**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ПП.09.02.07-1.24.202.07

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПМ.02 Осуществление интеграции программных модулей

|  |  |  |
| --- | --- | --- |
| Руководитель от предприятия:  М.П. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.А. Кличенко) |
| Руководитель от техникума: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (С.Н. Касьяненко) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (А.С. Замарацкий) |

Иркутск 2024

Содержание

[ВВЕДЕНИЕ 3](#_Toc163266856)

[1. Создание проекта на GitHub 4](#_Toc163266857)

[2.Техническое задание на разработку БД 5](#_Toc163266858)

[2.1 Загрузка файла технического задания 6](#_Toc163266859)

[3. Модуль приема данных извне MySql 8](#_Toc163266860)

[4. Выполнение отладки 10](#_Toc163266861)

[5. Тестовые наборы данных 11](#_Toc163266862)

[6. Настройка JSHint 13](#_Toc163266863)

[ЗАКЛЮЧЕНИЕ 15](#_Toc163266864)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 16](#_Toc163266865)

# ВВЕДЕНИЕ

Цель производственной практики в сфере информационных технологий и программирования – применить полученные в ходе обучения теоретические знания на практике, а также лучше понять будущую работу и свои профессиональные обязанности.

Цели практики:

* расширение и систематизация знаний на основе изучений деятельности;
* приобретение практических навыков в сфере информационных технологий;
* развитие навыков работы в команде и коммуникации с коллегами;
* укрепление теоретических знаний, полученный во время обучения.

Задачи практики:

* создание технического задания на разработку базы данных в соответствии с ГОСТ;
* создание модуля приёма данных извне;
* подготовка тестовых наборов данных для запросов GET и POST с помощью Postman.

# Создание проекта на GitHub

Пользовательские проекты могут отслеживать проблемы и получать запросы из репозиториев, принадлежащих вашей личной учетной записи.

В правом верхнем углу GitHub.com необходимо нажать на фотографию своего профиля, затем перейти в «Ваш профиль».



Рисунок 1 – Ваш профиль

В своем профиле перейти в «Проекты» как показано на рисунке 2.

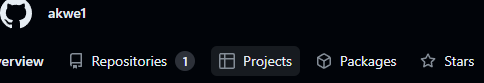


Рисунок 2 – Проекты

Нажать «Новый проект».

Выбрать тип проекта или шаблона, который необходимо использовать.

* Чтобы создать пустой проект, в разделе «Начать с нуля» нужно нажать «Таблица», «Дорожная карта» или «Доска».
* Чтобы создать проект на основе шаблона, нужно перейти к встроенному шаблону, который нужно использовать.

В текстовом поле «Имя проекта» нужно ввести имя нового проекта.

Затем нажать «Создать проект».

# Техническое задание на разработку БД

**1 Наименование проекта и цель создания базы данных**

Наименование проекта: «Киноклуб».

Цель создания: облегчение поиска необходимого контента для пользователей.

Исполнитель: студент Иркутского Авиационного Техникума, группы БД-20-2, Замарацкий Алексей Саидович.

Разработка базы данных проходит в рамках производственной практики для ГБПОУИО «ИАТ».

Сроки разработки базы данных с 14.03.2024 по 04.04.2024 года.

**2 Общие требования к базе данных**

В базе данных необходимо хранить информацию, содержащую описание всего контента кино, также данные пользователей (логин, пароль и т.д.).

Также необходимо присутствие возможности поиска информации в базе данных способом фильтрации по атрибуту жанр.

База данных должна поддерживать целостность данных и обеспечивать безопасность доступа.

В базе данных должна присутствовать возможность добавления нового контента и пользователей. Так же необходима возможность взаимодействия с существующими.

**3 Описание структуры данных**

База данных «Киноклуб» может состоять из следующих таблиц:

Пользователь. (Идентификатор, Фамилия, Имя, Отчество, Пароль, Адрес, Мобильный телефон, Логин, Идентификатор роли).

Роль. (Идентификатор, Название).

Рейтинг\_Комментарии. (Идентификатор, Идентификатор пользователя, Идентификатор контента, оценка рейтинга, комментарий).

Контент. (Идентификатор, Название, Эпизоды, Статус, Идентификатор жанра, Дата выпуска, Возрастное ограничение, Длительность, Идентификатор режиссёра, Картинка).

То есть таблицы: «users», «roles», «ratingcomments», «content» которые имеют связь по внешнему ключу между собой.

**4 Требования к целостности данных**

Требования к целостности данных таковы:

Уникальность данных: каждая запись в базе данных должна быть уникальной, чтобы избежать дублирования информации.

Согласованность данных: данные в базе данных должны быть согласованными и точными, чтобы избежать противоречий и ошибок.

Ограничения целостности: использование ограничений целостности, таких как уникальные ключи, внешние ключи и проверочные ограничения, для обеспечения правильности данных.

Актуальность данных: данные должны быть актуальными и обновленными, чтобы предоставлять точную информацию пользователям.

Безопасность данных: обеспечение защиты данных от несанкционированного доступа, изменений и удалений.

Резервное копирование данных: регулярное создание резервных копий данных для предотвращения потери информации в случае сбоев или аварий.

**5 Требования к документации**

Основным документам, регламентирующими использование базы данных является руководство пользователя.

Основным документам, регламентирующими разработку базы данных является техническое задание.

# Загрузка файла технического задания

Для того, чтобы загрузить техническое задание на GitHub необходимо зайти в ранее созданный проект. После нажать на кнопку «загрузки существующего файла» как показано на рисунке 3.

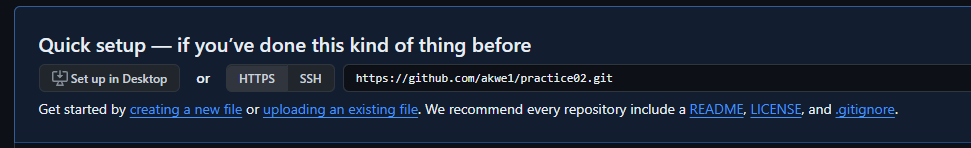


Рисунок 3 – Загрузка файла

Далее нужно поместить файл технического задания (рисунок 4).

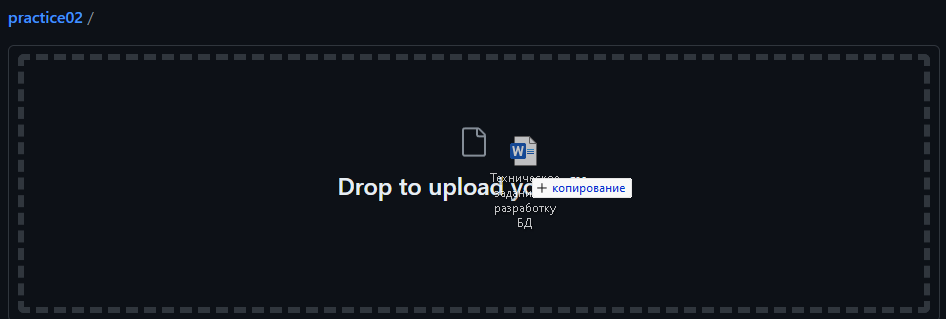


Рисунок 4 – Место для загрузки файла

Нажимаем на кнопку «Commit changes» и готово (рисунок 5).

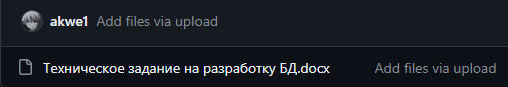


Рисунок 5 – Загруженный файл

GitHub – https://github.com/akwe1/practice02

# 3. Модуль приема данных извне MySql

Для написания модуля приёма данных извне, проверки на корректность последующей записи в базу данных необходимо обратиться к JavaScript и NodeJS. На следующем рисунке показано подключение к базе данных.

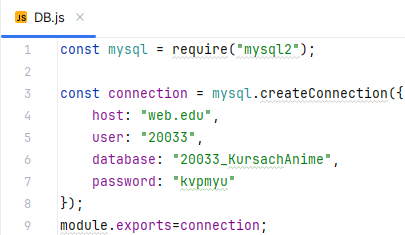


Рисунок 6 – Подключение к базе данных

Ниже на рисунке 7 представлен код файла навигации в проекте, с использованием Get и Post.



Рисунок 7 – Файл навигации

На следующем рисунке показан код функции регистрации, с использованием запроса на добавление пользователя с хэшированием пароля.

Рисунок 8 – Функция регистрации пользователя

Ниже представлен рисунок кода регистрации, с использованием валидации данных. В данном случае показана валидация пароля, с условием того, чтобы пароль содержал более 6 символов.



Рисунок 9 – Регистрация с валидацией

GitHub – https://github.com/akwe1/practice02

# 4. Выполнение отладки

Отладка программного кода – это процесс поиска и устранения ошибок в программе. Ошибки могут быть связаны с неправильным функционированием программы, неправильным вводом данных, неправильным выводом данных и т.д. Отладка может быть выполнена с помощью различных инструментов и методов, таких как использование отладчиков, тестирование программы, использование автоматических инструментов для поиска ошибок и т.д.

Этот фрагмент кода содержит логику обработки ошибок и возвращения результатов асинхронной операции.

В начале кода проверяется наличие ошибки (если она произошла). Если ошибка присутствует, то она передается в reject (). Если ошибки не произошло, то код продолжает выполнение и возвращается результат операции – элементы.

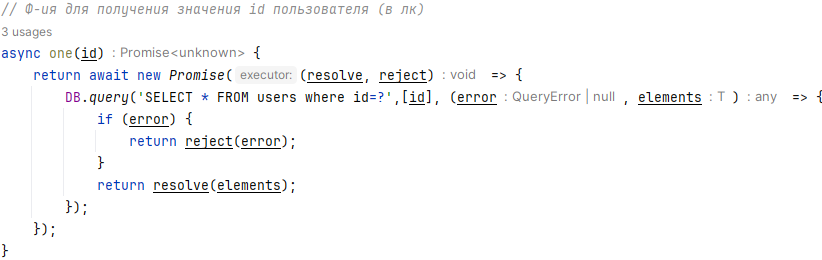


Рисунок 10 – Функция получения значения id user

Отладка имеет критическую значимость в процессе разработки программного обеспечения и способствует созданию надежных и качественных программ. Для успешной отладки необходимо применять тщательный, внимательный и систематический подход с целью обнаружения и устранения ошибок в коде.

# 5. Тестовые наборы данных

Postman – это инструмент для тестирования API, который позволяет разработчикам и тестировщикам создавать, управлять, отправлять и получать HTTP-запросы.

Postman позволяет создавать и отправлять запросы различных типов (GET, POST, PUT, DELETE и пр.) к API, устанавливать параметры запроса, заголовки и тело запроса, и многое другое. Также программа дает возможность сохранять запросы, формировать коллекции запросов, автоматизировать тестирование работы API и разрабатывать документацию.

Метод GET – это один из способов отправки запросов в протоколе HTTP, предназначенный для получения данных с сервера. При использовании GET-запроса состояние и данные на сервере не изменяются, так как клиент просто запрашивает информацию с сервера. На снимке 11 показан результат выполнения GET-запроса в формате JSON

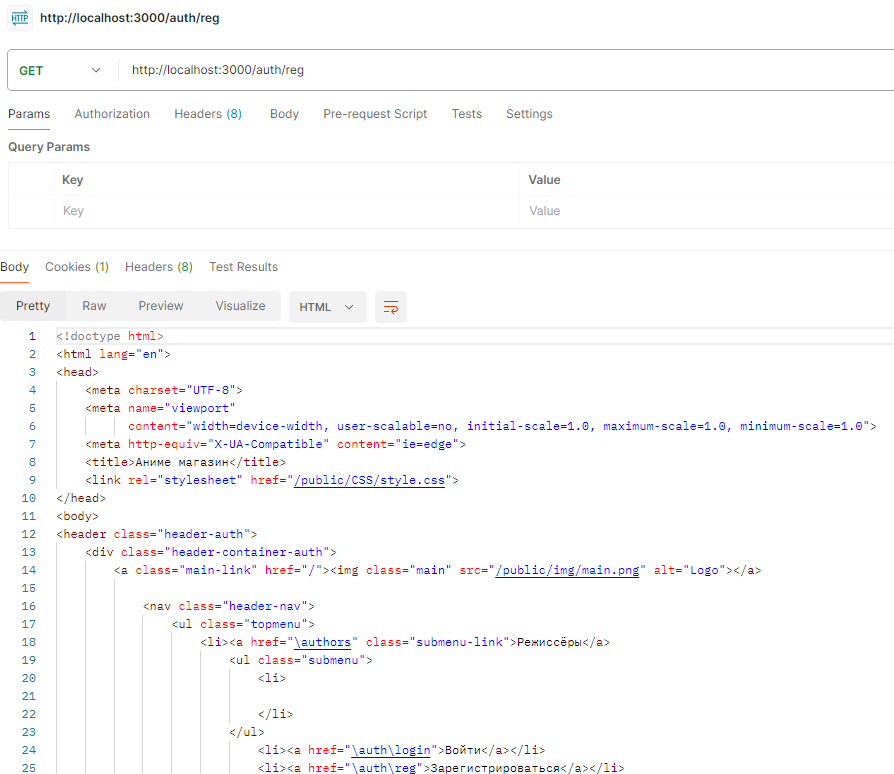


Рисунок 11 – GET запрос

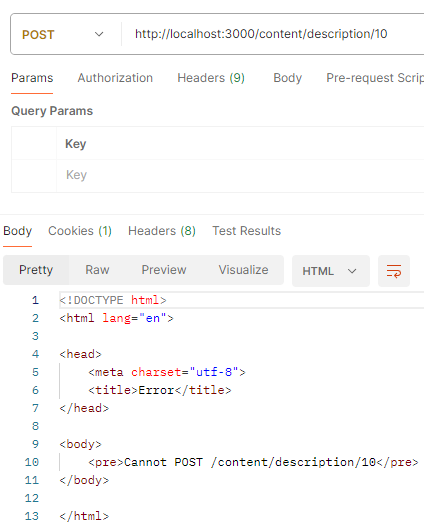


Рисунок 12 – POST запрос

Для передачи конфиденциальных данных рекомендуется использовать метод POST (рисунок 12).

GitHub – https://github.com/akwe1/practice02

# 6. Настройка JSHint

JSHint – это инструмент для проверки и оптимизации кода JavaScript. Он помогает находить ошибки, предупреждать о проблемах и предлагает способы улучшения кода. JSHint может использоваться как в процессе написания кода, так и после его создания для улучшения качества и производительности программы.

JSHint выполняет следующие функции:

* Обнаружение ошибок: JSHint помогает находить различные типы ошибок в коде JavaScript, такие как отсутствующие точки с запятой, неправильные кавычки и т. д.
* Предупреждения: JSHint также предупреждает о потенциальных проблемах в коде, таких как использование устаревших конструкций или нерекомендуемых методов.
* Рекомендации по улучшению кода: JSHint предлагает способы оптимизации и улучшения кода, например, путем сокращения избыточного кода или использования более современных конструкций.
* Поддержка стандартов кодирования: JSHint позволяет настроить код таким образом, чтобы он соответствовал определенным стандартам кодирования, таким как JavaScript Standard Style (JSS).
* Интеграция с инструментами разработки: JSHint легко интегрируется с различными инструментами разработки, такими как Visual Studio Code, WebStorm и другими, что позволяет использовать его возможности прямо во время написания кода.

Установка JSHint глобально, выполняется, используя npm:

npm install -g jshint jshint-stylish

Для запуска необходимо создать файл с Java Script кодом. После открыть терминал или командную строку и перейти в папку, где находится созданный файл. Далее ввести команду jshint имя файла для проверки кода.

Уровни JSHint определяют степень строгости проверки кода. Есть пять уровней:

* Мягкий (Soft): Этот уровень наименее строгий и позволяет некоторые отклонения от стандартов кодирования.
* Разумный (Reasonable): Этот уровень является более строгим, но все еще допускает некоторые отклонения.
* Строгий (Strict): Этот уровень очень строгий и не допускает никаких отклонений от стандартов кодирования.
* Идеал (Ideal): Этот уровень еще более строгий, чем строгий, и требует, чтобы код соответствовал стандартам кодирования без каких-либо отклонений.
* Ломает (Lax): Этот уровень самый слабый и позволяет много отклонений от стандартов кодирования.

# ЗАКЛЮЧЕНИЕ

В результате практики на тему «Осуществление программных интеграций», были изучены базовые принципы и методики интеграции различных программных платформ. Обретенные навыки включают работу с API, обменными протоколами данных, форматами данных и инструментарием для интеграции.

Практика предоставляет множество возможностей для получения новых знаний и навыков, а также для развития профессиональных компетенций. Были закреплены теоретические знания, получены новые профессиональный навыки и умения. Также, были выполнены поставленные задачи и достигнуты цели, такие как:

* Создание проекта на GitHub.
* Составление технического задания на разработку базы данных в соответствии с ГОСТ.
* Написание модуля приема данных извне.
* Выполнение отладки любым из доступных инструментов.
* Подготовка тестовых наборов данных для запросов GET и POST с помощью Postman.
* Настройка JSHint.

Мне удалось успешно освоить и применить новые умения и навыки в ходе практики, что позволило мне лучше понять специфику работы и получить ценный опыт для дальнейшего развития в выбранной области.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. arcgis.com – Отладка инструментов-скриптов – URL: https://desktop.arcgis.com/ru/arcmap/latest/analyze/creating-tools/debugging-script-tools.htm (Дата обращения: 02.04.2024) Тест: электронный
2. docs.github.com – Создание проекта – URL: https://docs.github.com/ru/issues/planning-and-tracking-withprojects/creatingprojects/creating-a-project (Дата обращения: 02.04.2024) Тест: электронный
3. habr.com – Разработка технического задания для БД по ГОСТ – URL: https://habr.com/ru/articles/432852/ (Дата обращения: 02.04.2024) Тест: электронный
4. habr.com – Синтаксис систем модулей – URL: https://habr.com/ru/companies/domclick/articles/532084/ (Дата обращения: 02.04.2024) Тест: электронный
5. myrusakov.ru – Как установить и настроить JSHint – URL: https://myrusakov.ru/js-jshint.html (Дата обращения: 02.04.2024) Тест: электронный
6. testengineer.ru – Гайд по тестированию с Postman – URL: https://testengineer.ru/gajd-po-testirovaniyu-v-postman/ (Дата обращения: 02.04.2024) Тест: электронный