МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИТ)
Кафедра инструментального и прикладного программного обеспечения (ИиППО)

ЗАДАНИЕ
на выполнение курсовой работы

по дисциплине: Разработка серверных частей интернет-ресурсов
по профилю: Разработка и дизайн компьютерных игр и мультимедийных приложений
направления профессиональной подготовки: Программная инженерия (09.03.04)

Студент: Боциева Элина Игоревна
Группа: ИКБО-03-20
Срок представления к защите: 01.12.2022
Руководитель: Стариковская Надежда Анатольевна, к.т.н., доцент

Тема: Серверная часть веб-приложения «Чат»

Исходные данные: используемые технологии: язык программирования PHP, среда разработки Jetbrains PHPStorm, СУБД MySQL, использование паттерна проектирования MVC. Нормативный документ: инструкция по организации и проведению курсового проектирования СМКО МИРЭА 7.5.1/04.И.05-18.

Перечень вопросов, подлежащих разработке, и обязательного графического материала: 1. Провести анализ предметной области разрабатываемого веб-приложения. 2. Обосновать выбор технологий разработки веб-приложения. 3. Разработать архитектуру веб-приложения на основе выбранного паттерна проектирования. 4. Реализовать слой серверной логики веб-приложения с применением выбранной технологии. 5. Реализовать слой логики базы данных. 6. Разработать слой клиентского представления веб-приложения 7. Создать презентацию по выполненной курсовой работе.

Руководителем произведён инструктаж по технике безопасности, противопожарной технике и правилам внутреннего распорядка.

Зав. кафедрой ИиППО: _____/Р. Г. Болбаков/, «16» сентября 2022 г.

Задание на КР выдал: _____/Н. А. Стариковская/, «16» сентября 2022 г.

Задание на КР получил: _____/Э. И. Боциева/, «16» сентября 2022 г.

УДК 004.414.2

Боциева Э.И. Курсовая работа на тему Серверная часть веб-приложения «Чат» / **Отчет по курсовой работе бакалавриата** в пятом семестре третьего курса обучения профиля «Инструментальное и прикладное программное обеспечение» направления профессиональной подготовки бакалавриата 09.03.04. «Программная инженерия» / руководитель к.т.н., доцент Стариковская Н.А. / кафедра ИиППО Института ИТ РТУ МИРЭА — 26 страниц, 15 таблиц, 24 иллюстрации, 14 информационных источников.

АННОТАЦИЯ

Целью данной работы является закрепление практических навыков по созданию серверных частей интернет-ресурсов по теме «Чат» с применением современных технологий согласно заданию.

В разделе «Введение» обосновывается актуальность выбранной темы, ставится цель.

Основная часть содержит материал, необходимый для достижения цели курсовой работы.

В разделе «Глава 1» производится анализ предметной области, начальное планирование функционала приложения, проводится и обосновывается выбор технологий и паттернов проектирования.

В разделе «Глава 2» описывается архитектура приложения.

В разделе «Глава 3» описывается разработка и тестирование требуемого приложения.

В разделе «ЗАКЛЮЧЕНИЕ» излагаются теоретические выводы, сформулированные после разработки приложения.

ГЛОССАРИЙ

API – Application Programming Interface, Прикладной Интерфейс Программирования – описание способов взаимодействия одной компьютерной программы с другими.

MVC – Model View Controller, Модель Представление, Котроллер – схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента может осуществляться независимо

HTTP – Hyper Text Transfer Protocol, Протокол Передачи Гипертекста - протокол прикладного уровня передачи данных, в настоящее время используется для передачи произвольных данных

REST – Representational State Transfer, Передача Репрезентативного Состояния – архитектурный стиль взаимодействия компонентов распределённого приложения в сети.

SQL – Structured Query Language, Структурируемый Язык Запросов – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных

СУБД – Система Управления Базой Данных – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В среднем за день в интернет входит 71,1% населения России [1]. Сегодня общение в интернете имеет большое значение: ежедневно пользователи обмениваются большим количеством корпоративной, учебной и развлекательной информации.

До появления форумов, социальных сетей и мессенджеров люди общались в чатах на своих персональных веб-сайтах, либо по специальным каналам, отведенным для общения.

Однако чаты до сих пор не утратили свою актуальности и нашли большое количество прикладных применений. Так на различных веб-ресурсах можно встретить чаты с технической поддержкой или консультацией [2], есть чаты, предназначенные для экстренной поддержки в кризисных ситуациях; на стриминговых сервисах присутствуют чаты для общения со зрителями. Но также остались веб-ресурсы, полностью посвященные общению, которые посещают регулярно.

Таким образом, создание чата является перспективной областью для разработки приложения. Данная курсовая работа фокусируется на проектировании и разработке серверной части веб-приложения «Чат».

1 ГЛАВА 1

1.1 Анализ предметной области

Для определения необходимых функций системы был проведен анализ веб-ресурсов и приложений, позволяющих общаться с помощью чата.

«Shoutbox» - пример «классического» чата для персональных веб-сайтов [3]. Данный веб-ресурс является демонстрацией приложения, однако можно выделить, что в данном чате мало функций: можно использовать любой никнейм и писать только в одну комнату (рисунок 1).

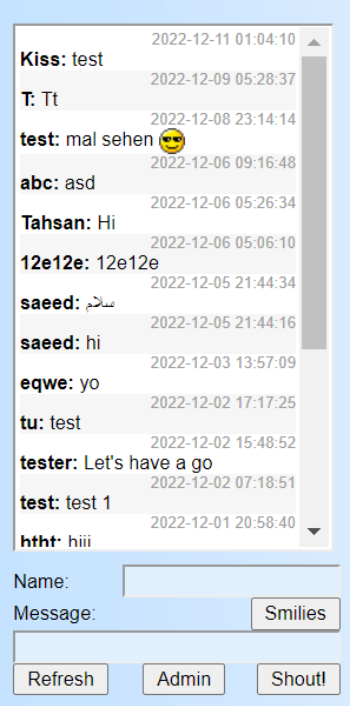

Рисунок 1 - Чат «Shoutbox»

Интернет-ресурс «omegle» [4] позволяет общаться со случайными собеседниками. Пользователь не регистрируется, ему предлагается вписать интересы и выбрать, в каком формате общаться: текстовом или с веб-камерой (рисунок 2). Был рассмотрен вариант общения только в текстовом формате. На странице чата пользователь может отправлять сообщения и переключаться на следующего собеседника (рисунок 3).

В силу того, что на сайте отсутствует активная модерация, в чат часто пишут боты, рассылающие спам.

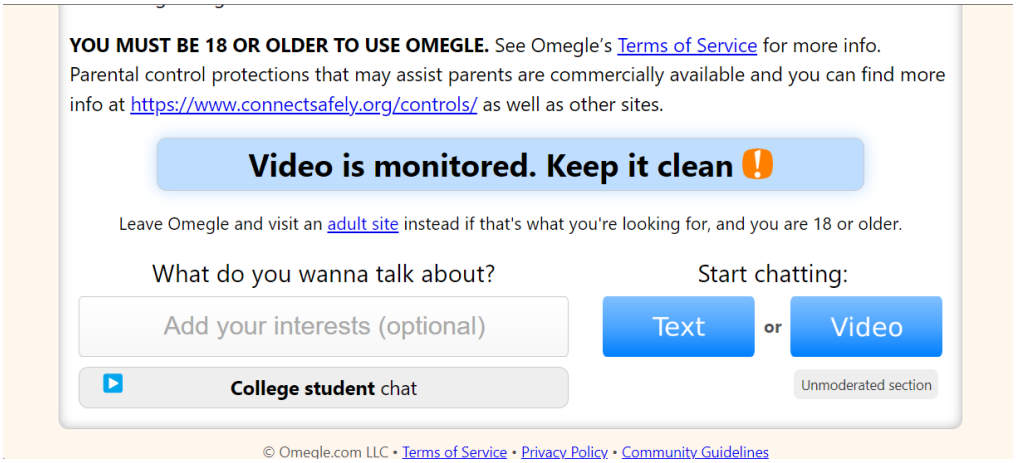

Рисунок 2 – Главная страница веб-ресурса «omegle»

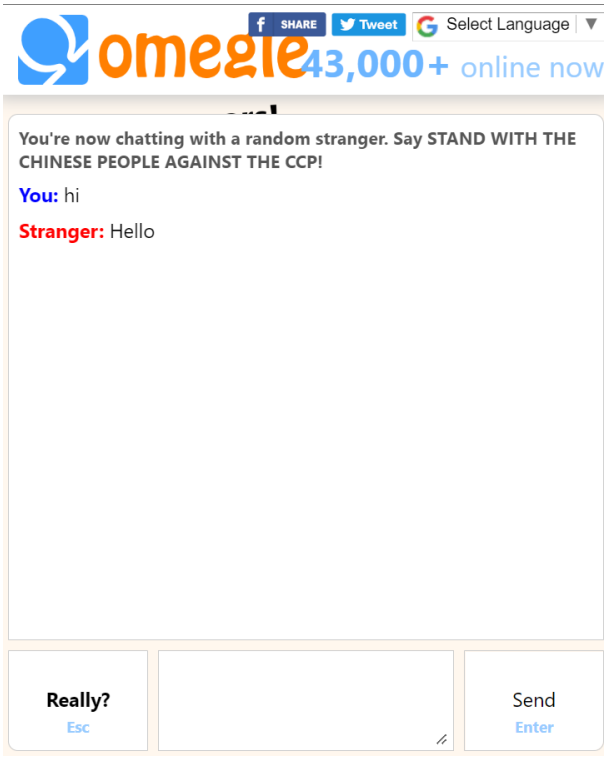

Рисунок 3 – Чат в «omegle»

Веб-ресурс «Chatzy» [5] позволяет пользователям создавать комнаты и приглашать в них других пользователей по специальной ссылке (рисунок 4). Создатели комнат могут изменять информацию о них позже. Кроме этого, они могут осуществлять модераторскую деятельность и удалять сообщения пользователей, либо самих пользователей из чата. Получить доступ к чату можно только по ссылке, нельзя просматривать список существующих чатов.

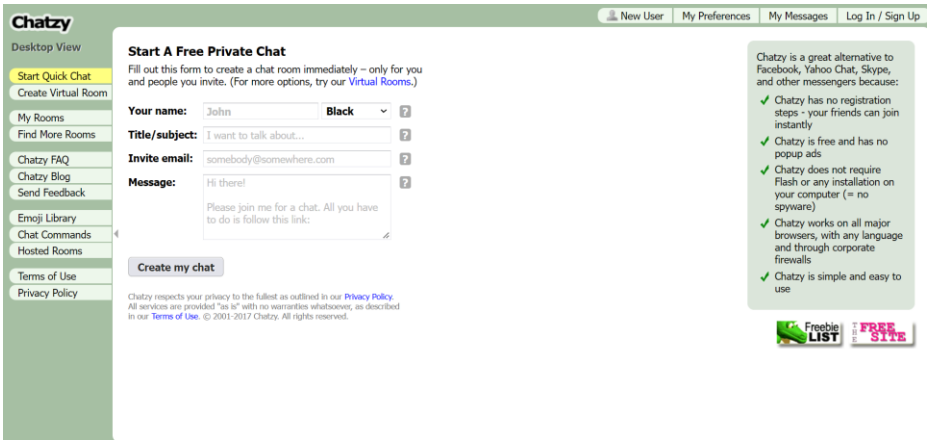

Рисунок 4 - Главная страница веб-ресурса «Chatzy»

Перечень функций для автоматизации и реализации

Проанализировав аналогичные веб-ресурсы, можно выделить ряд ключевых функций приложения, а также роли пользователей. Чат является местом для коммуникации, поэтому любой пользователь может отправлять и просматривать сообщения внутри чата под уникальным никнеймом для визуальной идентификации. Кроме основного чата, пользователи могут находиться в тематических чатах, называемых комнатами, которые тоже имеют уникальные имена. Все комнаты можно просматривать в каталоге, где указываются название и краткое описание комнаты. Пользователи могут вступать в любые комнаты.

Создатель может изменить название и описание комнаты, либо удалить её. Для обеспечения комфортного общения внутри тематического чата создатель может удалять сообщения пользователей или удалять самих пользователей из комнаты.

Таким образом, в данной системе можно выделить две роли пользователей – обычный пользователь, который может менять информацию о себе, отправлять сообщения, создавать комнаты, просматривать каталоги комнат и вступать в них, и создатель комнаты, который может менять информацию о комнате, удалять её и осуществлять деятельность модератора чата (таблица 1).

Таблица 1 – Роли пользователей и доступные действия

Роль пользователя	**Доступные действия**
Любой пользователь	Создание учетной записи
	Удаление учетной записи
	Изменение псевдонима
	Отправка сообщений
	Удаление своих сообщений
	Просмотр каталога комнат
	Вступление в комнату
	Создание комнаты
Создатель комнаты	Изменение информации о комнате
	Удаление сообщений пользователей
	Удаление пользователей из комнаты
	Удаление комнаты

1.2 Выбор и обоснование технологий

Подбор технологий следует проводить из расчета на выбранную архитектуру. При этом следует выбирать технологии, зарекомендовавшие себя в течение времени.

В качестве архитектуры выбран паттерн проектирования MVC (Model-View-Controller).

Для разрабатываемого приложения важна правильная архитектура серверной части, поэтому в качестве клиентской части было использовано ПО для тестирования API Postman.

Для серверной части используется язык программирования PHP, являющийся распространенным языком программирования общего назначения с открытым исходным кодом. Также язык PHP обладает рядом преимуществ: высокой скоростью работы и производительностью ресурсов и совместимость с разными платформами.

Для хранения данных используется открытая реляционная система управления базами данных MySQL, обеспечивающая высокую скорость

работы, безопасность, масштабируемость, высокую производительность и отказоустойчивость.

Также для создания окружения LAMP используется технология Docker, позволяющая развертывать виртуальное окружение в любой системе. Для разработки и отладки кода используется IDE PhpStorm, в качестве системы контроля версий используется GitHub.

1.3 ТТЗ

Для того чтобы правильно описать функционал приложения и доступные технологии необходимо представить тактико-техническое задание.

Согласно функциональным процессам и стеку доступных технологий получено следующее задание.

Список требований:

- Функциональная реализация
    - Для всех пользователей:
        - Авторизация и регистрация
        - Изменение информации о себе
        - Отправка сообщений
        - Удаление своих сообщений
        - Просмотр каталога комнат
        - Вступление в комнату
        - Создание комнаты
    - Для создателя чата:
        - Изменение информации о комнате
        - Удаление комнаты
        - Удаление сообщений
        - Удаление пользователей

Для обеспечение качественной работы приложения, необходимо выполнить разработку, используя современные инструменты, которые позволят оптимально составить структуру приложения. Были использованы следующие инструменты:

- PHP
- MySQL
- PhpStorm
- Postman

В современном мире необходимо ориентироваться на множество платформ (платформы компьютера, мобильные платформы и т.д.), поэтому к необходимым платформам можно отнести:

- IOS
- WEB
- Linux
- Android
- Windows

2 ГЛАВА 2

2.1 Описание общей архитектуры приложения

В основе разрабатываемого продукта используется паттерн проектирования MVC (Model-View-Controller). Так как разрабатывается серверная часть приложения, то будут реализованы два слоя: модель и контроллеры. В модели будет содержаться база данных, в то время как контроллеры будут осуществлять взаимодействие между ней и пользователем.

2.2 Структура данных в СУБД

База данных состоит из пяти таблиц, в которых содержатся необходимые для работы приложения данные об объектах. Так в таблице «person» пользователей содержатся идентификационный номер пользователя, его псевдоним и адрес электронной почты. В таблице «room» содержатся идентификационный номер комнаты, идентификационный номер создателя комнаты (из таблицы «person»), название и описание комнаты. В таблице «person_room» содержатся данные о том, в каких комнатах находятся пользователи: используются идентификационные номера пользователя и комнаты, а также уникальные номер записи в таблице. В таблице «message» содержатся данные о сообщениях: идентификационные номера отправителя и комнаты, в которую было отправлено сообщение, уникальные номер сообщения и его содержимое (рисунок 5).

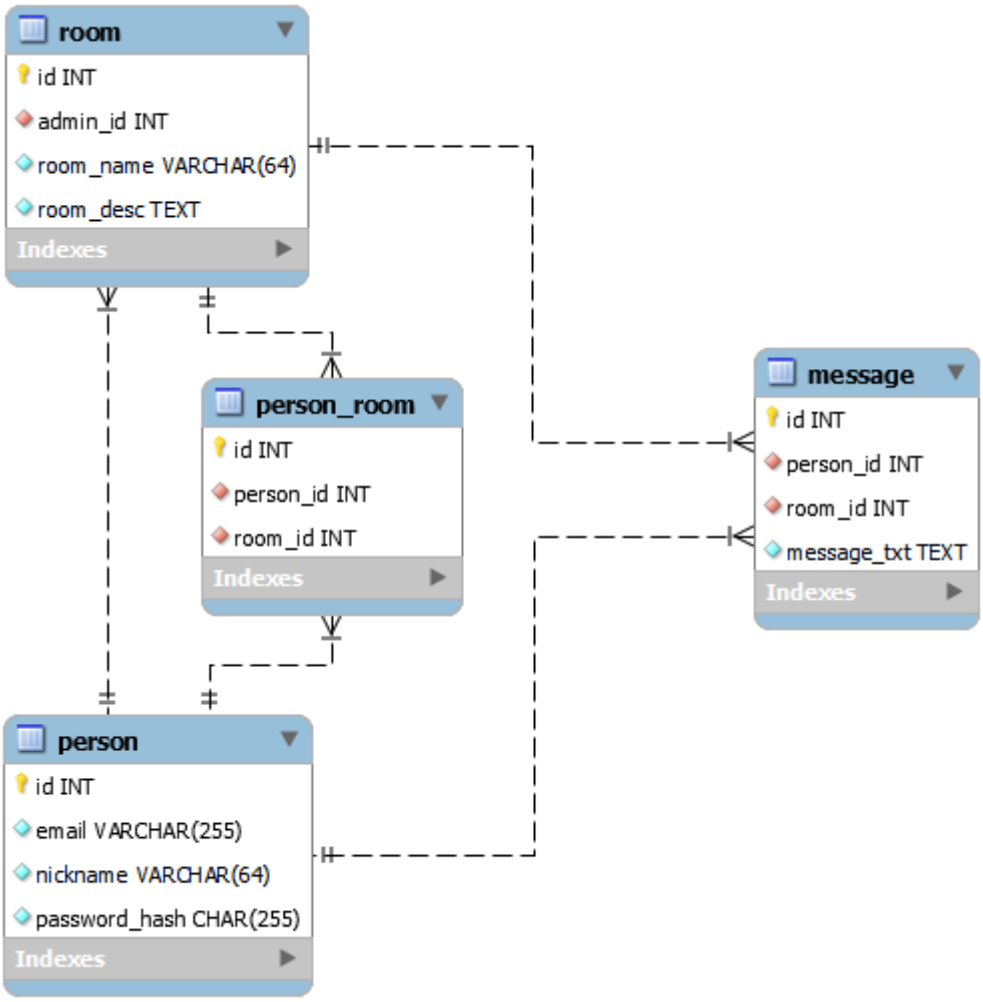

Рисунок 5 - Структура таблиц в базе данных

2.3 Протокол взаимодействия с серверной частью

Взаимодействие между слоями приложения основано на протоколе HTTP и архитектуре REST. Клиентская часть получает информацию с сервера или отправляет запросы на сервер. Серверная часть предоставляет HTTP адреса, по которым ожидаются запросы. Данное разделение обеспечивает независимость серверной части приложения от клиентской (рисунок 6).

В данном случае обмен данными и коммуникация с сервером будет осуществляться с помощью программного обеспечения для тестирования API Postman. Используются запросы GET для представления извлеченных данных, POST для добавления новых данных, PATCH для частичного изменения данных и DELETE для их удаления.

Для обеспечения защиты информации подразумевается использование файлов cookie, хранящих в клиентской части приложения информацию о сессии пользователя, на основании которой определяются пользовательские привилегии. Для избежания атак, связанных с отправкой запросов к базе данных, используются рекомендации по обработке SQL-запросов при написании кода.

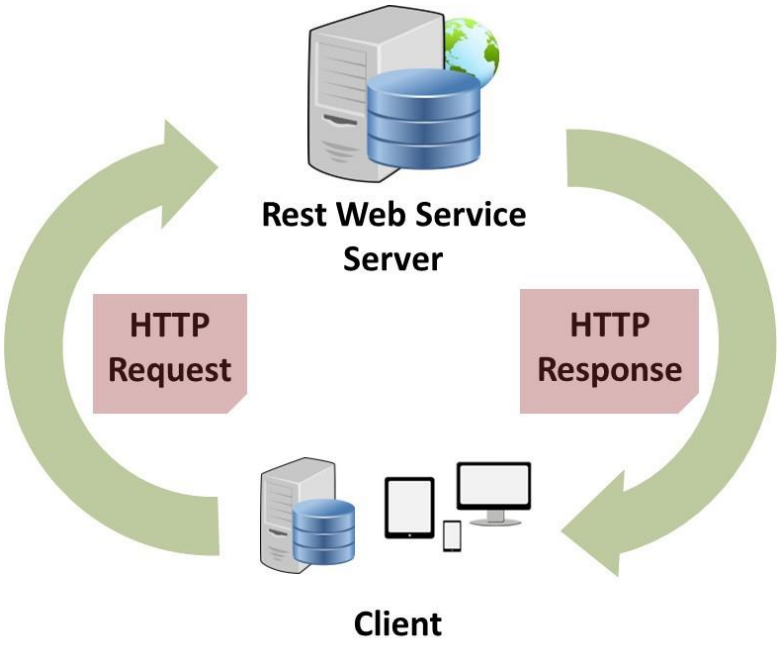

Рисунок 6 – Схема взаимодействия по протоколу HTTP

Для каждой сущности были определены соответствующие ей методы, позволяющие осуществлять полностью или частично операции группы CRUD: создание, чтение, обновление и удаление.

Так для сущности «Person» пользователя реализованы методы, позволяющие создать учетную запись, обновить псевдоним пользователя, удалить учетную запись и получить список пользователей. Подробное описание методов см. в таблице 2 – 5.

Таблица 2 – Метод создания учетной записи пользователя

Метод	POST
Адрес запроса	/api/person/create.php
Входные данные	nickname: string (псевдоним пользователя), email: string (электронная почта пользователя), password-hash: string (пароль пользователя).
Выходные данные	Status 200: Оповещение об успешной регистрации. Status 400: Оповещение об ошибке.

Таблица 3 – Метод изменения псевдонима пользователя

Метод	PATCH
Адрес запроса	/api/person/update.php
Входные данные	nickname: string (новый псевдоним пользователя), id: int (идентификационный номер пользователя).
Выходные данные	Status 200: Оповещение об успешном изменении. Status 400: Оповещение об ошибке.

Таблица 4 – Метод изменения псевдонима пользователя

Метод	GET
Адрес запроса	/api/person/read.php
Входные данные	Отсутствуют.
Выходные данные	Status 200: Массив объектов вида {nickname: string (псевдоним пользователя)}. Status 400: Оповещение об ошибке.

Таблица 5 – Метод удаления учетной записи пользователя

Метод	DELETE
Адрес запроса	/api/person/delete.php
Входные данные	id: int (идентификационный номер пользователя)
Выходные данные	Status 200: Оповещение об успешном удалении. Status 400: Оповещение об ошибке.

Для сущности «Message» сообщения реализованы методы, позволяющие отправить сообщение в комнату, удалить сообщение и посмотреть все сообщения в комнате. Подробное описание методов см. в таблице 6 – 8.

Таблица 6 – Метод отправки сообщения

Метод	POST
Адрес запроса	/api/message/create.php
Входные данные	person_id: int (идентификационный номер создателя), room_id: string (идентификационный номер комнаты), message_txt: string (содержимое сообщения).
Выходные данные	Status 200: Оповещение об успешной отправке. Status 400: Оповещение об ошибке.

Таблица 7 – Метод получения списка сообщений

Метод	GET
Адрес запроса	/api/message/read.php
Входные данные	room_id: int (идентификационный номер комнаты).
Выходные данные	Status 200: Массив объектов вида {nickname: string (псевдоним отправителя), message_txt: string (содержимое сообщения)}. Status 400: Оповещение об ошибке.

Таблица 8 – Метод удаления сообщения

Метод	DELETE
Адрес запроса	/api/message/delete.php
Входные данные	id: int (идентификационный номер сообщения)
Выходные данные	Status 200: Оповещение об успешном удалении. Status 400: Оповещение об ошибке.

Для сущности «Person_room» таблицы участников комнат реализованы методы, позволяющие добавить пользователя в комнату, удалить пользователя из неё и посмотреть список всех пользователей в комнате. Подробное описание методов см. в таблице 9 – 11.

Таблица 9 – Метод получения списка пользователей

Метод	GET
Адрес запроса	/api/person_room/read.php
Входные данные	room_id: int (идентификационный номер комнаты).
Выходные данные	Status 200: Массив объектов вида {nickname: string (псевдоним пользователя)}. Status 400: Оповещение об ошибке.

Таблица 10 – Метод добавления пользователя в комнату

Метод	POST
Адрес запроса	/api/person_room/create.php
Входные данные	person_id: int (идентификационный номер человека), room_id: int (идентификационный номер комнаты).
Выходные данные	Status 200: Оповещение об успешном добавлении. Status 400: Оповещение об ошибке.

Таблица 11 – Метод удаления пользователя из комнаты

Метод	DELETE
Адрес запроса	/api/person_room/delete.php
Входные данные	person_id: int (идентификационный номер человека), room_id: int (идентификационный номер комнаты).
Выходные данные	Status 200: Оповещение об успешном удалении. Status 400: Оповещение об ошибке.

Для сущности «Room» комнаты реализованы методы, позволяющие создать комнату, обновить информацию о комнате, удалить комнату и получить список комнат. Подробное описание методов см. в таблице 12 – 15.

Таблица 12 – Метод создания учетной записи пользователя

Метод	POST
Адрес запроса	/api/room/create.php
Входные данные	admin_id: int (идентификационный номер создателя), room_name: string (название комнаты), room_desc: string (описание комнаты).
Выходные данные	Status 200: Оповещение об успешном создании. Status 400: Оповещение об ошибке.

Таблица 13 – Метод изменения псевдонима пользователя

Метод	PATCH
Адрес запроса	/api/room/update.php
Входные данные	room_name: string (новое название комнаты), room_desc: string (новое описание комнаты), id: int (идентификационный номер комнаты).
Выходные данные	Status 200: Оповещение об успешном изменении. Status 400: Оповещение об ошибке.

Таблица 14 – Метод изменения псевдонима пользователя

Метод	GET
Адрес запроса	/api/room/read.php
Входные данные	Отсутствуют.
Выходные данные	Status 200: Массив объектов вида {room_name: string (название комнаты), room_desc: string (описание комнаты)}. Status 400: Оповещение об ошибке.

Таблица 15 – Метод удаления учетной записи пользователя

Метод	DELETE
Адрес запроса	/api/room/delete.php
Входные данные	id: int (идентификационный номер комнаты)
Выходные данные	Status 200: Оповещение об успешном удалении. Status 400: Оповещение об ошибке.

3 ГЛАВА 3

3.1 Диаграмма классов приложения

Согласно паттерну проектирования MVC было реализовано два слоя архитектуры приложения, позволяющих реализовать работу серверной части и основную бизнес-логику: классы, являющиеся представлением и хранилищем данных (модель) и скрипты, обеспечивающее взаимодействие клиента с моделью (контроллеры). Таким образом все модели содержатся в директории /objects/, а контроллеры – в соответствующих им директориях (рисунок 7).

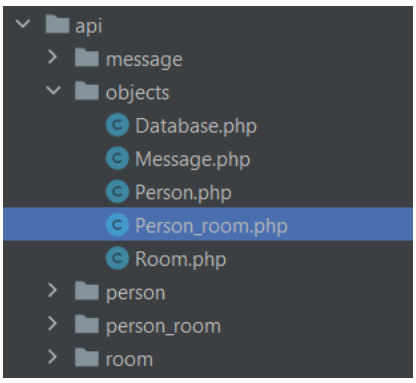

Рисунок 7 – Архитектура приложения

Каждой сущности, используемой для хранения данных, соответствует свой класс – «Message», «Person», «Person_room» и «Room». Класс «Database» предназначен для установки соединения с базой данных.

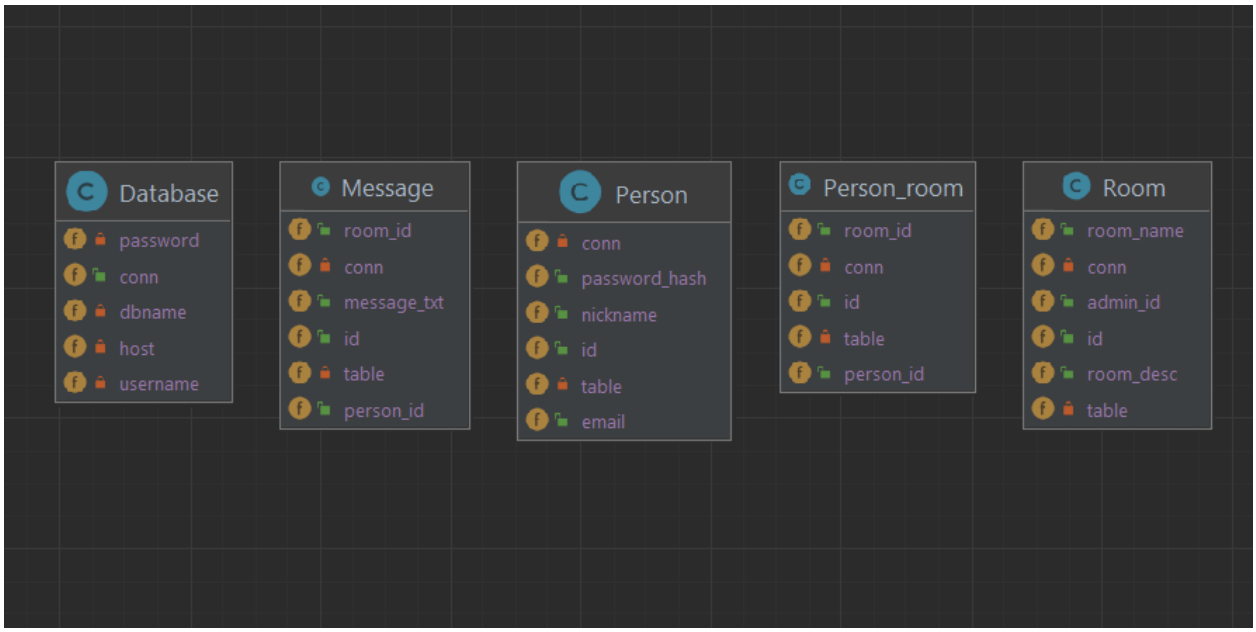

Рисунок 8 – Диаграмма классов сущностей

Как было упомянуто ранее, логика контроллеров описана в соответствующих каждой сущности скриптах. Для каждого запроса создан специальный скрипт, позволяющий реализовать основной функционал запроса. В каждом скрипте проходит обработка пользовательских параметров перед их взаимодействием с моделью и формируется результат запроса.

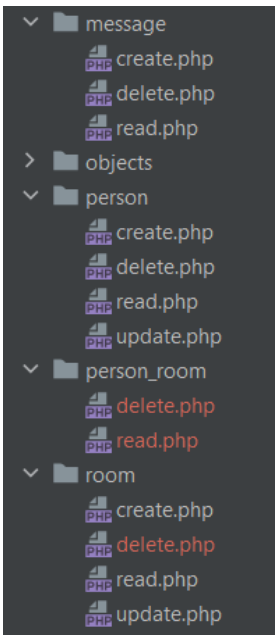

Рисунок 9 – Скрипты контроллеров приложения

3.2 Методика тестирования

Тестирование приложения проводилось вручную с помощью специализированного программного обеспечения Postman. Для удобства был создан каталог запросов (рисунок 10).

Рисунок 10 – Каталог запросов в Postman

Рисунок – Каталог запросов для тестирования серверной части

Получение ответа от серверной части является показателем успешного создания окружения с помощью Docker и успешного подключения к базе

данных. Для проверки корректной работы системы запросы отправлялись по определенному сценарию. Ниже представлены успешные результаты различных запросов.

На рисунках 11 – 14 представлены запросы, обслуживающие модель пользователя: создание и удаление учетной записи, просмотр всех пользователей и изменение псевдонима.

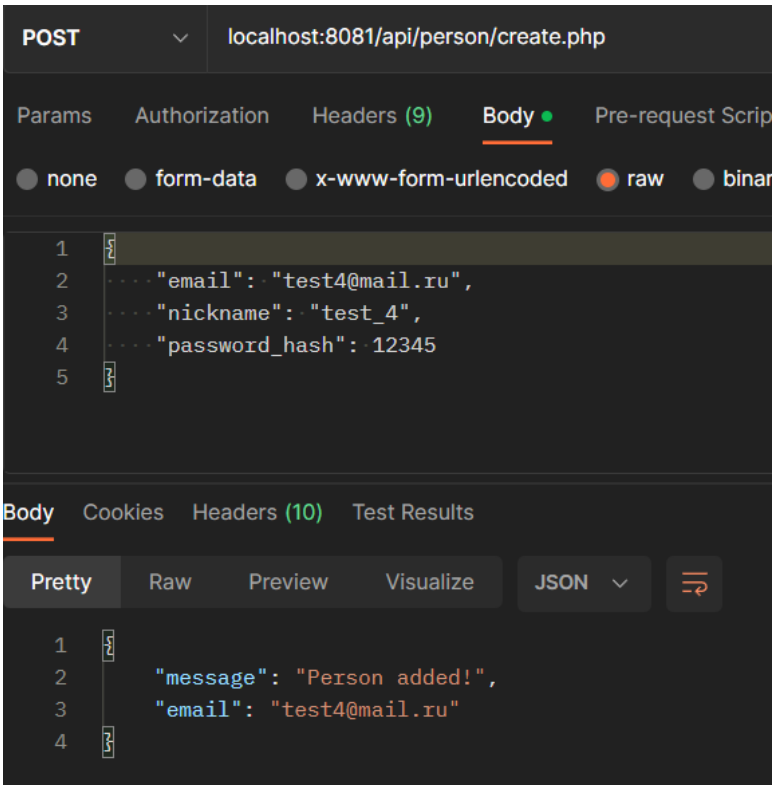

Рисунок 11 – Создание учетной записи

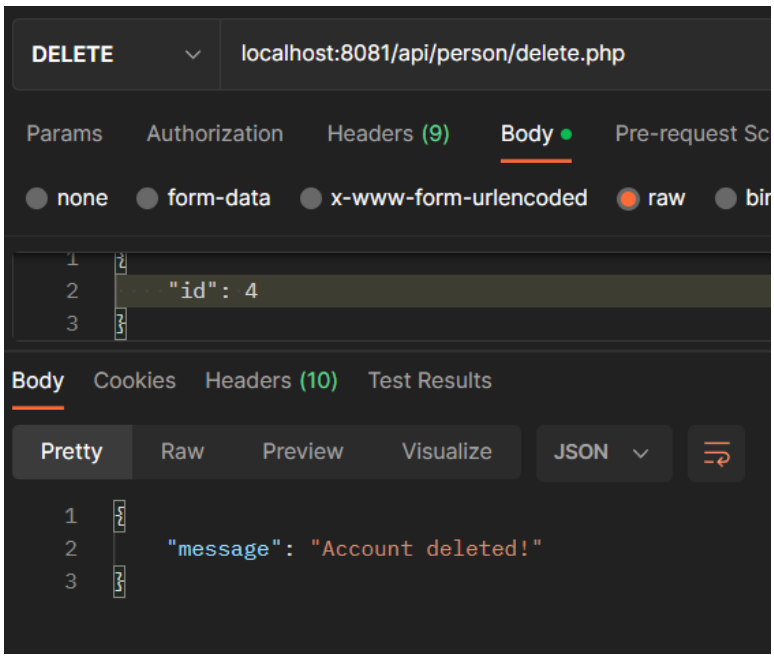

Рисунок 12 – Удаление учетной записи

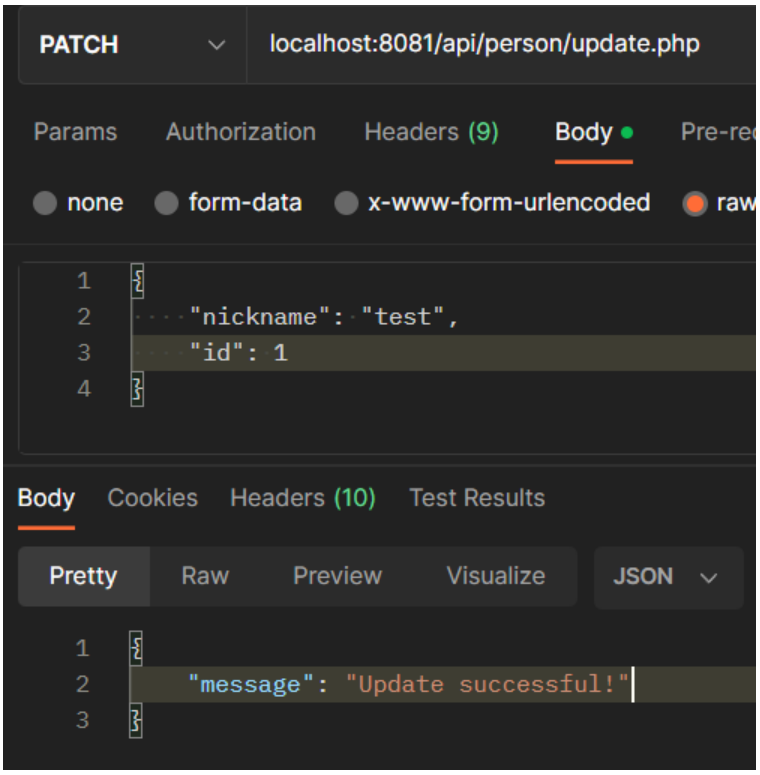

Рисунок 13 – Изменение псевдонима

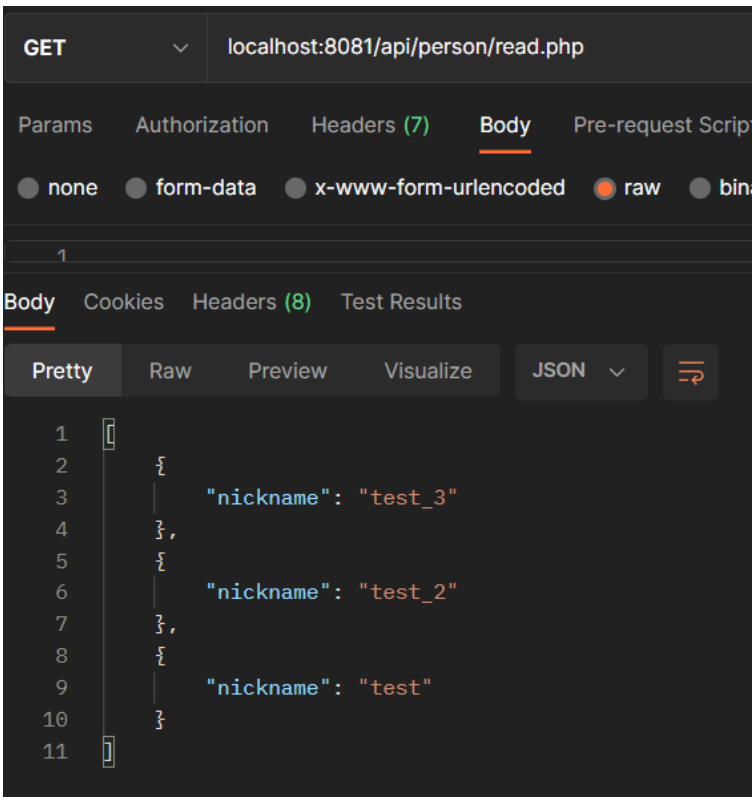

Рисунок 14 – Просмотр списка пользователей

На рисунках 15 – 18 представлены запросы, обслуживающие модель комнаты: создание и удаление комнаты, просмотр всех комнат и изменение данных комнаты.

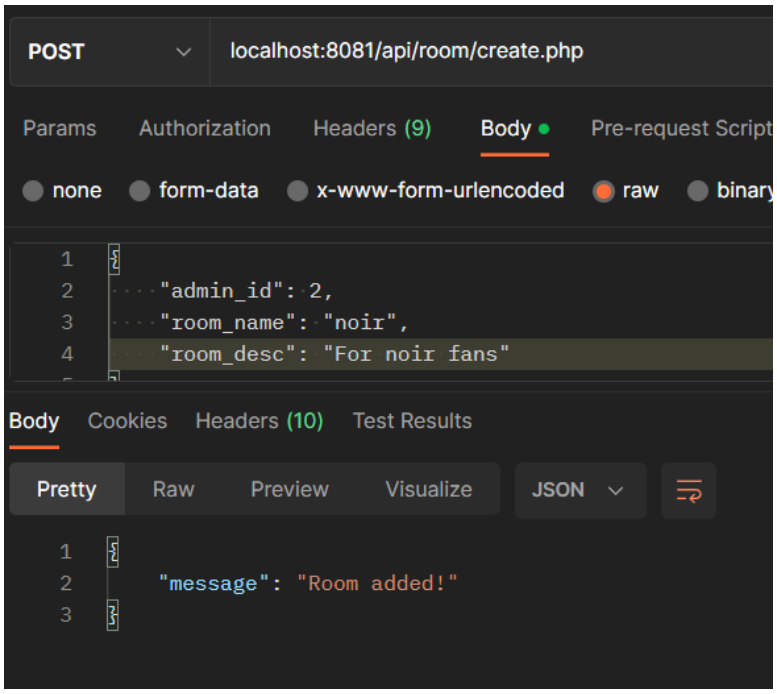

Рисунок 15 – Создание комнаты

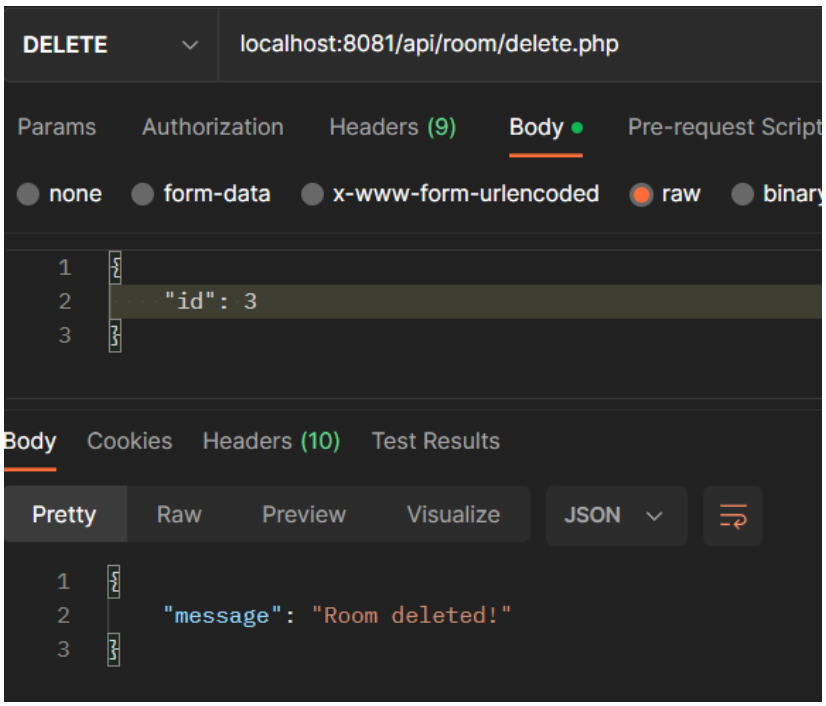

Рисунок 16 – Удаление комнаты

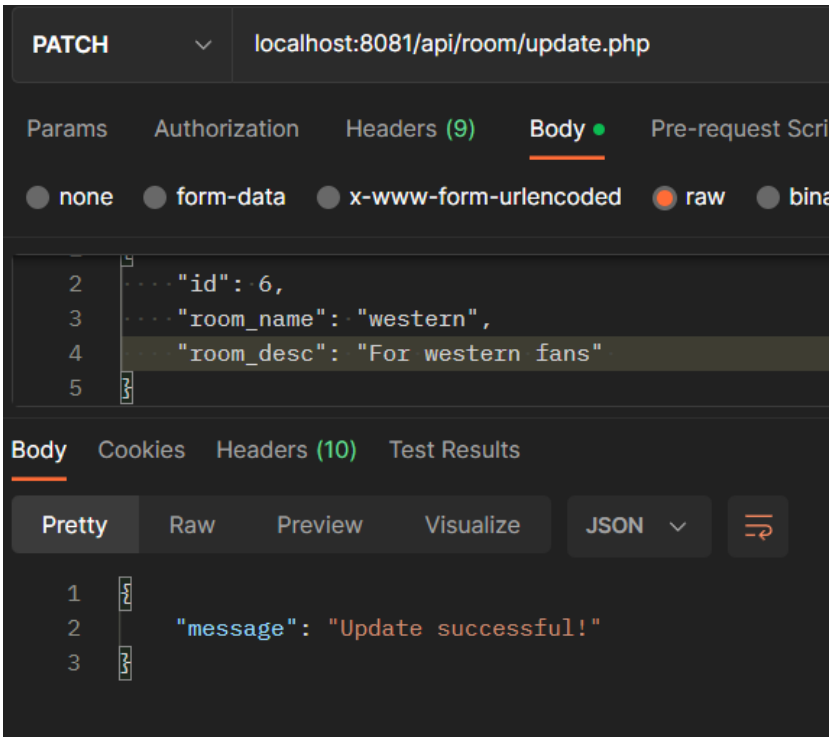

Рисунок 17 – Обновление данных о комнате

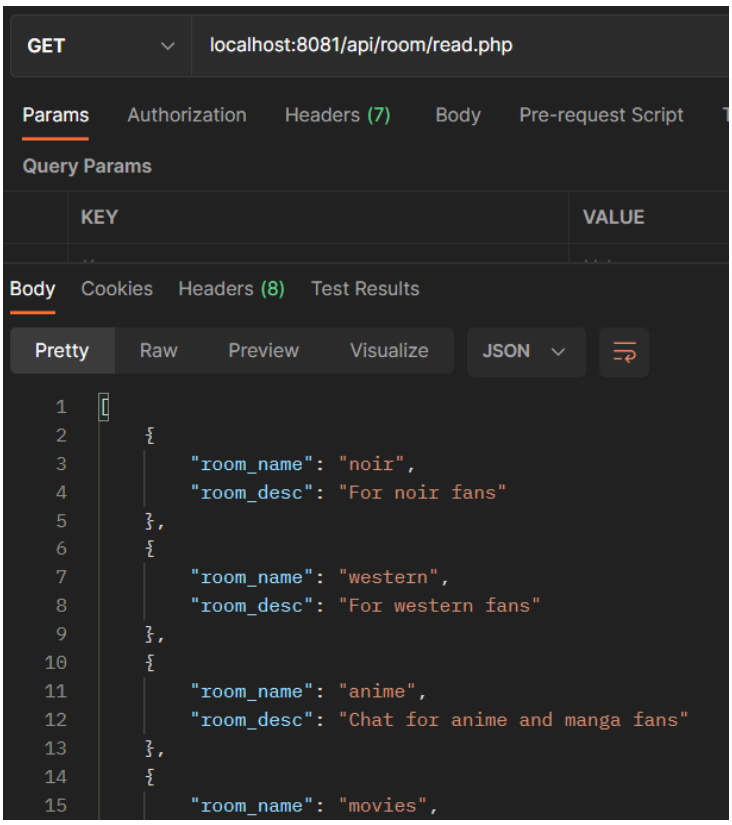

Рисунок 18 – Просмотр списка комнат

На рисунках 19 – 21 представлены запросы, обслуживающие модель сообщения: создание и удаление сообщения, просмотр всех сообщений в комнате.

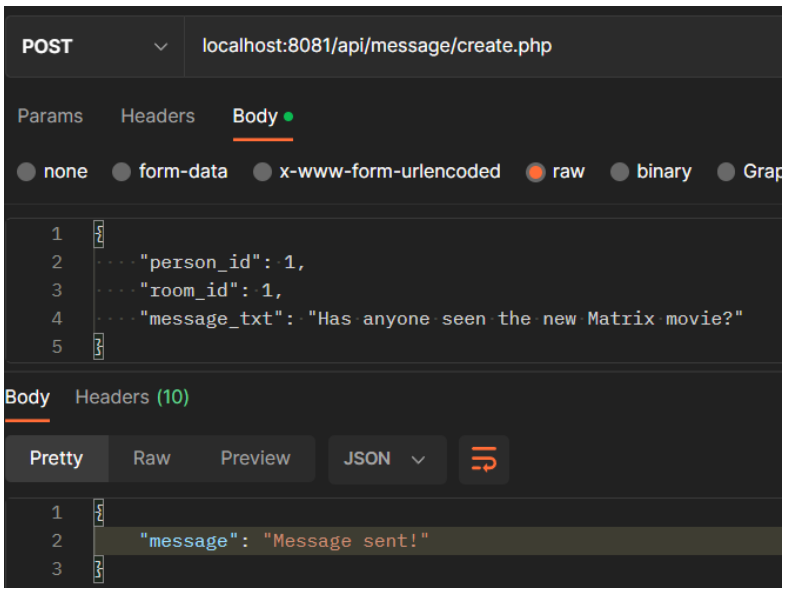

Рисунок 19 – Отправление сообщения

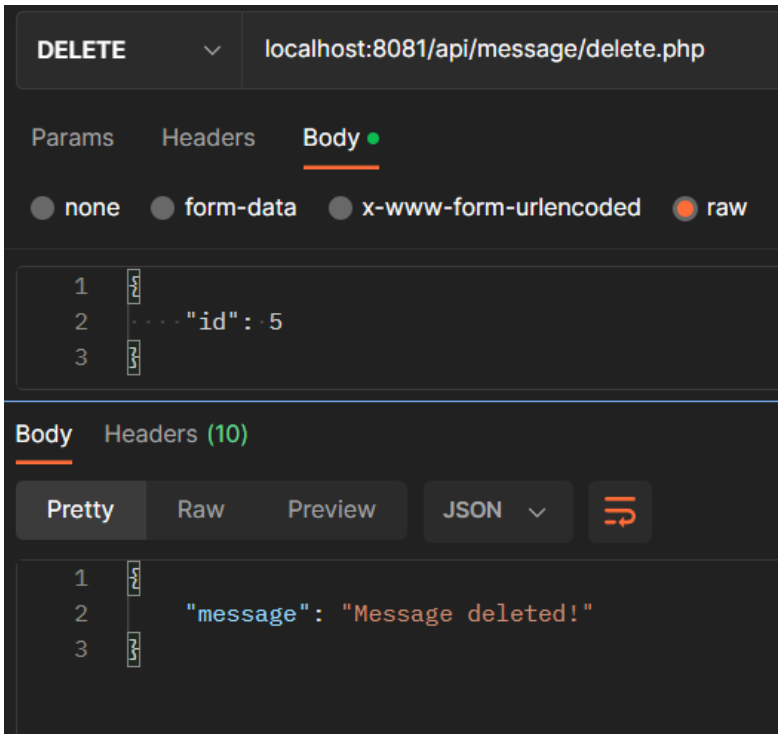

Рисунок 20 – Удаление сообщения

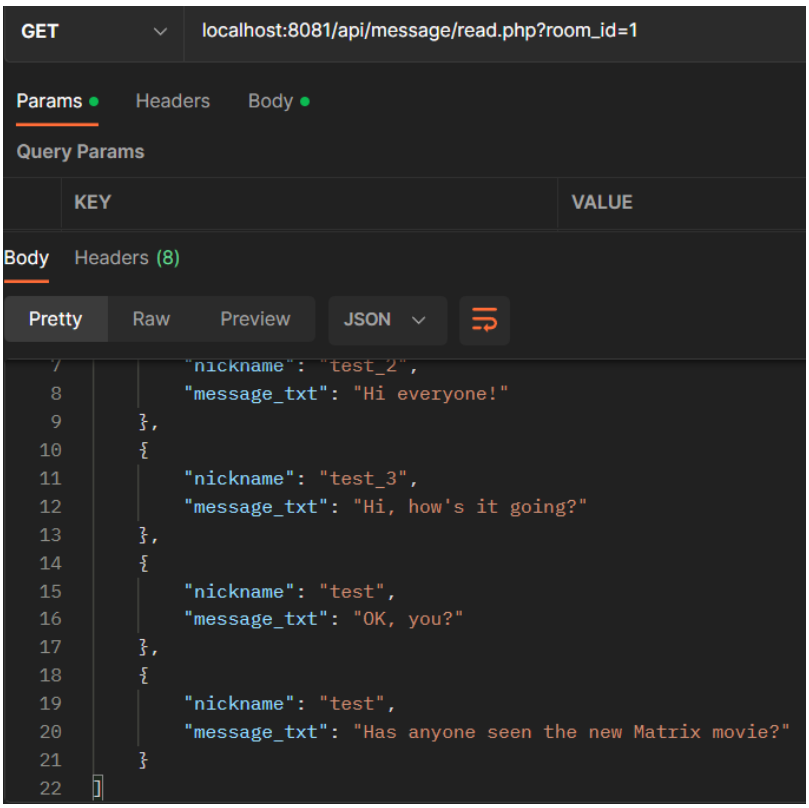

Рисунок 21 – Просмотр сообщений внутри комнаты

На рисунках 22 – 24 представлены запросы, обслуживающие модель с таблицей участников комнат: добавление и удаление участника из комнаты, просмотр всех участников.

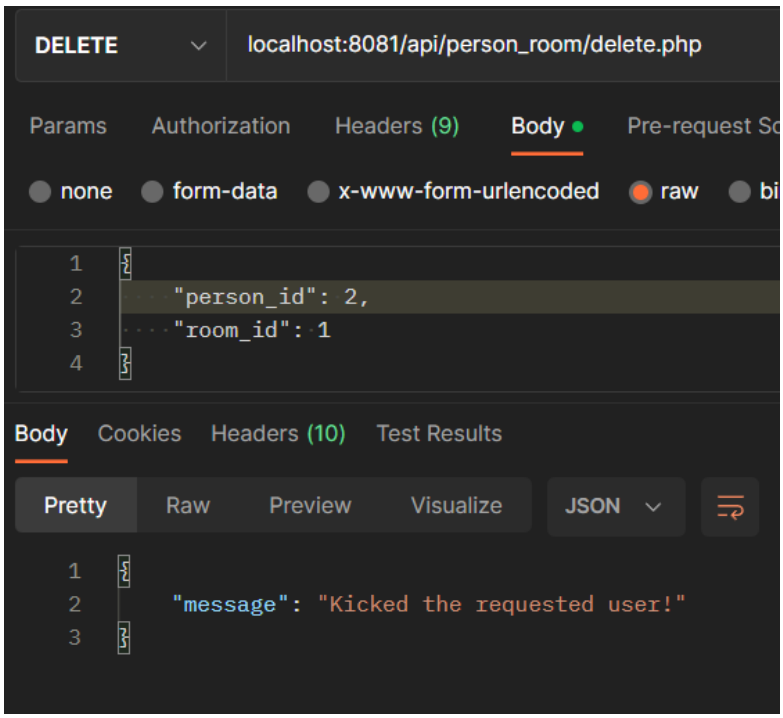

Рисунок 22 – Удаление пользователя из комнаты

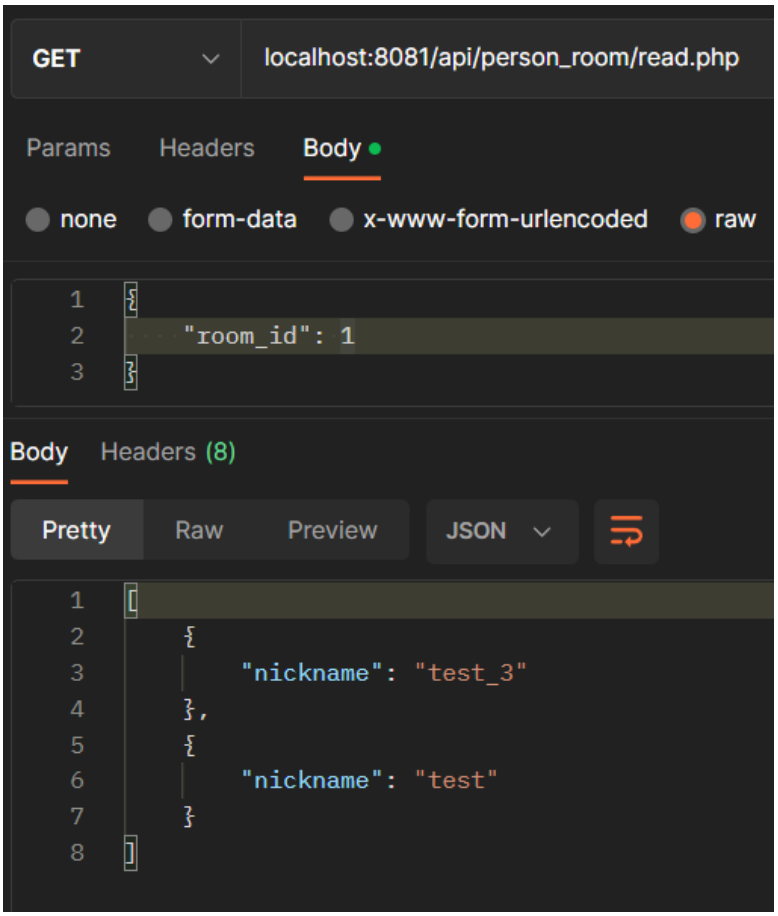

Рисунок 23 – Просмотр пользователей в комнате

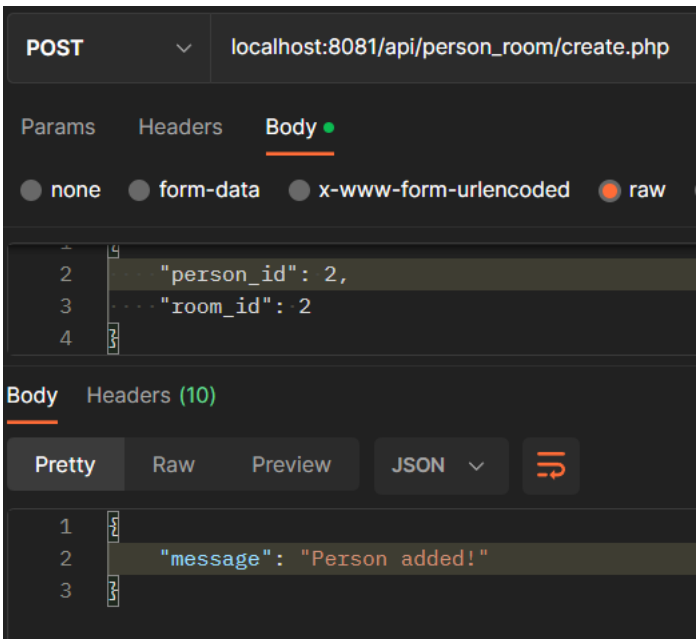

Рисунок 24 – Добавление пользователя в комнату

ЗАКЛЮЧЕНИЕ

В результате выполнения данной курсовой работы была выполнена поставленная цель, которая заключалась в разработке программного приложения с использованием паттерна проектирования MVC. Весь код был размещен на удаленном репозитории по ссылке https://github.com/Digilith/chitchat.

Согласно требованиям, в процессе разработки курсового проекта было выполнено проектирование информационной системы для чата. Отчёт по курсовой работе был оформлен в соответствии с ГОСТ 7.32—2017 и сформирован на основе инструкции по организации и проведению курсовых работ [11, 12].

Для достижения поставленной цели было необходимо решить следующие задачи: провести анализ предметной области, обосновать выбор средств ведения разработки, разработать архитектуру веб-приложения на основе выбранного паттерна проектирования, реализовать слой серверной логики веб-приложения с применением выбранной технологии, реализовать слой логики базы данных, разработать слой клиентского представления веб-приложения, подготовить презентацию выполненной курсовой работы.

В практической части работы были определены сущности системы, спроектированы таблицы в базе данных. Далее спроектирована информационная система для работы с приложением. Проведено контрольное тестирование приложения. Таким образом поставленные задачи выполнены, цель достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Потапенко Д.Ю. СОЦИАЛЬНАЯ СЕТЬ "ИНТЕРНЕТ" КАК ЯВЛЕНИЕ СОВРЕМЕННОЙ КОММУНИКАЦИИ // Вестник магистратуры. 2021. №3-2 (114). URL: https://cyberleninka.ru/article/n/sotsialnaya-set-internet-kak-yavlenie-sovremennoy-kommunikatsii (дата обращения: 11.12.2022).

2 Беловодская Е.А., Лагута К.А. СИСТЕМНОЕ ИССЛЕДОВАНИЕ ИСПОЛЬЗОВАНИЯ ЧАТ-БОТА В КОММУНИКАЦИИ С КЛИЕНТАМИ // Формирование рыночных отношений в Украине. 2020. №5 (228). URL: https://cyberleninka.ru/article/n/sistemnoe-issledovanie-ispolzovaniya-chat-bota-v-kommunikatsii-s-klientami (дата обращения: 11.12.2022).

3 Shoutbox [Электронный ресурс]. URL: http://gerd-tentler.de/tools/shoutbox/ (Дата последнего обращения: 01.11.2022)

4 Omegle: Talk to strangers! [Электронный ресурс] URL: https://www.omegle.com/ (Дата последнего обращения: 12.10.2022)

5 Chatzy: Free Private Chat Rooms [Электронный ресурс] URL: http://www.chatzy.com/ (Дата последнего обращения: 20.11.2022)

6 4 MVC on Servlet Stack [Электронный ресурс]. — URL: https://docs.spring.io/spring-framework/docs/current/reference/html/web.html (Дата последнего обращения: 15.10.2022)

7 Архитектура REST [Электронный ресурс]. — URL: https://habr.com/ru/post/38730/ (Дата последнего обращения: 03.11.2022)

8 HTTP Methods for RESTful Services [Электронный ресурс]. — URL: https://www.restapitutorial.com/lessons/httpmethods.html (Дата последнего обращения: 29.11.2022)

9 MySQL Documentation [Электронный ресурс]. — URL: https://dev.mysql.com/doc/ (Дата последнего обращения: 01.11.2022)

10 API | Documentation tool | Postman [Электронный ресурс]. — URLhttps://www.postman.com/api-documentation-tool/ (Дата последнего обращения: 11.12.2022)

11 JSON Web Tokens [Электронный ресурс]. — URL: https://jwt.io (Дата последнего обращения: 13.11.2022)

12 Documentation - PHP [Электронный ресурс]. URL: https://www.php.net/docs.php (Дата последнего обращения: 10.11.2022)

13 ГОСТ 7.32-2017 [Электронный ресурс] URL: https://www.rea.ru/ru/org/managements/orgnirupr/Documents/gost_7.32-2017.pdf (Дата последнего обращения: 01.11.2022)

14 Инструкция по организации и проведению курсового проектирования [Электронный ресурс] // СМКО МИРЭА 7.5.1/04. И.05-18 — URL: https://www.mirea.ru/upload/medialibrary/992/1325.pdf (Дата последнего обращения: 16.11.2022)