МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Кафедра информационных систем и технологий

**ОТЧЕТ ПО ПРАКТИКЕ**

Вид практики производственная

(учебная, производственная)

Тип практики технологическая (проектно-технологическая) практика

(в соответствии с ОПОП ВО)

Сроки прохождения практики: с 01.07.2024 по 13.07.2024

(в соответствии с календарным учебным графиком)

по направлению подготовки 09.03.01 Информатика и вычислительная техника

(уровень бакалавриата)

направленность (профиль) «Информационные системы»

Обучающийся группы № 6303-090301D\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С. Лутонин

Руководитель практики

доцент кафедры ИСТ, к.т.н., доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.П. Солдатова

Руководитель практики

от профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Цветков

Дата сдачи 13.07.2024

Дата защиты 13.07.2024

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Содержание**

1. Задание(я) для выполнения определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований).
2. Описательная часть.
3. Заключение.
4. Список использованных источников.
5. Отзыв о прохождении практики.
6. Приложения.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение

высшего образования «Самарский национальный исследовательский университет

имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Кафедра информационных систем и технологий

**Задание(я) для выполнения определенных видов работ, связанных с будущей профессиональной деятельностью (сбор и анализ данных и материалов, проведение исследований).**

Обучающемуся Лутонину Александру Сергеевичу

группы 6303-090301D

Направление на практику оформлено приказом по университету № 349-ПР от 28.06.2024 г.

в\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ООО «Ай-Сис Лабс» г. Самара\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование профильной организации или структурного подразделения университета)

|  |  |  |
| --- | --- | --- |
| Планируемые результаты освоения образовательной программы (компетенции) | Планируемые результаты практики | Содержание задания |
| ПК-2 Способен осуществлять разработку тестовых случаев, проведение тестирования и исследование результатов.  ПК-2.2 Проводит тестирование по разработанным тестовым случаям, проводит анализ результатов тестирования. | Знать: современные методы тестирования и анализа результатов тестов.  Уметь: применять на практике современные методы тестирования и анализа результатов.  Владеть: навыками разработки тестов, проведения тестирования и анализа результатов. | Сделать описание тестов для проведения экспериментов по верификации программного обеспечения.  Провести анализ полученных результатов. |
| ПК-3 Способен осуществлять интеграцию программных модулей и компонент и верификацию выпусков программного продукта.  ПК-3.1 Разрабатывает процедуры интеграции программных модулей. | Знать: методы интеграции программных модулей.  Уметь: применять на практике методы интеграции программных модулей и компонентов.  Владеть: навыками разработки процедур интеграции программных модулей и компонентов | Провести анализ современных методов и технологий интеграции программных модулей.  Сделать обоснование выбора используемых технологий интеграции программных модулей, описание основных алгоритмов системы Event API. |
| ПК-3 Способен осуществлять интеграцию программных модулей и компонент и верификацию выпусков программного продукта.  ПК 3.2 Осуществляет интеграцию программных модулей и компонентов и верификацию выпусков программного продукта. | Знать: методы интеграции программных модулей и компонент, и методологию разработки контрольных примеров для их верификации.  Уметь: обосновывать выбор используемых методов интеграции программных модулей и компонент для разработки программных продуктов.  Владеть: навыками реализации интеграции программных модулей и компонент и разработки контрольных примеров для их верификации. | Сделать описание контрольных примеров для проведения экспериментов по верификации программного обеспечения и анализ полученных результатов. |
| ПК-4 Способен осуществлять администрирование сетевой подсистемы инфокоммуникационной системы организации  ПК-4.1 Осуществляет настройку сетевых элементов инфокоммуникационной системы. | Знать: методы настройки сетевых элементов инфокоммуникационной системы.  Уметь: применять на практике методы настройки сетевых элементов инфокоммуникационной системы.  Владеть: навыками настройки сетевых элементов инфокоммуникационной системы | Сделать описание применяемых методов настройки сетевых элементов, контроля использования ресурсов и диагностики отказов и ошибок сетевых устройств и программного обеспечения системы Event API*.* |
| ПК-4 Способен осуществлять администрирование сетевой подсистемы инфокоммуникационной системы организации  ПК-4.2 Осуществляет контроль использования ресурсов и проводит диагностику отказов и ошибок сетевых устройств и программного обеспечения. | Знать: методы контроля использования ресурсов и методы диагностики отказов и ошибок сетевых устройств и программного обеспечения.  Уметь: применять на практике методы контроля использования ресурсов, а также методы диагностики отказов и ошибок сетевых устройств и программного обеспечения.  Владеть: навыками реализации контроля использования ресурсов, а также проведения диагностики отказов и ошибок сетевых устройств и программного обеспечения. | Провести анализ методических основ контроля использования ресурсов диагностики отказов и ошибок сетевых устройств и программного обеспечения.  Сделать описание применяемых методов контроля использования ресурсов и диагностики отказов и ошибок сетевых устройств и программного обеспечения системы Event API*.* |
| ПК-5 Способен осуществлять разработку компонентов системных программных продуктов  ПК 5.2 Создает инструментальные средства программирования. | Знать: современные технологии разработки инструментальных средств программирования.  Уметь: применять современные технологии для разработки инструментальных средств программирования.  Владеть: навыками разработки инструментальных средств программирования. | Провести анализ современных методов и технологий разработки инструментальных средств программирования.  Сделать обоснование выбора используемых технологий разработки инструментальных средств программирования, описание основных алгоритмов системы Event API*.* |
| ПК-6 Способен осуществлять разработку требований и проектирование программного обеспечения.  ПК-6.1 Проводит анализ требований и проектирует программное обеспечение. | Знать: основные методы проектирования программного обеспечения.  Уметь: применять на практике основные методы проектирования программного обеспечения.  Владеть: навыками проектирования программного обеспечения. | Провести анализ современных методов проектирования программного обеспечения.  Сделать обоснование выбора используемых методов проектирования программного обеспечения. |
| ПК-6 Способен осуществлять разработку требований и проектирование программного обеспечения.  ПК-6.3 Демонстрирует способность понимать, совершенствовать и применять современный инструментарий в рамках использования проектной методологии в профессиональной деятельности. | Знать: современные инструментальные средства проектирования программного обеспечения.  Уметь: применять на практике современные инструментальные средства проектирования программного обеспечения.  Владеть: навыками использования современных инструментальных средств для проектирования программного обеспечения. | Провести анализ современных инструментальных средств проектирования программного обеспечения.  Сделать обоснование выбора используемых инструментальных средств проектирования программного обеспечения.  Сделать описание проекта программного обеспечения информационной системы Event API*.* |
| ПК-7 Способен осуществлять выполнение работ и управление работами по созданию (модификации) и сопровождению информационных систем, автоматизирующих задачи организационного управления и бизнес-процессы.  ПК-7.3 Разрабатывает прототипы информационных систем. | Знать: технологии разработки прототипов информационных систем.  Уметь: применять на практике технологии разработки прототипов информационных систем.  Владеть: навыками разработки прототипов информационных систем. | Провести анализ современных технологий разработки прототипов информационных систем.  Сделать обоснование выбора используемых технологий разработки прототипов информационных систем. |
| ПК-7 Способен осуществлять выполнение работ и управление работами по созданию (модификации) и сопровождению информационных систем, автоматизирующих задачи организационного управления и бизнес-процессы.  ПК-7.5 Разрабатывает базы данных информационных систем. | Знать: технологии проектирования логических и физических моделей баз данных информационных систем.  Уметь: применять на практике технологии проектирования логических и физических моделей баз данных информационных систем.  Владеть: навыками разработки баз данных информационных систем. | Провести анализ современных технологий разработки логических и физических моделей баз данных информационных систем.  Сделать обоснование выбора используемых технологий разработки логической и физической моделей базы данных информационной системы Event API*.*  Сделать описание логической и физической моделей базы данных информационной системы Event API*.* |

Дата выдачи задания 01.07.2024.

Срок представления на кафедру отчета о практике 13.07.2024.

Руководитель практики

доцент кафедры ИСТ, к.т.н., доцент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.П. Солдатова

*(подпись)*

Руководитель практики

от профильной организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Цветков

*(подпись)*

Задание принял к исполнению

обучающийся группы № 6303-090301D \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.С. Лутонин

*(подпись)*

Описательная часть

1 Анализ современных методов, технологий и инструментальных средств проектирования программного обеспечения, прототипов информационных систем, а также технологий разработки логических и физических моделей баз данных

Современные методы и технологии проектирования программного обеспечения играют ключевую роль в создании высококачественных, надёжных и эффективных веб-приложений.

Существует множество языков для веб-разработки, приведу пример некоторых из них:

* HTML;
* CSS;
* JavaScript;
* PHP;
* Java.

HTML (HyperText Markup Language) - язык гипертекстовой разметки сайта, — код, помогающий структурировать содержание каждой веб-страницы. С помощью HTML разработчик собирает «скелет». После чего работает с CSS-кодом для стилизации страницы — покрывает «кожей».

Плюсы html:

* подходит почти для всех браузеров, а код можно написать в любом текстовом редакторе;
* HTML можно использовать бесплатно;
* HTML интегрируется с другими языками программирования.

Минусы html:

* все повторяющиеся изменения на сайте придётся вносить вручную;
* если нужно, чтобы сайт выглядел красиво и реагировал на действия пользователя, придётся учить ещё CSS и JS [1].

**CSS (Cascading Style Sheets)** — язык описания внешнего вида HTML-документа. Это одна из базовых технологий в современном интернете. Практически ни один сайт не обходится без CSS, поэтому HTML и CSS действуют в единой связке [2].

Плюсы css:

* если вы освоите CSS-верстку, будете с легкостью разбираться в коде, менять стили;
* текстовые браузеры и прочие ограниченные устройства вывода информации беспроблемно «примут» макет;
* сайт более быстро отображается в браузерах.

Минусы css:

* не работает в старых браузерах, таких как Internet Explorer 4, Netscape Navigator 4 и прочие подобные;
* CSS-верстку (создание и позиционирование блоков) современные браузеры хотя и поддерживают, но не без проблем [3].

JavaScript – это мультипарадигменный язык программирования, который обычно применяется в качестве встраиваемого инструмента для программного доступа к различным объектам приложений. С помощью этого языка реализуется возможность реакции страницы или отдельных ее элементов на действия посетителя. Сегодня JavaScript является базовым языком программирования для браузеров. Он полностью совместим с операционными системами Windows, Linux, Mac OS, а также всеми популярными мобильными платформами.

Плюсы JavaScript:

* можно быстро интегрировать в код страницы любые теги;
* можно определять внешний вид элементов;
* можно выполнять позиционирование объектов;
* можно получать пользовательские данные.

Минусы JavaScript:

В чистом виде, то есть без специальных надстроек язык JS может функционировать только в рамках браузера. Но даже здесь есть ограничения, обусловленные требованиями безопасности. В общем, этот язык программирования создан для браузеров и интернета, а за их пределами он и не должен был использоваться [4].

PHP — это популярный язык программирования, особенно в среде веб-разработчиков. Автором изначальной версии является Расмус Лердорф, идея которого состояла в разработке набора инструментов для упрощения процесса создания динамических веб-страниц.

Плюсы Php:

* код PHP легко читается независимо от способа использования;
* PHP совместим со всеми популярными платформами. Написанные на нем приложения успешно работают на различном серверном ПО;
* PHP позволяет добиться максимальной производительности приложений, написанных на нем, с ростом аппаратных ресурсов. Веб-приложения, распределенные на несколько серверов, способны справляться с существенными нагрузками.

Минусы Php:

* наличие «дыр» в системе безопасности;
* нацеленность на веб-разработку. Несмотря на наличие прочих сфер применения, язык программирования PHP продолжает ассоциироваться с веб;
* некоторые противоречия синтаксиса. Код содержит элементы, характерные для различных языков, а также различные наименования встроенных функций, в связи с чем может возникать путаница [5].

Java относится к категории объектно-ориентированных языков программирования. Он был создан в девяностых годах прошлого столетия специалистами американского производителя ПО Sun Microsystems.

Плюсы Java:

* самоуправление — созданный вами код сможет работать на всех платформах, которые поддерживают объектно-ориентированный язык программирования;
* безопасность — в большой степени достигается из-за строгости статичной типизации;
* многофункциональность;
* несложный синтаксис.

Минусы Java:

* невысокая скорость, сравнительно с языками С и С++;
* необходимость в наличии больших объемов памяти;
* так как Java относится к высокоуровневым языкам, отсутствует поддержка низкоуровневого программирования. К примеру, здесь нет указателей [6].

Современные информационные системы невозможно представить без эффективных баз данных, которые являются основой хранения, обработки и управления данными. Ключевыми этапами создания таких систем являются разработка логических и физических моделей баз данных. Логическая модель определяет структуру данных и их взаимосвязи на концептуальном уровне, обеспечивая целостность и согласованность данных. Физическая модель, в свою очередь, отвечает за реализацию этой структуры на уровне конкретной системы управления базами данных (СУБД).

Программы для создания логической модели базы данных:

* **ERwin**: Это одно из ведущих коммерческих решений для проектирования баз данных. ERwin предоставляет мощные инструменты для создания ER-диаграмм и логических моделей данных;
* **Microsoft Visio**: Visio предлагает широкие возможности для создания различных видов диаграмм, включая ER-диаграммы и логические модели данных. Это коммерческое ПО, входящее в пакет Microsoft Office;
* **Oracle SQL Developer Data Modeler**: Это бесплатное ПО от Oracle, специально разработанное для моделирования данных. Оно поддерживает создание ER-диаграмм, логических и физических моделей;
* **MySQL Workbench**: Это бесплатная программа, предоставляемая MySQL, которая включает инструменты для создания ER-диаграмм и моделирования баз данных.

Существует множество систем управления базами данных, приведу пример некоторых из них:

MySQL - это система управления базами данных (СУБД), позволяющая хранить, организовывать большие объемы данных, и манипулировать ими.

MySQL предоставляет мощные функции обработки данных. Этот программный продукт использует стандартный язык SQL для обработки данных. При этом он дает пользователю возможности для создания таблиц, выполнения запросов, вставки, обновления и удаления данных без знания языка запросов SQL.

Плюсы MySql:

* MySQL имеет в своем функционале различные функции безопасности, включая возможность установки привилегий пользователя, шифрование данных, аутентификацию и аудит;
* MySQL обладает хорошей производительностью и быстродействием благодаря оптимизированным алгоритмам выполнения запросов;
* MySQL может использоваться как для небольших веб-приложений, так и для серьезных корпоративных систем.

Минусы MySql:

* MySQL может иметь проблемы с производительностью при обработке больших объемов данных или когда требуется обработка сложных запросов;
* MySQL имеет ограниченный набор типов данных по сравнению с некоторыми другими СУБД;
* MySQL может быть сложным для администрирования, особенно для новичков [7].

Microsoft SQL Server (MS SQL) – это реляционная система управления базами данных, которую применяют для работы с данными веб-приложений. Данная СУБД использует свой диалект [SQL](https://workspace.ru/tools/language/sql/) – Transact-SQL. По данным Stack Overflow за 2022 год, Microsoft SQL Server занимает 5 место по популярности среди всех СУБД.

Плюсы MS SQL:

* интеграция с продуктами и сервисами Microsoft, включая облачный сервис Microsoft Azure и ПО для бизнес-анализа – Microsoft PowerBI;
* поддержка Kubernetes для масштабирования;
* продвинутая система безопасности, которую Microsoft позиционирует как лучшую в отрасли;
* расширенная аналитика благодаря интеграции с Big Data и службами машинного обучения [8].

2 Анализ современных методов и технологий разработки инструментальных средств программирования и методов интеграции программных модулей

Развитие информационных технологий и рост требований к качеству программного обеспечения привели к необходимости создания и усовершенствования инструментальных средств программирования. Интегрированные среды разработки (IDE) играют ключевую роль в этом процессе, предоставляя разработчикам мощные и удобные инструменты для написания, тестирования и отладки программного кода. Современные IDE значительно расширяют возможности программистов, объединяя в себе редакторы кода, компиляторы, отладчики, системы контроля версий и многие другие инструменты в единый интерфейс.

Существует множество сред разработки, приведу пример некоторых из них:

* Eclipse;
* Visual Studio Code;
* IntelliJ IDEA.

Eclipse – это интегрированная среда разработки мультиплатформенных приложений с открытым исходным кодом. Главным образом это приложение работает как платформа для программирования, и оно может компилировать и отлаживать программы на различных языках, но наибольшей популярностью она пользуется у программистов на Java. Тем не менее модульность этой среды позволяет использовать ее для программирования на C, Python и на многих других языках [9].

Плюсы:

* множество плагинов. У Eclipse едва ли не самое большое число надстроек — «на все случаи жизни»;
* активное сообщество. Помогает быстрее освоить среду разработки, выпускает новые плагины;
* отличные компилятор и отладчик;
* благодаря плагинам и настройкам можно полностью персонализировать Eclipse;
* это open-source проект, абсолютно бесплатный;
* высокая функциональность.

Минусы:

* как и любой функциональный продукт, Eclipse может показаться новичку слишком сложным;
* нет гарантий надежности. Так как плагины создаются сообществом, за их качество отвечает только разработчик [10].

Visual Studio Code – редактор кода с функциями IDE, разработанный компанией Microsoft. Продукт распространяется по свободной лицензии и идет с открытым исходным кодом. Чаще всего, Visual Studio Code применяют в разработке сайтов. По результатам опроса Stack Overflow за 2022 год, Visual Studio Code является наиболее популярным инструментом в общем рейтинге для редакторов кода и IDE [11].

Плюсы:

* работает на MacOS, Ubuntu и Windows. Пока недоступен на Android и iOS;
* простой open-source редактор и плагины — платить не надо;

Минусы:

* низкая функциональность. Несмотря на поддержку .NET-платформы, VCS неудобен для сложных проектов;
* сомнительная надежность. Многие надстройки имеют низкое качество сборки и не всегда выполняют даже основные функции [10].

IntelliJ IDEA - cреда разработки от компании JetBrains обладает обширным набором инструментов для создания приложений. Для пользователей IntelliJ IDEA доступна внушительная экосистема плагинов, которые расширяют функционал, набор языков, фреймворков и дополнительных инструментов.

Поддерживаемые языки: Java, JavaScript, PHP, Python, Ruby, Go, SQL, С/С++, Groovy, Kotlin, Scala, TypeScript. IDE доступна для Windows, macOS, Linux [12].

Современные программные системы состоят из множества отдельных компонентов, которые должны работать вместе, обеспечивая надежность, производительность и масштабируемость. Эффективная интеграция программных модулей позволяет не только упростить разработку и сопровождение программного обеспечения, но и значительно повысить его качество и гибкость.

Доступны 3 вида интеграции:

* облачная;
* локальная;
* гибридная.

Каждый из видов, в свою очередь, различается по следующим методам:

* на уровне брокеров. Данный вид интеграции считается универсальным. При необходимости задействуется дополнительный модуль – брокер. Он подключается к другим необходимым модулям. Такой вид интеграции считается сложным в реализации, требует определенных знаний;
* на уровне интерфейсов. Целью данного вида интеграции изначально было объединение разноплановых приложений. Сложность такого типа в последовательном подключении элементов. Это вызывает ряд ошибок в процессе взаимодействия. К тому же часто встречаются Legacy софт;
* на уровне сервисов. Здесь при помощи программного обеспечения осуществляется фиксация данных и интерфейсов с двух сторон. Это один из немногих видов неавтоматизированной интеграции, то есть участие человека здесь остро необходимо;
* функционально-прикладная и организационная интеграция. Ключевым моментом здесь является объединение нескольких схожих или однотипных приложений. Этот вид наиболее удобен для крупных предприятий, корпораций. Именно за счет интеграции этого вида удается снизить затраты на обслуживающий персонал, так как практически все процессы максимально доступны;
* корпоративные программные приложения. Здесь используются не только приложения внутри системы, но и сам исполняемый код. Специализированное ПО и API позволяют использовать отдельные компоненты приложений в единое ядро. Такую систему легче администрировать и масштабировать при необходимости. Доступ к ядру осуществляется при помощи стандартных протоколов доступа, например, SOAP [13].

3 Анализ методических основ настройки сетевых элементов, контроля использования ресурсов и диагностики отказов и ошибок сетевых устройств и программного обеспечения

Ошибки в сетевой инфраструктуре можно классифицировать по разным категориям. Эти ошибки могут приводить к различным проблемам, начиная от небольших неудобств и заканчивая серьезными сбоями, влияющими на бизнес-процессы. Распространённые сетевые ошибки:

* ошибки оборудования - физические неисправности устройств, таких как маршрутизаторы, коммутаторы, серверы и кабели. Они могут возникать из-за износа, производственных дефектов или внешних факторов, таких как электропомехи и физическое повреждение;
* ошибки программного обеспечения - возникают из-за сбоев в операционных системах, прошивках сетевых устройств, приложениях и конфигурациях. Эти ошибки могут приводить к неожиданным поведениям, нарушению работы сетевых сервисов и уязвимостям безопасности;
* конфигурационные ошибки - связаны с неправильными настройками сетевых устройств и сервисов. Они могут быть вызваны как человеческим фактором, так и ошибками при автоматизированных настройках;
* ошибки сети передачи данных - возникают на уровне передачи данных и связаны с потерями пакетов, задержками, дупликацией данных и другими проблемами, влияющими на качество и эффективность сетевых соединений;
* ошибки безопасности - уязвимости и недочеты в защите сети, которые могут привести к несанкционированному доступу, утечке данных и другим видам атак.

Виды ошибок программного обеспечения:

* синтаксические ошибки - ошибки в коде, возникающие из-за неправильного использования синтаксиса языка программирования. Они обнаруживаются компилятором или интерпретатором и препятствуют выполнению программы;
* логические ошибки - ошибки, при которых программа выполняется без синтаксических ошибок, но не достигает ожидаемого результата из-за неправильной логики;
* ошибки времени выполнения - ошибки, возникающие во время выполнения программы, часто связанные с неверными данными или состоянием системы;
* ошибки при работе с памятью - ошибки, связанные с неправильным управлением памятью, которые могут приводить к утечкам памяти или нестабильной работе программы;
* ошибки ввода-вывода - ошибки, связанные с неправильной обработкой ввода и вывода данных;
* ошибки взаимодействия с базами данных - ошибки, связанные с некорректными запросами к базам данных или неправильной обработкой результатов запросов.

Контроль использования ресурсов помогает обеспечить стабильную и эффективную работу программного обеспечения. Ресурсы включают в себя процессорное время, оперативную память, сетевые соединения и дисковое пространство. Распространённые методы для контроля ресурсов:

* мониторинг — это непрерывный процесс отслеживания состояния системы и ее компонентов. Он позволяет собирать данные о текущем состоянии ресурсов и выявлять потенциальные проблемы;
* профилирование — это анализ работы программы для определения узких мест в использовании ресурсов. С его помощью можно выявить, какие части программы потребляют наибольшее количество процессорного времени или памяти;
* оптимизация кода — это процесс улучшения кода для повышения его эффективности и сокращения использования ресурсов;
* использование кэша - позволяет уменьшить количество обращений к ресурсоемким операциям, например, к базам данных.

Диагностика отказов — это процесс выявления и анализа причин сбоев в работе программного обеспечения или системы. Отказы могут быть вызваны различными факторами, включая аппаратные сбои, ошибки в коде, некорректные настройки, аномальные нагрузки и другие проблемы. Эффективная диагностика помогает быстро обнаружить и устранить причины отказов, минимизируя время простоя и предотвращая повторение проблем:

* логирование — один из самых важных методов диагностики отказов, поскольку позволяет собирать и анализировать информацию о работе приложения и системе в целом. Основная идея заключается в том, чтобы записывать события, ошибки и другую важную информацию в лог-файлы, которые затем можно использовать для анализа и поиска причин сбоев;
* трассировка и мониторинг позволяют отслеживать и анализировать запросы, проходящие через различные компоненты системы, а также мониторить состояние системы в реальном времени;
* анализ дампов памяти и файлов — метод, позволяющий детально исследовать состояние приложения в момент сбоя и выявлять проблемы, связанные с управлением памятью и обработкой данных. Он помогает выявлять утечки памяти и другие ошибки управления памятью, такие как доступ к уже освобожденной памяти или переполнения буфера.

4 Обоснование выбора используемых методов, технологий и инструментальных средств проектирования программного обеспечения, прототипов информационных систем, а также технологий разработки логической и физической моделей базы данных. Описание проекта программного обеспечения информационной системы и описание логической и физической моделей базы данных

В рамках реализации проекта были использованы Java, ERwin и MySQL. Java, будучи основным языком программирования, применялся для разработки серверной части приложения. Для управления и хранения данных использовалась система управления базами данных MySQL. Причины выбора Java в качестве языка для создания системы Event API:

* позволяет запускать одно и то же приложение на различных платформах без необходимости внесения изменений в код. Это значительно упрощает разработку и развертывание веб-приложений на различных операционных системах и устройствах;
* обладает обширной экосистемой, включающей множество фреймворков и инструментов, специально разработанных для создания веб-приложений;
* поддержка многопоточности и распределенных вычислений.

Причины выбора ERwin для создания логической модели базы данных:

* обладает удобным и интуитивно понятным интерфейсом, который упрощает процесс создания логических схем;
* поддерживает работу с различными системами управления базами данных (СУБД), включая SQL Server, Oracle, MySQL и другие. Это делает его универсальным инструментом, который можно использовать в различных средах и проектах;
* имеется опыт работы с данной программой.

Причины выбора MySQL для создания физической модели базы данных:

* является одной из самых популярных СУБД в мире;
* обеспечивает высокую надежность данных и гарантирует их целостность;
* легко интегрируется с различными инструментами и языками программирования.

Диаграмма вариантов использования системы Event API представлена на рисунке 1:

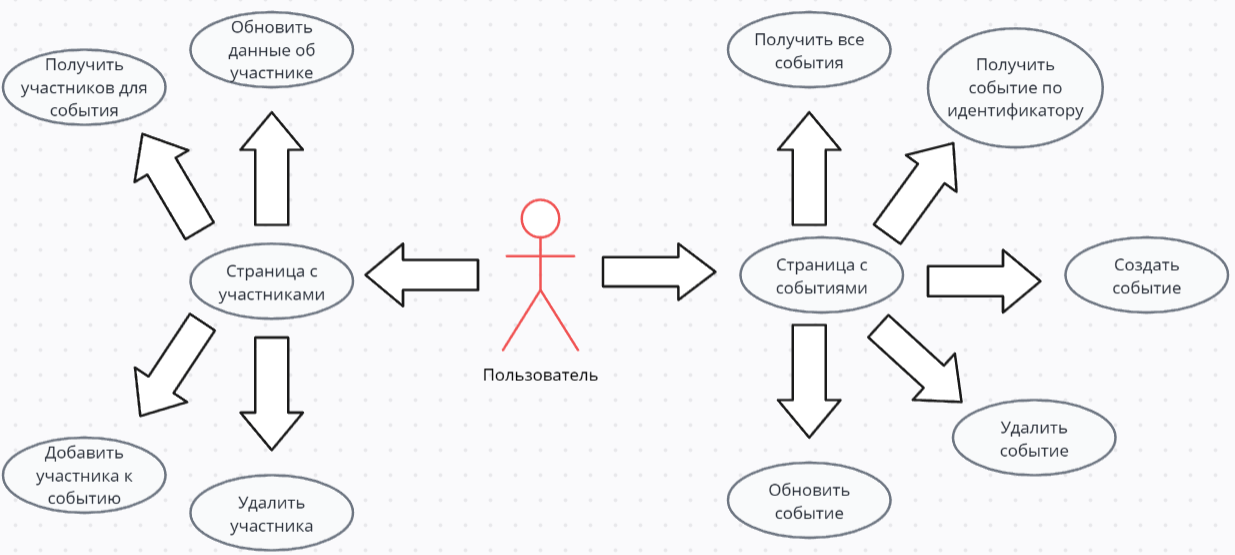


Рисунок 1 – Диаграмма вариантов использования системы Event API

Логическая схема базы данных представлена на рисунке 2:

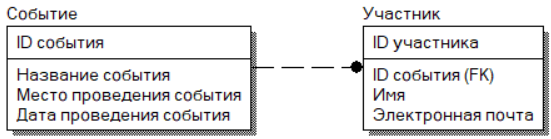


Рисунок 2 – Логическая схема базы данных

Физическая схема базы данных представлена на рисунке 3:

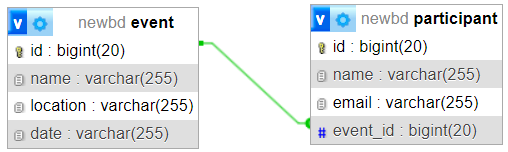


Рисунок 3 – Физическая схема базы данных

На рисунке 4 представлена диаграмма классов, которая показывает набор классов и их связь между собой:

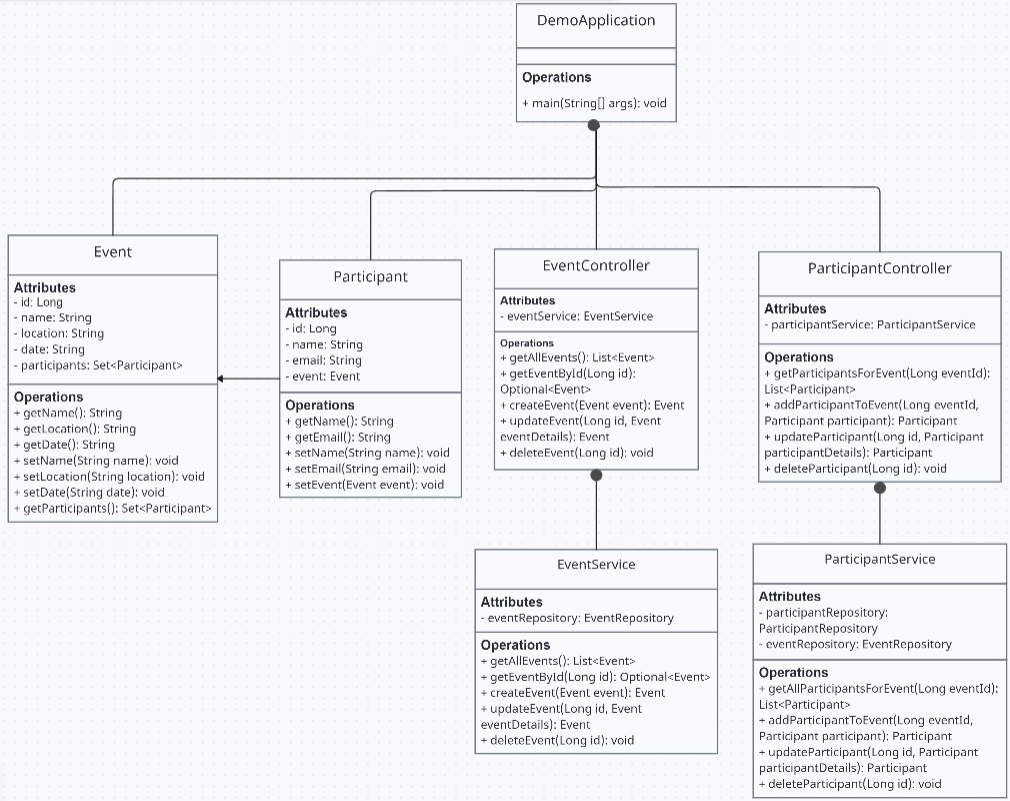


Рисунок 4 – Диаграмма классов

5 Обоснование выбора используемой технологии и среды разработки инструментальных средств программирования и методов интеграции программных модулей, описание основных алгоритмов

В качестве среды разработки был выбран Eclipse. В качестве метода интеграции программных модулей был выбран корпоративный метод с доступом к ядру с помощью REST API на Spring Boot.

Причины выбора Eclipse в качестве среды разработки:

* предоставляет широкий спектр встроенных инструментов и плагинов для разработки, отладки и тестирования приложений;
* обеспечивает простую интеграцию с различными серверами приложений;
* предлагает интуитивно понятный и многофункциональный редактор кода с подсветкой синтаксиса, автодополнением и рефакторингом;
* имеет огромное сообщество пользователей и разработчиков, что обеспечивает доступ к большому количеству обучающих материалов.

Причины выбора REST API на Spring Boot в качестве метода интеграции программных модулей:

* упрощает процесс создания RESTful сервисов. С его помощью можно быстро развернуть веб-приложение с минимальной настройкой, используя простую и понятную аннотацию для определения маршрутов и обработки HTTP-запросов;
* поддерживает широкий спектр стандартов и протоколов для веб-разработки, таких как JSON, XML, и многие другие;
* предоставляет множество возможностей для настройки и масштабирования приложений.

Для работы с базой данных MySQL был выбран **phpMyAdmin** в качестве графического интерфейса пользователя (GUI). Этот инструмент предоставляет удобный и интуитивно понятный способ управления базами данных и выполнения SQL-запросов.

В качестве локального сервера был использован **XAMPP** из-за его простоты установки и использования, а также поддержки всех необходимых компонентов для веб-разработки, включая Apache, MySQL.

Схема алгоритма создания нового события приведена на рисунке 5:

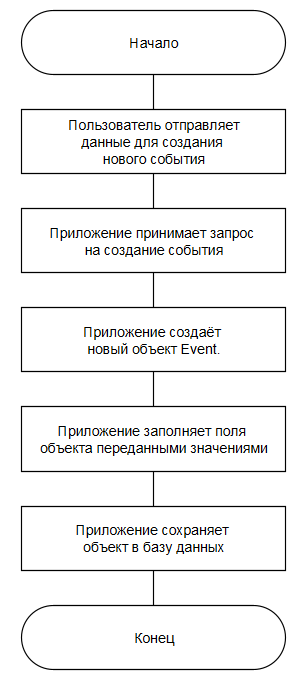


Рисунок 5 – Схема алгоритма создания нового события

Схема алгоритма получения события по id приведена на рисунках 6 и 7:

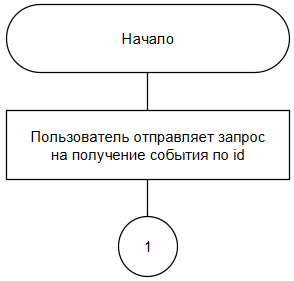


Рисунок 6 – Схема алгоритма получения события по id №1

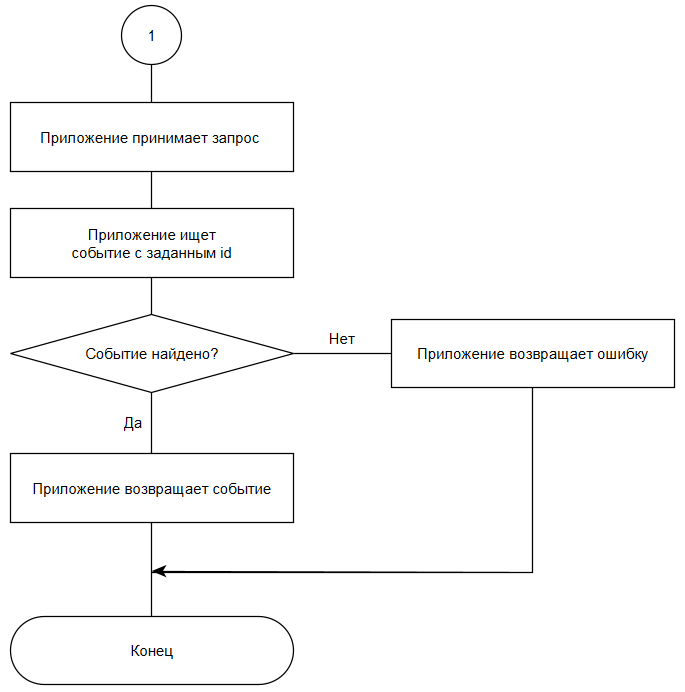


Рисунок 7 – Схема алгоритма получения события по id №2

Схема алгоритма получения всех событий приведена на рисунке 8:

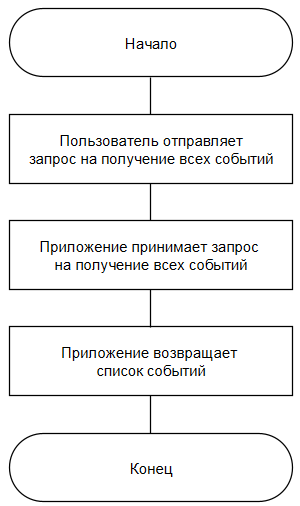


Рисунок 8 – Схема алгоритма получения всех событий

Схема алгоритма обновления события по id приведена на рисунке 9:

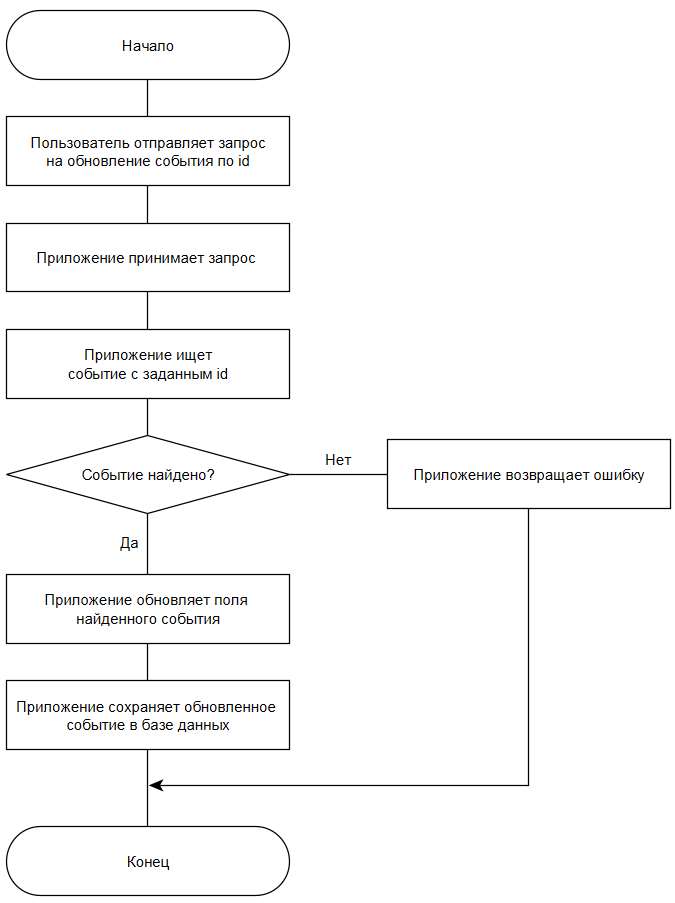


Рисунок 9 – Схема алгоритма обновления события по id

Схема алгоритма добавления участника к событию приведена на рисунке 10:

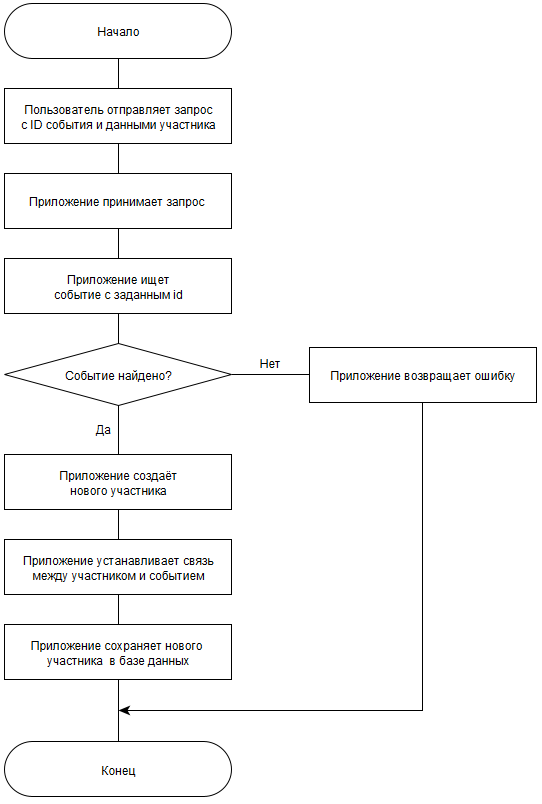


Рисунок 10 – Схема алгоритма добавления участника к событию

Алгоритм удаления события по id аналогичен алгоритму получения события по id. Алгоритмы удаления, получения списка, обновления для участников аналогичны тем же алгоритмам для события.

Листинг программы представлен в приложениях А-Г.

6 Описание применяемых методов настройки сетевых элементов, контроля использования ресурсов и диагностики отказов и ошибок сетевых устройств и программного обеспечения. Описание тестов для проведения экспериментов по верификации программного обеспечения и анализ полученных результатов

Для контроля использования ресурсов использовался инструмент JConsole. Благодаря своей тесной интеграции с JVM и широким возможностям мониторинга. Как встроенный инструмент в JDK, JConsole обеспечивает простой и удобный доступ к ключевым метрикам работы Java-приложений без необходимости в установке дополнительных программных компонентов. Он позволяет отслеживать использование памяти, активность потоков, загрузку процессора и динамическую загрузку классов, предоставляя детализированную информацию в удобном графическом формате.

Для диагностики отказов и ошибок программного обеспечения были выбраны средства Eclipse. Он предлагает мощные и гибкие инструменты для отладки и анализа, которые помогают эффективно выявлять и устранять проблемы в коде и сетевых компонентах. Возможность использовать расширенные функции отладки, такие как пошаговое выполнение, установка точек останова, просмотр значений переменных в реальном времени и анализ стеков вызовов, значительно упрощает процесс диагностики и устраняет необходимость в дополнительных инструментах.

Консоль Eclipse позволяет выводить логи и сообщения об ошибках, связанные с сетевыми операциями. Можно настроить приложение так, чтобы оно выводило детализированные сообщения в консоль, что помогает в диагностике и устранении сетевых ошибок.

Для диагностики отказов и ошибок базы данных были выбраны средства phpMyAdmin. Он предоставляет подробную информацию о текущих соединениях с базой данных, включая количество активных пользователей и сессий, что помогает определить наличие проблем с подключением. Можно просматривать и анализировать активные соединения, выявляя потенциальные проблемы, такие как перегруженные или зависшие сессии, которые могут быть связаны с сетевыми сбоями.

PhpMyAdmin предоставляет инструменты для проверки целостности данных и индексов. Можно выполнить проверки и восстановление таблиц, чтобы убедиться, что данные не повреждены и структура базы данных не нарушена. Это особенно важно при выявлении ошибок, связанных с потерей данных или повреждением информации, которые могут быть вызваны сетевыми сбоями или неисправностями аппаратного обеспечения.

Для тестирования работы веб-приложения использовались следующие инструменты: браузер для отправки GET-запросов и Postman для отправки Post, Delete и Put запросов.

Postman — популярный инструмент для тестирования и разработки API. Он позволяет разработчикам отправлять запросы к API, анализировать ответы и упрощает процессы разработки, тестирования и документирования API.

Добавим 2 события. Для этого отправим 2 Post запроса по адресу “ http://localhost:8080/api/events” с данными событий в формате Json. Результат работы изображен на рисунках 11-13:

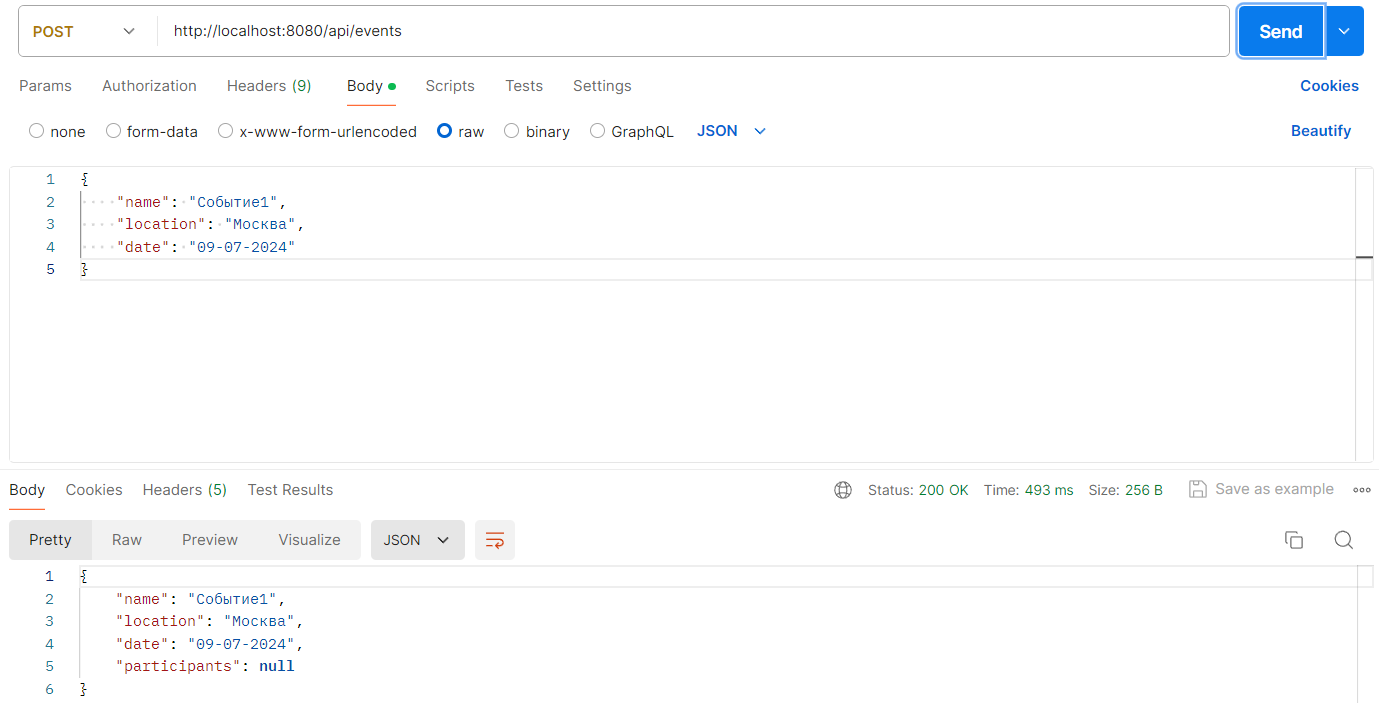
****

Рисунок 11 – Отправка Post запроса на добавление события №1

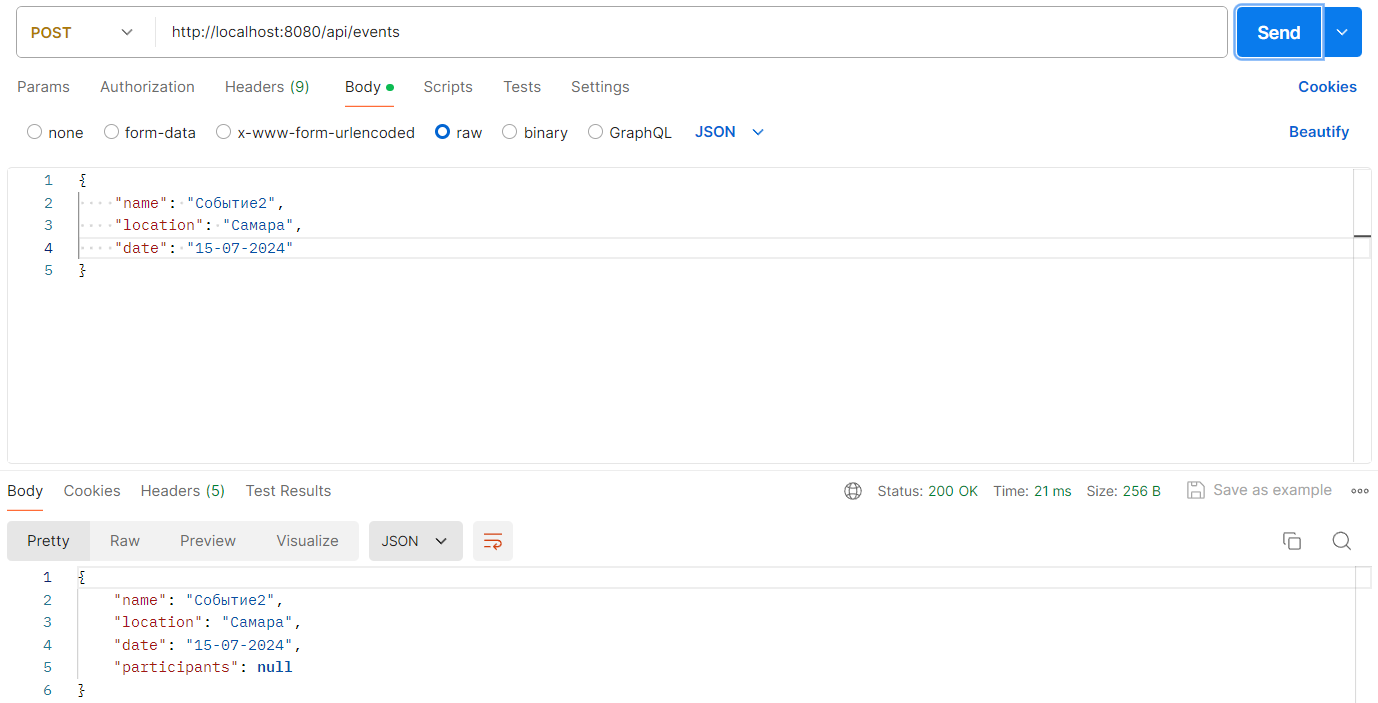
****

Рисунок 12 – Отправка Post запроса на добавление события №2

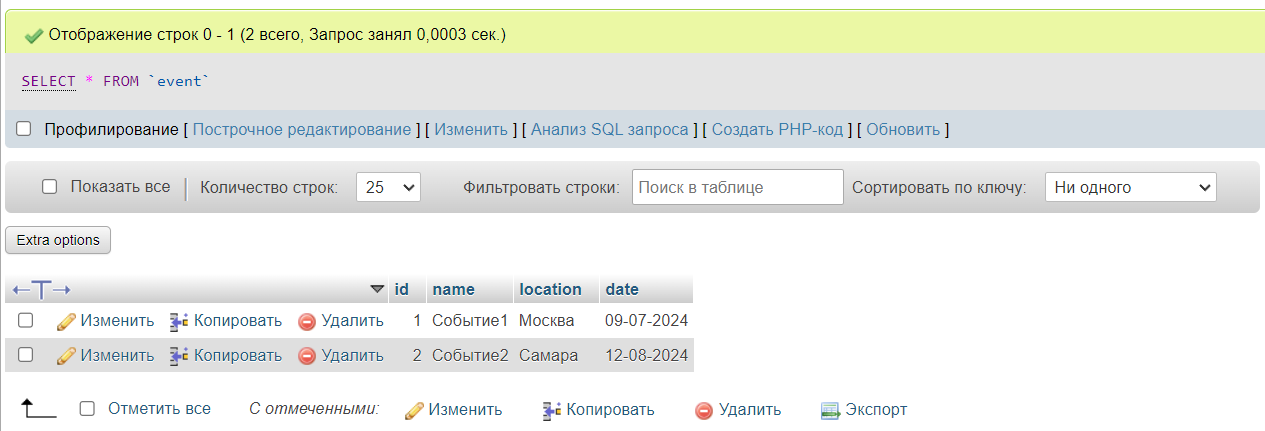


Рисунок 13 – Результат добавления событий

Изменим название 1 события. Для этого отправим Put запрос по адресу “http://localhost:8080/api/events/1”. Результат работы изображен на рисунках 14-15:

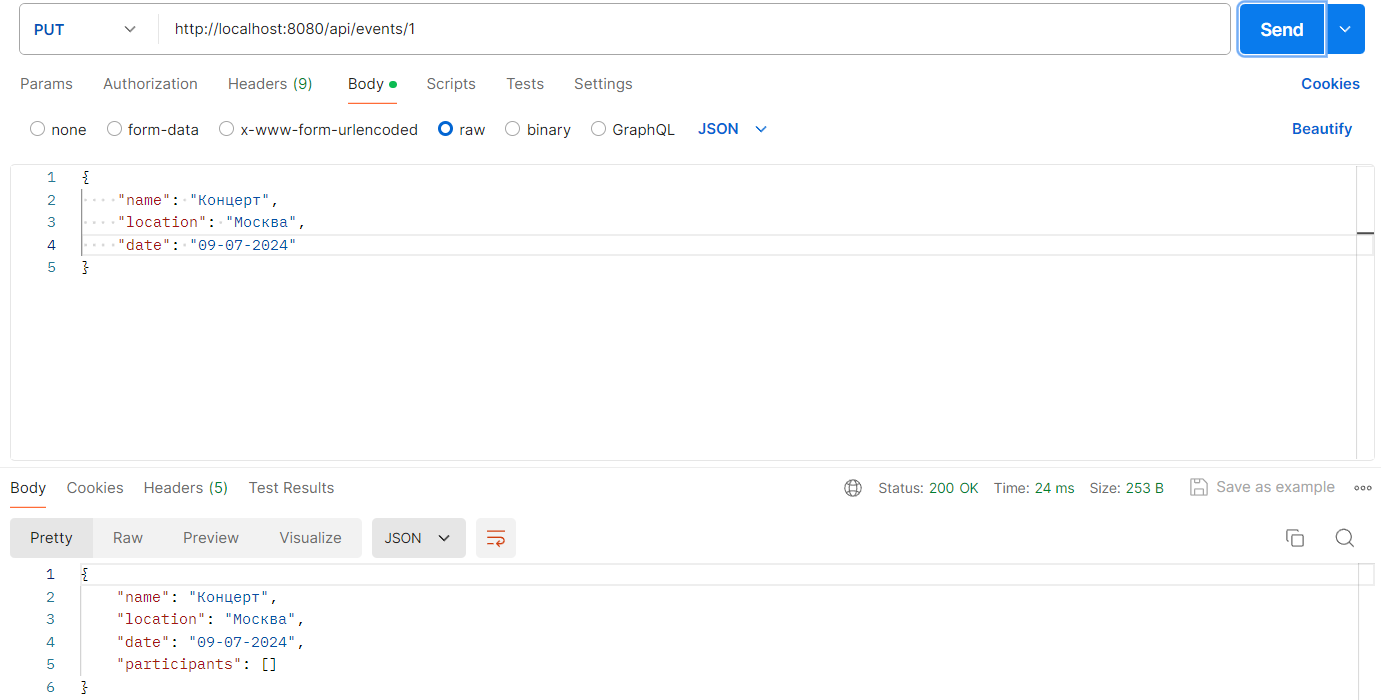


Рисунок 14 – Отправка Put запроса на обновление события

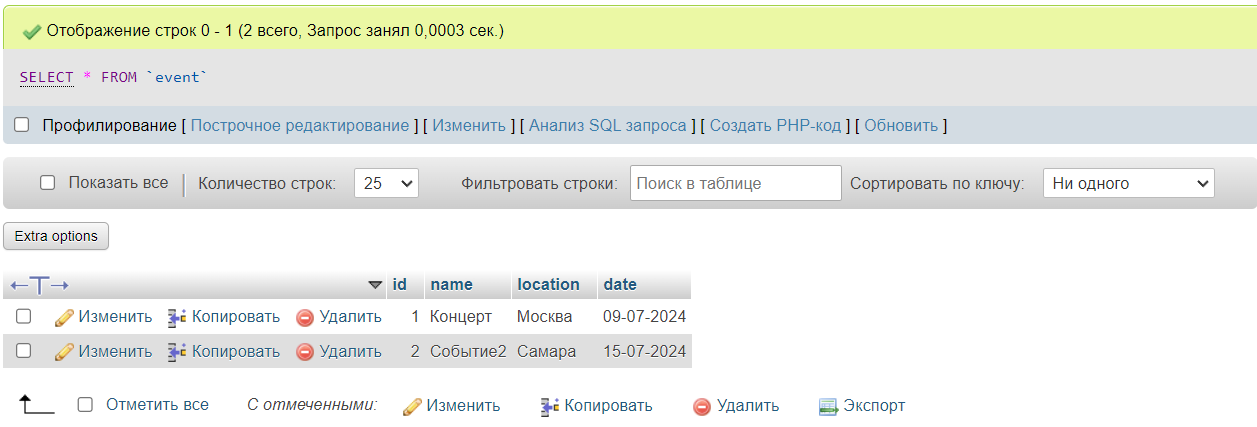


Рисунок 15 – Результат обновления события

Добавим участника к первому событию. Для этого отправим Post запрос по адресу “ http://localhost:8080/api/participants/event/1”. Результат работы изображен на рисунках 16-17:

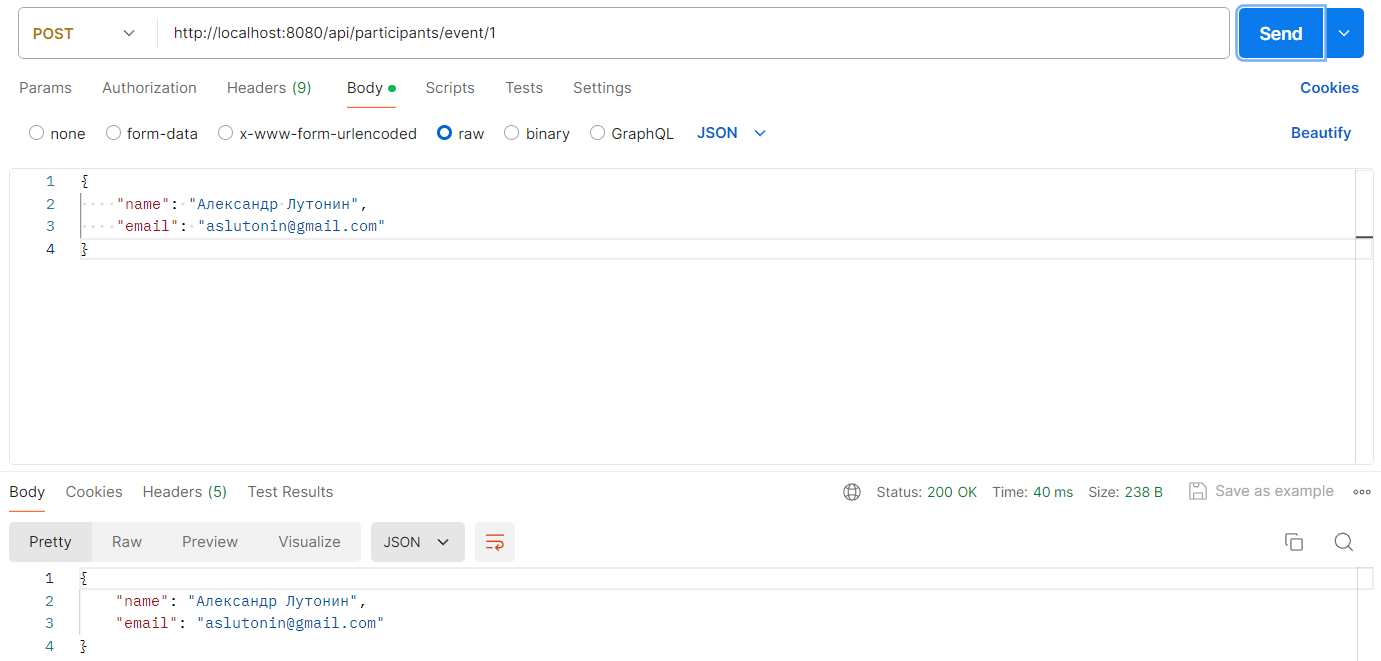


Рисунок 16 – Отправка Post запроса на добавление участника

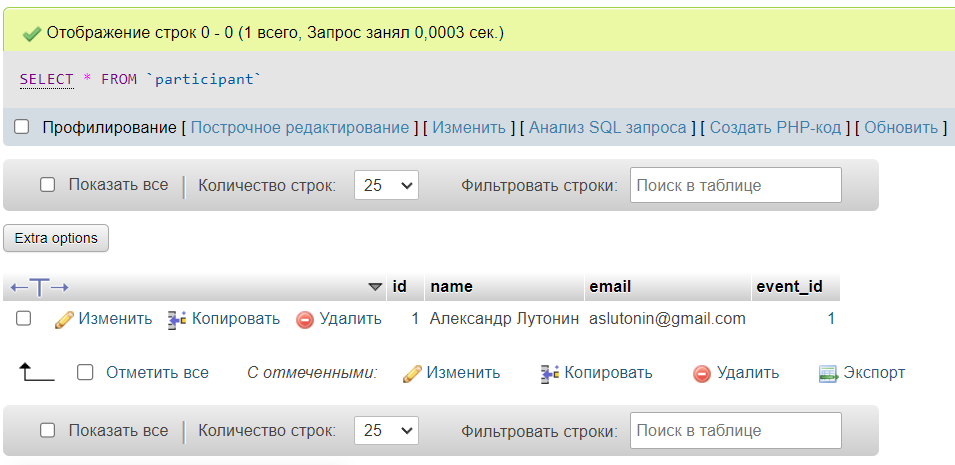


Рисунок 17 – Результат добавление участника

Изменим имя участника. Для этого отправим Put запрос по адресу “http://localhost:8080/api/participants/1”. Результат работы изображен на рисунках 18-19:

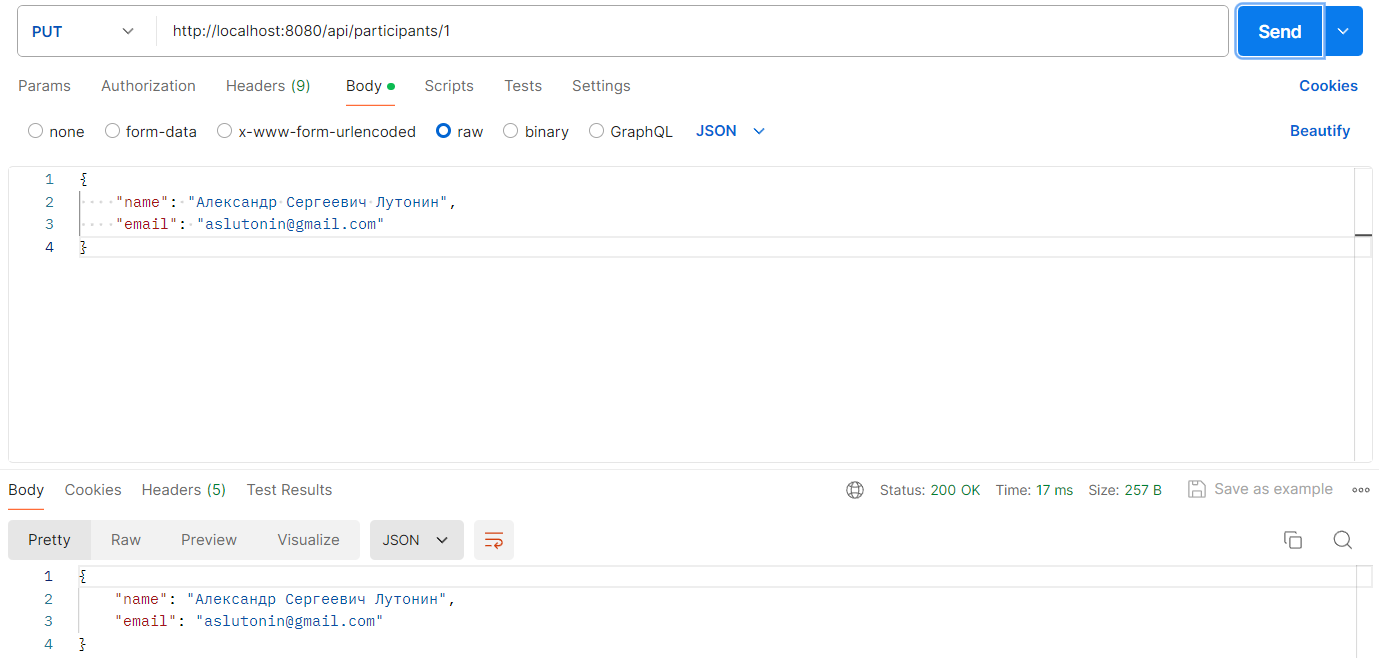


Рисунок 18 – Отправка Put запроса на обновление участника

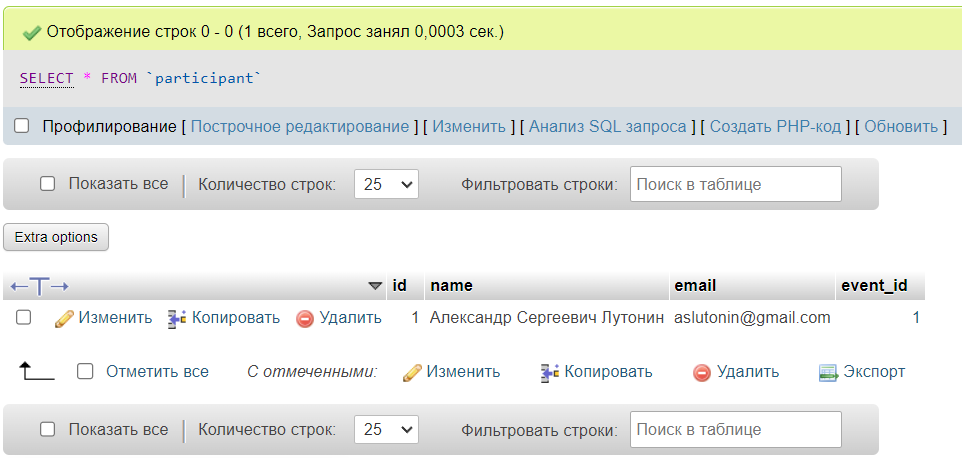


Рисунок 19 – Результат обновления участника

Чтобы получить список всех событий нужно перейти по ссылке “http://localhost:8080/api/events”. Результат работы изображен на рисунке 20:

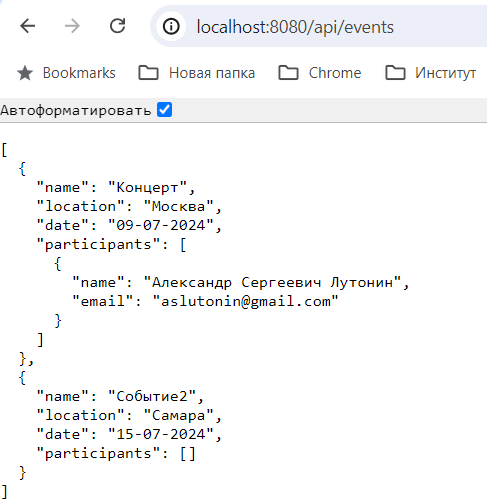


Рисунок 20 – Получение списка всех событий

Чтобы получить событие по id нужно перейти по ссылке “http://localhost:8080/api/events/1”. Результат работы изображен на рисунке 21:

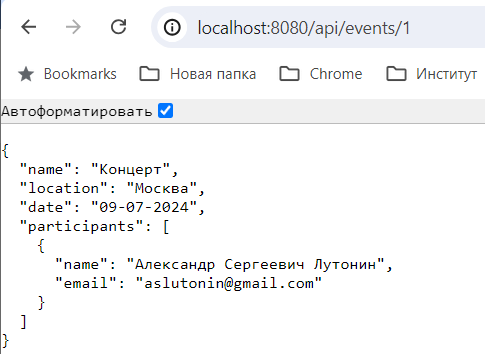


Рисунок 21 – Получение события по id

Чтобы получить список участников 1 события нужно перейти по ссылке “http://localhost:8080/api/participants/event/1”. Результат работы изображен на рисунке 22:

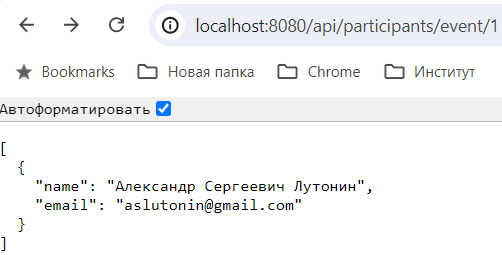


Рисунок 22 – Получение списка участников 1 события

Чтобы удалить участника с id 1 нужно отправить Delete запрос по ссылке “http://localhost:8080/api/participants/1”. Результат работы изображен на рисунках 23-24:

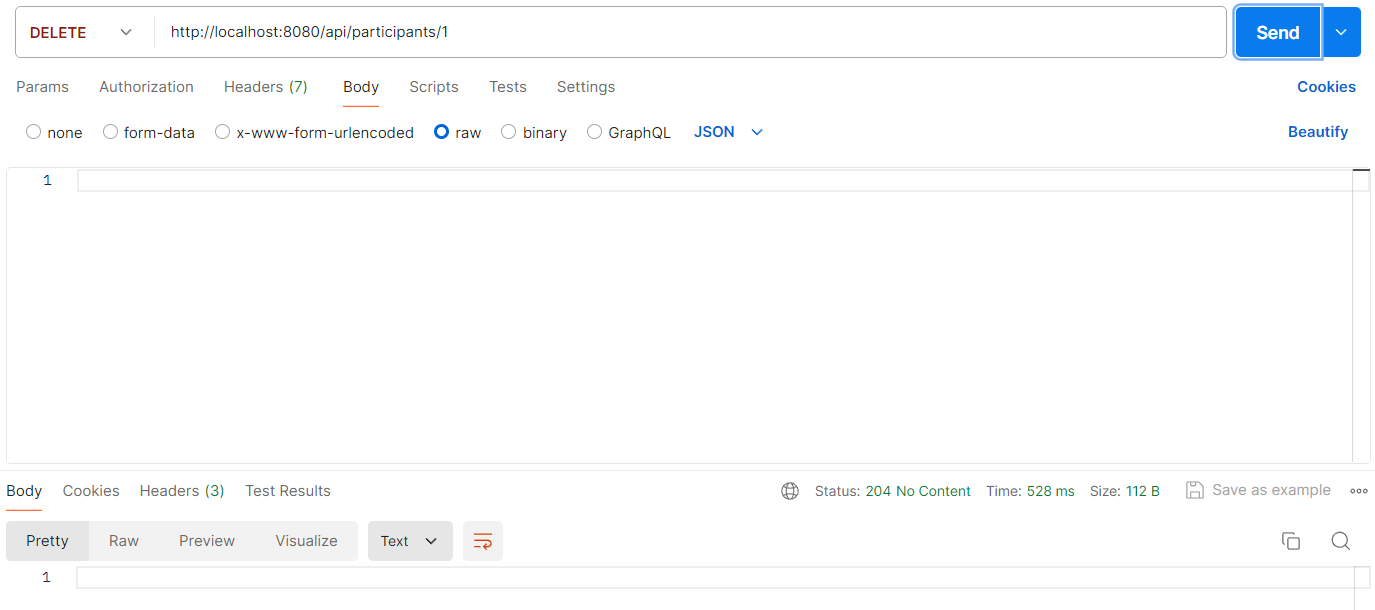


Рисунок 23 – Отправка Delete запроса на удаление участника

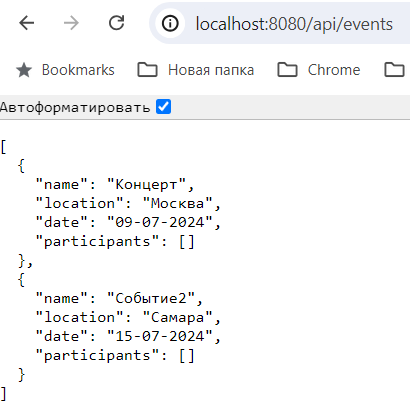


Рисунок 24 – Результат удаления участника

Чтобы удалить событие с id 2 нужно отправить Delete запрос по ссылке “http://localhost:8080/api/events/2”. Результат работы изображен на рисунках 25-26:

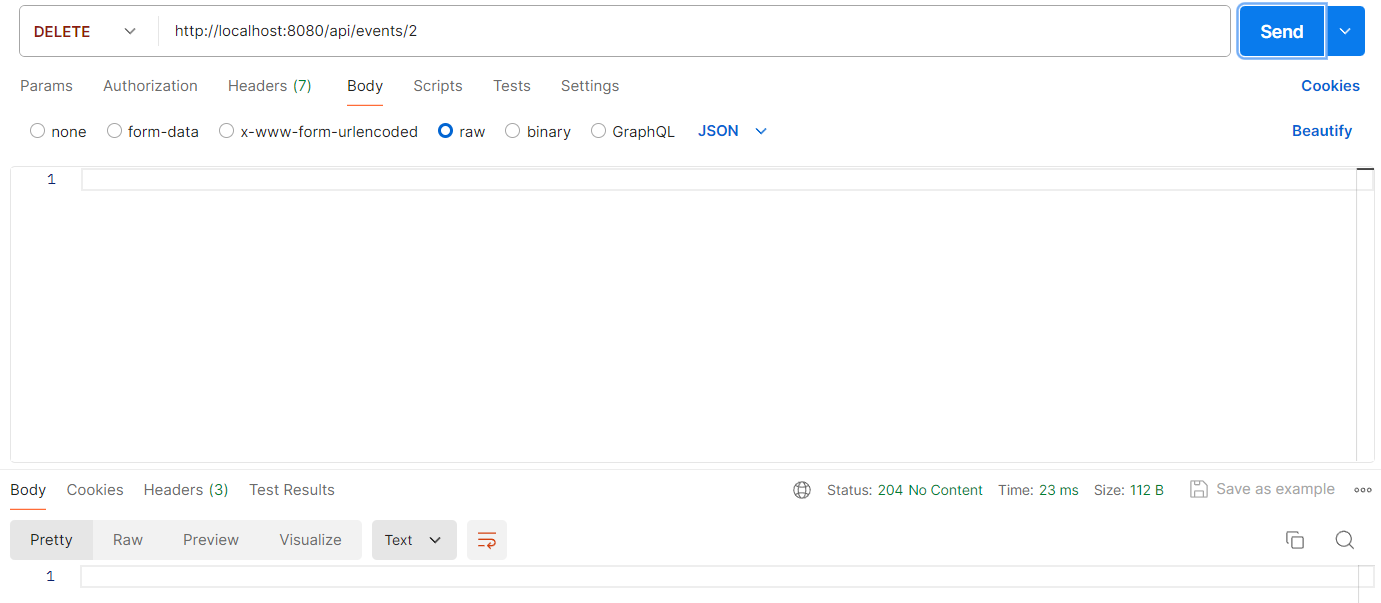


Рисунок 25 – Отправка Delete запроса на удаление события

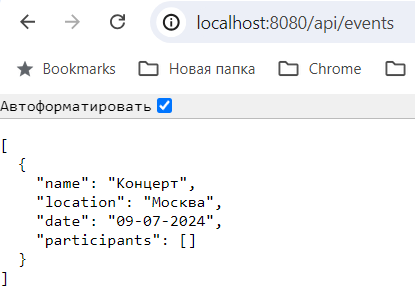


Рисунок 26 – Результат удаления события

# ЗАКЛЮЧЕНИЕ

В процессе прохождения технологической (проектно-технологической) практики были решены следующие задачи:

- проведен анализ современных методов, технологий и инструментальных средств проектирования программного обеспечения, прототипов информационных систем, а также технологий разработки логических и физических моделей баз данных информационных систем;

- проведен анализ современных методов и технологий разработки инструментальных средств программирования и интеграции программных модулей.

- проведен анализ методических основ настройки сетевых элементов, контроля использования ресурсов диагностики отказов и ошибок сетевых устройств и программного обеспечения;

- сделано обоснование выбора используемых методов, технологий и инструментальных средств проектирования программного обеспечения, прототипов информационных систем, а также технологий разработки логической и физической моделей базы данных;

- сделано описание проекта программного обеспечения и логической и физической моделей базы данных информационной системы Event API.

- сделано выбора используемой технологии и среды разработки инструментальных средств программирования и методов интеграции программных модулей, описание основных алгоритмов системы Event API.

- сделано описание применяемых методов настройки сетевых элементов, контроля использования ресурсов и диагностики отказов и ошибок сетевых устройств и программного обеспечениясистемы Event API;

- сделано описание тестов и контрольных примеров для проведения экспериментов по верификации программного обеспечения;

- проведен анализ полученных результатов.

Таким образом, в процессе прохождения практики были освоены все необходимые индикаторы ПК-2.2, ПК-3.1, ПК-3.2, ПК-4.1, ПК-4.2, ПК 5.2, ПК-6.1, ПК-6.3, ПК-7.3, ПК-7.5 компетенций ПК-2, ПК-3, ПК-4, ПК-5, ПК-6 и ПК-7.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Что такое HTML: зачем он нужен, возможности языка гипертекстовой разметки [Электронный ресурс] URL: <https://practicum.yandex.ru/blog/zachem-nuzhen-html/> (дата обращения: 01.07.2024).
2. Что такое CSS — и как на этом вёрстать / Skillbox Media [Электронный ресурс] URL: https://skillbox.ru/media/code/chto\_takoe\_css/ (дата обращения: 01.07.2024).
3. В помощь верстальщику: Статьи - Плюсы и минусы CSS-верстки [Электронный ресурс] URL: <https://web-skyline.narod.ru/articles/41v.htm> (дата обращения: 01.07.2024).
4. JavaScript (JS) – что это такое за язык программирования [Электронный ресурс] URL: <https://blog.ingate.ru/seo-wikipedia/java-script/> (дата обращения: 01.07.2024).
5. Php что это? Чем PHP лучше других языков программирования? [Электронный ресурс] URL: https://freehost.com.ua/faq/wiki/chto-takoe-php/ (дата обращения: 01.07.2024).
6. Java-приложения: плюсы и минусы языка [Электронный ресурс] URL: <https://gb.ru/blog/java-prilozheniya/> (дата обращения: 01.07.2024).
7. MySQL преимущества и недостатки [Электронный ресурс] URL: <https://servergate.ru/articles/mysql-preimushchestva-i-nedostatki/> (дата обращения: 01.07.2024).
8. Что такое база данных MS SQL и зачем она нужна - разработка и ведение БД в Microsoft SQL Server [Электронный ресурс] URL: https://workspace.ru/tools/database/mssql/ (дата обращения: 01.07.2024).
9. Eclipse IDE для Windows - Скачайте бесплатно с Uptodown [Электронный ресурс] URL: https://eclipse.ru.uptodown.com/windows (дата обращения: 01.07.2024).
10. Лучшие IDE для разработки на C# [Электронный ресурс] URL: https://gb.ru/blog/luchshie-ide-dlya-razrabotki-na-c/ (дата обращения: 01.07.2024).
11. Что такое Visual Studio Code и зачем он нужен - frontend и backend-разработка в VSCode [Электронный ресурс] URL: https://workspace.ru/tools/environment/visual-studio-code/ (дата обращения: 01.07.2024).
12. ТОП-10 лучших сред для разработки [Электронный ресурс] URL: https://mixtelecom.ru/blog/top-10-ide (дата обращения: 01.07.2024).
13. Интеграция программных модулей - Dynamicsun [Электронный ресурс] URL: https://dynamicsun.ru/blog/integraciya-moduley.html (дата обращения: 01.07.2024).

ПРИЛОЖЕНИЕ А

Листинг сущностей

Сущность Event:

package com.example.demo;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Event {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private String location;

private String date;

@OneToMany(mappedBy = "event", cascade = CascadeType.ALL, orphanRemoval = true)

private Set<Participant> participants;

public String getName()

{

return this.name;

}

public String getLocation()

{

return this.location;

}

public String getDate()

{

return this.date;

}

public void setName(String name)

{

this.name=name;

}

public void setLocation(String location)

{

this.location=location;

}

public void setDate(String date)

{

this.date=date;

}

public Set<Participant> getParticipants()

{

return participants;

}

}

Сущность Participant:

package com.example.demo;

import jakarta.persistence.\*;

@Entity

public class Participant {

@Id

@GeneratedValue(strategy = GenerationType.*IDENTITY*)

private Long id;

private String name;

private String email;

@ManyToOne

@JoinColumn(name = "event\_id", nullable = false)

private Event event;

public String getName()

{

return this.name;

}

public String getEmail()

{

return this.email;

}

public void setName(String name)

{

this.name=name;

}

public void setEmail(String email)

{

this.email=email;

}

public void setEvent(Event event)

{

this.event=event;

}

}

ПРИЛОЖЕНИЕ Б

Листинг репозиториев

Репозиторий для Event:

package com.example.demo;

import org.springframework.data.jpa.repository.JpaRepository;

public interface EventRepository extends JpaRepository<Event, Long> {

}

Репозиторий для Participant:

package com.example.demo;

import org.springframework.data.jpa.repository.JpaRepository;

public interface ParticipantRepository extends JpaRepository<Participant, Long> {

}

ПРИЛОЖЕНИЕ В

Листинг сервисов

Сервис для Event:

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.Optional;

@Service

public class EventService {

@Autowired

private EventRepository eventRepository;

public List<Event> getAllEvents() {

return eventRepository.findAll();

}

public Optional<Event> getEventById(Long id) {

return eventRepository.findById(id);

}

public Event createEvent(Event event) {

return eventRepository.save(event);

}

public Event updateEvent(Long id, Event eventDetails) {

Optional<Event> event = eventRepository.findById(id);

if (event.isPresent()) {

Event existingEvent = event.get();

existingEvent.setName(eventDetails.getName());

existingEvent.setLocation(eventDetails.getLocation());

existingEvent.setDate(eventDetails.getDate());

return eventRepository.save(existingEvent);

} else {

return null;

}

}

public void deleteEvent(Long id) {

eventRepository.deleteById(id);

}

}

Сервис для Participant:

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import java.util.List;

import java.util.Optional;

@Service

public class ParticipantService {

@Autowired

private ParticipantRepository participantRepository;

@Autowired

private EventRepository eventRepository;

public List<Participant> getAllParticipantsForEvent(Long eventId) {

Optional<Event> event = eventRepository.findById(eventId);

if (event.isPresent()) {

return event.get().getParticipants().stream().toList();

} else {

return List.of();

}

}

public Participant addParticipantToEvent(Long eventId, Participant participant) {

Optional<Event> event = eventRepository.findById(eventId);

if (event.isPresent()) {

participant.setEvent(event.get());

return participantRepository.save(participant);

} else {

return null;

}

}

public Participant updateParticipant(Long id, Participant participantDetails) {

Optional<Participant> participant = participantRepository.findById(id);

if (participant.isPresent()) {

Participant existingParticipant = participant.get();

existingParticipant.setName(participantDetails.getName());

existingParticipant.setEmail(participantDetails.getEmail());

return participantRepository.save(existingParticipant);

} else {

return null;

}

}

public void deleteParticipant(Long id) {

participantRepository.deleteById(id);

}

}

ПРИЛОЖЕНИЕ Г

Листинг контроллеров

Контроллер для Event:

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

import java.util.Optional;

@RestController

@RequestMapping("/api/events")

public class EventController {

@Autowired

private EventService eventService;

@GetMapping

public List<Event> getAllEvents() {

return eventService.getAllEvents();

}

@GetMapping("/{id}")

public ResponseEntity<Event> getEventById(@PathVariable Long id) {

Optional<Event> event = eventService.getEventById(id);

if (event.isPresent()) {

return ResponseEntity.*ok*(event.get());

} else {

return ResponseEntity.*notFound*().build();

}

}

@PostMapping

public Event createEvent(@RequestBody Event event) {

return eventService.createEvent(event);

}

@PutMapping("/{id}")

public ResponseEntity<Event> updateEvent(@PathVariable Long id, @RequestBody Event eventDetails) {

Event updatedEvent = eventService.updateEvent(id, eventDetails);

if (updatedEvent != null) {

return ResponseEntity.*ok*(updatedEvent);

} else {

return ResponseEntity.*notFound*().build();

}

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteEvent(@PathVariable Long id) {

eventService.deleteEvent(id);

return ResponseEntity.*noContent*().build();

}

}

Контроллер для Participant:

package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.\*;

import java.util.List;

@RestController

@RequestMapping("/api/participants")

public class ParticipantController {

@Autowired

private ParticipantService participantService;

@GetMapping("/event/{eventId}")

public List<Participant> getParticipantsForEvent(@PathVariable Long eventId) {

return participantService.getAllParticipantsForEvent(eventId);

}

@PostMapping("/event/{eventId}")

public ResponseEntity<Participant> addParticipantToEvent(@PathVariable Long eventId, @RequestBody Participant participant) {

Participant addedParticipant = participantService.addParticipantToEvent(eventId, participant);

if (addedParticipant != null) {

return ResponseEntity.*ok*(addedParticipant);

} else {

return ResponseEntity.*notFound*().build();

}

}

@PutMapping("/{id}")

public ResponseEntity<Participant> updateParticipant(@PathVariable Long id, @RequestBody Participant participantDetails) {

Participant updatedParticipant = participantService.updateParticipant(id, participantDetails);

if (updatedParticipant != null) {

return ResponseEntity.*ok*(updatedParticipant);

} else {

return ResponseEntity.*notFound*().build();

}

}

@DeleteMapping("/{id}")

public ResponseEntity<Void> deleteParticipant(@PathVariable Long id) {

participantService.deleteParticipant(id);

return ResponseEntity.*noContent*().build();

}

}

ОТЗЫВ

О ПРОХОЖДЕНИИ ПРАКТИКИ

Вид практики производственная

(учебная, производственная)

Тип практики технологическая (проектно-технологическая) практика

(в соответствии с ОПОП ВО)

Сроки прохождения практики: с 01.07.2024 по 13.07.2024

(в соответствии с календарным учебным графиком)

по направлению подготовки 09.03.01 Информатика и вычислительная техника

(уровень бакалавриата)

направленность (профиль) «Информационные системы»

обучающимся группы № 6303-090301D Лутониным Александром Сергеевичем

*(фамилия, имя, отчество)*

|  |  |  |
| --- | --- | --- |
| № п/п | Критерии оценки | Оценка  (по 5-балльной шкале) |
|  | Общая систематичность и ответственность работы в ходе практики | 5 |
|  | Достижение планируемых результатов практики | 5 |
|  | Корректность в сборе, анализе и интерпретации представляемых данных | 5 |
|  | Степень личного участия и самостоятельности практиканта в представляемом отчета о практике | 5 |
|  | Качество оформления отчетной документации | 5 |
|  | **ИТОГОВАЯ ОЦЕНКА** (выставляется как среднее арифметическое оценок по пяти критериям оценки) | 5 |

Руководитель практики

от профильной организации,

Ведущий разработчик ПО, к.ф.-м.н. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Цветков

*(подпись, печать)*