть иконка;

- должна быть обращения к API через ссылки, который предоставляет HATEOAS реализация;

- последнее место в приложении автоматически сохраняется

**3.6 Desktop Client “Диспетчерская”**

- должен запускаться на UNIX или Windows 7, 8, 10;

- должна быть иконка;

- должен быть многостраничный интерфейс;

- должен быть приятный интерфейс;

- должна быть обработка ошибок;

- должно быть сообщение в случае отказа сервиса в работе;

- должно быть сочетание клавиш для быстрой навигации в приложении;

- Поддержка защиты пин-кодом;

- при нажатии на ENTER идет переход на следующий TextBox;

- соблюдения архитектурной целостности приложения;

- должен быть экспорт в Excel файл

- должно быть добавления пары;

- должно быть удаление пары;

- должно быть редактирование пары;

- при добавлении пары ComboBox автоматически заполняются в зависимости от выбранных параметров;

- в случае непредвиденных обстоятельств, в виде того, что

- должны быть подсказки для всех полей для ввода;

- должна быть фильтрация, где отображаются все пары;

- обращение к API идет с помощью ссылок, которые дает HATEOAS реализация;

- должна быть быстрый просмотр, то что пойдет на печать, либо экспорт;

**3.7 Desktop Client “Аудитория”**

- должен запускаться на UNIX или Windows 7, 8, 10;

- должна быть иконка;

- должен быть многостраничный интерфейс;

- должен быть приятный интерфейс;

- должна быть обработка ошибок;

- должно быть сообщение в случае отказа сервиса в работе;

- должно быть сочетание клавиш для быстрой навигации в приложении;

- Поддержка защиты пин-кодом;

- при нажатии на ENTER идет переход на следующий TextBox;

- соблюдения архитектурной целостности приложения;

- должно быть добавления аудитории;

- должно быть редактирование аудитории;

- должно быть смена статуса аудитории со сломана на подготовлена и обратно;

- обращение к API идет с помощью ссылок, которые дает HATEOAS реализация;

- Должна быть фильтрация по разным полям;

**3.8 Desktop Client “Преподаватель”**

- должен запускаться на UNIX или Windows 7, 8, 10;

- должна быть иконка;

- должен быть многостраничный интерфейс;

- должен быть приятный интерфейс;

- должна быть обработка ошибок;

- должна быть подсветка исправных преподавателей

- должно быть сообщение в случае отказа сервиса в работе;

- должно быть сочетание клавиш для быстрой навигации в приложении;

- Поддержка защиты пин-кодом;

- при нажатии на ENTER идет переход на следующий TextBox;

- соблюдения архитектурной целостности приложения;

- должно быть добавления преподавателя;

- должно быть редактирование преподавателя;

- должно быть смена статуса преподавателя с не готов на готов и обратно;

- должна быть подсветка исправных преподавателей

- обращение к API идет с помощью ссылок, которые дает HATEOAS реализация;

- Должна быть фильтрация по разным полям;

**4. Диаграммы**

Для хранения всей информации была создана ER – диаграмма, с учетом микросервисной архитектуры;



Диаграмма, показывающая как будет связана вся система:

