МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

Специальность 1-40 04 01 «Информатика и технологии программирования»

Отчет по преддипломной практике

на тему: «Программный комплекс автоматизации обслуживания жилого фонда студенческого общежития»

Исполнитель: студент гр. ИП-42
Пархоменко П.Л.
Руководитель от предприятия:
Макаревич В.Л.
Руководитель: преподаватель
Шибеко В.Н.

Дата проверки: _____
Дата допуска к защите: _____
Дата защиты: _____
Оценка работы: _____

Подписи членов комиссии _____

Гомель 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Автоматизация процессов позволяет увеличить эффективность работы, сократить время и затраты на выполнение задач, снизить вероятность ошибок и улучшить качество услуг. Также автоматизация помогает оптимизировать использование ресурсов, повысить скорость реакции на изменения внешней среды и упростить процессы для пользователя.

Целью дипломной работы является создание программного комплекса автоматизации жилого фонда студенческого общежития. Данный программный комплекс будет предназначен для оптимизации управления ресурсами общежития, упрощения и автоматизации процессов, связанных с проживанием студентов и поддержанием работы всего жилого комплекса.

Задачами дипломной работы являются:

– изучение методик разработки клиент-серверных приложений на базе стека MERN;

– классификация ролей пользователей и их ролевые политики;

– изучение методов реализации серверной части для приложения;

– проектирование структуры приложения, базы данных для хранения информации, формирование пользовательских правил для доступа к ресурсам и функциям приложения;

– разработка программных модулей, обеспечивающих авторизацию и аутентификацию пользователей; работу с данными с помощью графического интерфейса;

– верификация и опытная эксплуатация разработанного программного обеспечения.

1 АНАЛИТИЧЕСКИЙ ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ И СРЕДСТВ АВТОМАТИЗАЦИИ ЖИЛОГО ФОНДА СТУДЕНЧЕСКОГО ОБЩЕЖИТИЯ

1.1 Обзор существующих систем автоматизации

Многие университеты имеют свои личные жилые фонды (общежития), но не многие пытаются их автоматизировать, ведь в них протекают много различных и сложных процессов. Например, управление жилплощадью, распределение мест, контроль доступа, управление уборкой и ремонтом, а также учет платежей за проживание и услуги – все эти процессы могут быть оптимизированы и автоматизированы с помощью специальных программных решений. Такая автоматизация помогает университетам повысить эффективность управления общежитиями, снизить затраты и улучшить качество обслуживания студентов.

В большинстве случаем университеты предоставляют общую информацию об общежитии, сформированную в виде одной онлайн страницы. Например, на сайте университета БГПУ приведены следующие данные:

- адреса общежитий;
- фамилия, имя и отчество заведующего;
- фамилии, имена и отчества воспитателей и педагогов-организаторов,
- общее описание здания (количество мест, год постройки);
- наличие в общежитии общественных пространств (теннисные и тренажерные залы, актовый зал, кафе, прачечная, комната самоподготовки).

На сайте «Белорусского государственного университета» имеется информация, выраженная в виде рекомендаций и фотографий, например:

- какие бытовые приборы лучше всего взять с собой;
- как распланировать бюджет родителям.

Также существую другие средства автоматизация жилого фонда студенческого общежития. Эти системы включают в себя различные программные и аппаратные компоненты, такие как системы безопасности, системы управления доступом, системы учета и бронирования проживающих, системы мониторинга и управления энергопотреблением и т.д. Такие системы могут быть полезны для упрощения управления общежитием и улучшения комфорта проживания студентов. Например, системы управления доступом позволяют управлять доступом в здание и в отдельные комнаты, что обеспечивает безопасность и предотвращает несанкционированный доступ. Системы мониторинга и управления энергопотреблением позволяют управлять энергоресурсами и экономить деньги на коммунальных услугах.

Кроме того, системы учета и бронирования проживающих позволяют упростить процесс бронирования мест в общежитии и учета проживающих, что может значительно сократить время, затрачиваемое на административные

процессы.

В целом, автоматизация студенческих общежитий может быть очень полезна для улучшения условий проживания студентов и оптимизации управления общежитием.

1.2 Анализ разработки и проектирования веб приложений

Разработка и проектирование веб-приложений – это процесс создания программного обеспечения, которое работает на сервере и обеспечивает взаимодействие с пользователем через веб-браузер. Веб-приложения могут быть использованы для различных целей, включая электронную коммерцию, социальные сети, онлайн-банкинг, различные средства автоматизации и т.д.

Согласно [1], для создания веб-приложения нужно выполнить несколько шагов. Сначала определяются требования к приложению, которые могут включать в себя функциональные и нефункциональные требования, дизайн интерфейса, спецификации базы данных и т.д. Затем происходит проектирование архитектуры приложения, разработка базы данных и программного кода, а также тестирование и развертывание приложения на сервер.

Веб-приложение – это программное обеспечение, которое работает через веб-браузер и взаимодействует с пользователем посредством веб-интерфейса. В общем случае, веб-приложение состоит из следующих компонентов:

1. Клиентская часть (frontend) – это часть веб-приложения, которая выполняется в браузере пользователя и отображает интерфейс. Она написана на языке HTML, CSS и JavaScript, и обеспечивает взаимодействие пользователя с приложением. Клиентская часть может включать в себя различные компоненты, такие как фреймворки, библиотеки и т.д.
2. Серверная часть (backend) – это часть веб-приложения, которая работает на сервере и обрабатывает запросы пользователя, взаимодействует с базой данных и обеспечивает доступ к ресурсам и функциям приложения. Серверная часть может быть написана на различных языках программирования, таких как PHP, Python, Java, Ruby и т.д.
3. База данных – это хранилище данных, которые используются в приложении. База данных может быть реляционной или нереляционной и хранить информацию о пользователях, продуктах, заказах, и т.д.
4. API (Application Programming Interface) – это интерфейс, который позволяет взаимодействовать между клиентской и серверной частями приложения. API может использовать различные протоколы, такие как HTTP или WebSocket, и форматы данных, такие как JSON или XML.
5. Дополнительные компоненты – веб-приложение может содержать и другие компоненты, такие как библиотеки, плагины, сервисы сторонних разработчиков и т.д.

Компоненты веб-приложения взаимодействуют между собой и обеспечивают полноценное функционирование приложения. Они позволяют пользователям работать с приложением, выполнять различные действия и получать результаты их работы.

Разработка веб-приложений может включать в себя использование различных языков программирования, фреймворков, библиотек, систем управления базами данных и других инструментов. Например, для создания веб-приложения может использоваться язык программирования JavaScript, фреймворк React, база данных MySQL и сервер Apache.

Работа веб-приложения обычно происходит следующим образом: пользователь запрашивает определенную страницу через веб-браузер, сервер обрабатывает запрос и отправляет ответ обратно в браузер в виде HTML-страницы. Эта страница может содержать данные из базы данных, такие как пользовательский профиль или список продуктов, а также интерактивные элементы, такие как кнопки и формы, которые позволяют пользователю взаимодействовать с приложением.

В целом, веб-приложения могут быть очень полезны для обеспечения взаимодействия пользователей с программным обеспечением через веб-браузер.

Разработка и проектирование веб-приложений требует определенных навыков и знаний в области программирования, веб-технологий, дизайна пользовательского интерфейса и баз данных. Для успешной разработки веб-приложений необходимо понимание требований к приложению и его пользователей, умение разрабатывать эффективную архитектуру, использовать соответствующие инструменты и библиотеки, а также тестировать и развертывать приложение на сервер. Важным аспектом является также поддержка и обновление приложения после его запуска, а также обеспечение безопасности при обработке пользовательских данных и защите от взломов.

1.3 Анализ используемых технологий для реализации поставленной задачи

Веб-приложения могут быть написаны на разных языках программирования и фреймворках, но одним из наиболее популярных стеков технологий для разработки веб-приложений является стек MERN. MERN – это аббревиатура, состоящая из четырех популярных инструментов для веб-разработки: MongoDB, Express, React и Node.js.

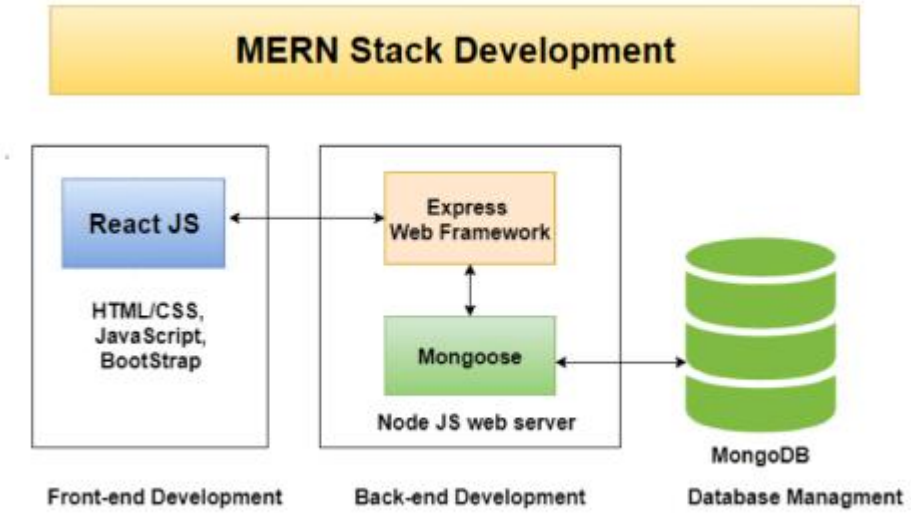

Рисунок 1.1 – Компоненты стека MERN

В основе этих фреймворков и библиотек находится язык программирования JavaScript.

JavaScript – мультипарадигменный (с одновременным использованием множества парадигм) язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript (стандарт ECMA-262). Его обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

Основные архитектурные черты:

– динамическая типизация;
– слабая типизация;
– автоматическое управление памятью;
– прототипное программирование;
– функции как объекты первого класса.

JavaScript включает в себя объектную модель браузера – браузер-специфичная часть языка, являющаяся прослойкой между ядром и объектной моделью документа. Основное предназначение объектной модели браузера — управление окнами браузера и обеспечение их взаимодействия. Каждое из окон браузера представляется объектом window, центральным объектом DOM. Объектная модель браузера на данный момент не стандартизирована, однако спецификация находится в разработке WHATWG и W3C.

В JavaScript используется в AJAX, популярном подходе к построению интерактивных пользовательских интерфейсов веб-приложений, заключающемся в «фоновом» асинхронном обмене данными браузера с веб-сервером. В результате, при обновлении данных веб-страница не

перезагружается полностью и интерфейс веб-приложения становится быстрее, чем это происходит при традиционном подходе (без применения AJAX) [2].

MongoDB – это документо-ориентированная NoSQL база данных, которая используется для хранения, управления и обработки больших объемов данных. Она позволяет хранить данные в формате документов BSON (Binary JSON), которые имеют структуру, аналогичную формату JSON, но могут содержать бинарные данные, даты и другие типы данных.

MongoDB разрабатывалась с учетом масштабируемости и производительности, что делает ее особенно привлекательной для проектов, которые требуют быстрой и эффективной обработки больших объемов данных. Она имеет множество функций, которые делают ее полезной для многих типов приложений, включая:

1. Гибкий документо-ориентированный подход: MongoDB предоставляет гибкую модель документо-ориентированной базы данных, что позволяет хранить данные любого формата, включая структурированные, полуструктурированные и неструктурированные данные;
2. Высокая доступность: MongoDB предоставляет высокую доступность благодаря встроенному механизму репликации, который позволяет хранить несколько копий данных на разных серверах. Если один сервер не работает, MongoDB может быстро переключиться на другой сервер и продолжить работу;
3. Масштабируемость: MongoDB легко масштабируется, что позволяет ей обрабатывать большие объемы данных. MongoDB поддерживает масштабирование горизонтальным и вертикальным способами;
4. Мощный язык запросов: MongoDB предоставляет мощный язык запросов, который позволяет быстро и эффективно извлекать данные из базы данных;
5. Индексы: MongoDB поддерживает множество типов индексов, которые позволяют ускорить поиск и фильтрацию данных;

MongoDB широко используется в различных областях, включая социальные сети, онлайн-магазины, приложения для анализа данных и многие другие. Ее популярность объясняется ее гибкостью, производительностью и масштабируемостью [3].

Express.js – это легковесный фреймворк для Node.js, который используется для разработки веб-приложений и API. Он предоставляет удобный и гибкий механизм для обработки запросов и ответов, маршрутизации и создания модульной структуры приложений.

Основные преимущества Express.js:

1. Удобство: Express.js облегчает разработку веб-приложений благодаря простому и интуитивно понятному API. Он позволяет быстро создавать маршруты, обрабатывать запросы и ответы, а также работать с различными middleware-пакетами;
2. Гибкость: Express.js позволяет разработчикам создавать приложения с различными функциональными возможностями. Он не навязывает строгую

структуру приложения, что дает возможность гибко настраивать его под конкретные задачи;

3. Масштабируемость: Express.js позволяет легко масштабировать приложения, что особенно важно для больших и сложных проектов. Он поддерживает работу с кластерами и позволяет распределять нагрузку между несколькими серверами;

4. Поддержка middleware: Express.js предоставляет широкие возможности для работы с middleware, что позволяет улучшать функциональность приложения и повышать его безопасность. С помощью middleware можно добавлять авторизацию, обработку ошибок, логгирование и многое другое;

5. Большое сообщество: Express.js имеет большое сообщество разработчиков, которые создают и поддерживают множество пакетов и расширений. Это позволяет разработчикам быстро решать задачи и получать поддержку при возникновении проблем.

Express.js используется для создания различных типов приложений, включая API, веб-серверы, приложения для обработки данных и многие другие. Он позволяет быстро создавать и масштабировать приложения, обеспечивая при этом гибкость и удобство разработки [4].

React – это библиотека JavaScript, разработанная Facebook, которая используется для создания пользовательских интерфейсов. React использует декларативный подход для описания компонентов пользовательского интерфейса, что делает его более простым и понятным для разработчиков. Он позволяет создавать переиспользуемые компоненты, которые могут быть легко использованы для создания сложных пользовательских интерфейсов.

Основные преимущества React:

1. Декларативный подход: React использует декларативный подход для описания пользовательского интерфейса, что делает его более понятным для разработчиков. Он позволяет описывать, как должен выглядеть интерфейс, а не как его создать;

2. Переиспользуемые компоненты: React позволяет создавать переиспользуемые компоненты, которые могут быть легко использованы для создания сложных пользовательских интерфейсов. Это сокращает время разработки и улучшает качество кода;

3. Эффективный: React использует виртуальный DOM, который позволяет изменять только те элементы, которые действительно изменились. Это уменьшает количество дорогостоящих операций, связанных с обновлением интерфейса, и улучшает производительность приложения;

4. Большое сообщество: React имеет большое сообщество разработчиков, которые создают и поддерживают множество пакетов и расширений. Это позволяет разработчикам быстро решать задачи и получать поддержку при возникновении проблем;

5. Простота: React является относительно простым и понятным инструментом для создания пользовательских интерфейсов. Это делает его

доступным для начинающих разработчиков и уменьшает время на обучение;

React используется для создания интерактивных пользовательских интерфейсов, включая веб-приложения, мобильные приложения, игры и многое другое. Он позволяет создавать переиспользуемые компоненты, которые могут быть легко использованы для создания сложных пользовательских интерфейсов [5].

Node.js – это среда выполнения JavaScript на стороне сервера, которая позволяет разрабатывать высокопроизводительные и масштабируемые веб-приложения. Она основана на движке V8, разработанном Google для браузера Chrome, и позволяет использовать JavaScript для создания приложений на серверной стороне.

Основные преимущества Node.js:

1. Высокая производительность: Node.js основан на движке V8, который обеспечивает быстрое выполнение JavaScript. Node.js также позволяет использовать асинхронное программирование, что улучшает производительность приложений;
2. Масштабируемость: Node.js позволяет создавать масштабируемые приложения с помощью механизма обработки запросов в нескольких потоках. Это позволяет распределять нагрузку на несколько серверов и обеспечивать высокую доступность приложения;
3. Широкие возможности: Node.js имеет большое количество библиотек и модулей, которые позволяют упростить разработку и расширить функциональность приложения. Это позволяет создавать приложения для различных сфер, включая веб-приложения, мобильные приложения, игры и многое другое;
4. Единый язык: Node.js использует JavaScript как единый язык для программирования на серверной и клиентской стороне, что упрощает разработку и повышает эффективность работы разработчика;
5. Активное сообщество: Node.js имеет большое сообщество разработчиков, которые создают и поддерживают множество пакетов и расширений. Это позволяет разработчикам быстро решать задачи и получать поддержку при возникновении проблем;

Node.js используется для создания различных приложений на серверной стороне, включая веб-приложения, микросервисы, API и многое другое. Он позволяет разработчикам создавать высокопроизводительные и масштабируемые приложения с использованием JavaScript на стороне сервера, что упрощает разработку и повышает эффективность работы разработчика.

1.4 Анализ инструментальных средств автоматизации разработки и тестирования

Для создания программного комплекса по обслуживанию жилого фонда студенческого общежития будет использована среда разработки Webshtorm от

компании JetBrains. Для визуализации базы данных MongoDb лучше всего подходит приложение MongoDb Compass. Хранить исходный код только на локальном компьютере плохая практика, поэтому будет создан удаленный репозиторий на GitHub. Для постройки различных диаграмм, которые позволят упростить разработку, а также предоставят полное понимание работы приложения, будет использоваться StarUml.

Рассмотрим подробнее каждый из этих инструментов.

Webshtorm – это интегрированная среда разработки (IDE), которая предоставляет обширный функционал для разработки веб-приложений. Среда разработки позволяет работать с различными языками программирования, включая JavaScript, TypeScript, HTML, CSS, Node.js, Angular, React и другие. Webshtorm включает в себя встроенные инструменты отладки, систему автодополнения кода, автоматическую проверку ошибок, систему контроля версий, анализаторы кода и многие другие полезные функции.

Webshtorm обладает многоплатформенностью, что позволяет разрабатывать на разных операционных системах, включая Windows, macOS и Linux. Среда разработки также имеет мощную систему плагинов, которая позволяет расширять функционал IDE, добавляя поддержку новых языков программирования и инструментов. Она также имеет множество инструментов для работы с базами данных, включая поддержку MongoDB, MySQL, PostgreSQL, Oracle и других, обеспечивает интеграцию с браузерами для отладки веб-приложений в режиме реального времени.

Одним из основных преимуществ Webshtorm является его эффективность и производительность. Среда разработки использует многопоточную архитектуру и оптимизированный механизм работы с памятью, что обеспечивает быстродействие и позволяет работать с большими проектами. Кроме того, Webshtorm имеет обширную документацию и активное сообщество пользователей, которые создают полезные плагины, советы и обучающие ресурсы, что делает процесс разработки еще более комфортным и эффективным.

Webshtorm – это мощная среда разработки, которая облегчает и ускоряет процесс создания высококачественных веб-приложений, идеально подходящая для опытных и начинающих разработчиков [6].

MongoDB Compass – это визуальный интерфейс для работы с базами данных MongoDB. Этот инструмент позволяет разработчикам и администраторам баз данных MongoDB легко просматривать, анализировать и манипулировать данными. Он предоставляет графический интерфейс для создания, редактирования и удаления коллекций, документов и индексов базы данных MongoDB. Он также обеспечивает визуализацию структуры коллекций, что помогает быстро понять структуру данных.

MongoDB Compass имеет встроенный механизм запросов, который позволяет быстро и легко создавать и выполнять запросы к базе данных. Он также обеспечивает удобный механизм фильтрации и сортировки данных, а также поддержку агрегационных запросов.

Среди других возможностей MongoDB Compass – поддержка графического интерфейса для выполнения команд в MongoDB, создание и сохранение запросов для повторного использования, анализ статистики запросов и многое другое.

MongoDB Compass также обеспечивает удобный механизм подключения к серверу базы данных MongoDB, что позволяет работать с удаленными базами данных. Он также поддерживает механизм аутентификации и авторизации, что обеспечивает безопасность при работе с базами данных.

В целом, MongoDB Compass – это удобный и мощный инструмент для работы с базами данных MongoDB, который обеспечивает широкий спектр возможностей для работы с данными, а также удобный интерфейс для управления базой данных. Этот инструмент идеально подходит для разработчиков и администраторов баз данных MongoDB, которые хотят упростить и ускорить работу с данными [7].

GitHub – это веб-сервис для хранения и совместной работы над Git-репозиториями. Это платформа, которая позволяет разработчикам хранить и совместно работать над кодом, отслеживать ошибки, создавать ветки, а также просматривать и редактировать код, управлять версиями, изменениями и запросами на слияние.

Репозиторий в GitHub – это хранилище для кода и других файлов, которые могут быть загружены и управляемы в Git. Репозиторий в GitHub позволяет хранить и управлять кодом в облаке, а также предоставляет множество функций для работы с кодом, включая возможность комментирования кода, управления задачами и многое другое.

Если хранить код только на локальном компьютере, то это может привести к потере данных в случае сбоя жесткого диска или других проблем с компьютером. Кроме того, хранение кода на локальном компьютере не обеспечивает возможность совместной работы и синхронизации изменений между различными разработчиками.

Таким образом, GitHub – это мощный и удобный инструмент для хранения и совместной работы над кодом. Он предоставляет широкий спектр возможностей для управления кодом, удобный интерфейс для просмотра и редактирования кода, а также возможность совместной работы и синхронизации изменений между различными разработчиками.

StarUML – это приложение для создания UML-диаграмм, которое позволяет разработчикам создавать модели проектов, планировать архитектуру и дизайн приложений. StarUML имеет графический интерфейс пользователя, который предоставляет множество инструментов для создания и редактирования диаграмм, включая диаграммы классов, диаграммы последовательностей, диаграммы состояний, диаграммы активностей и многое другое.

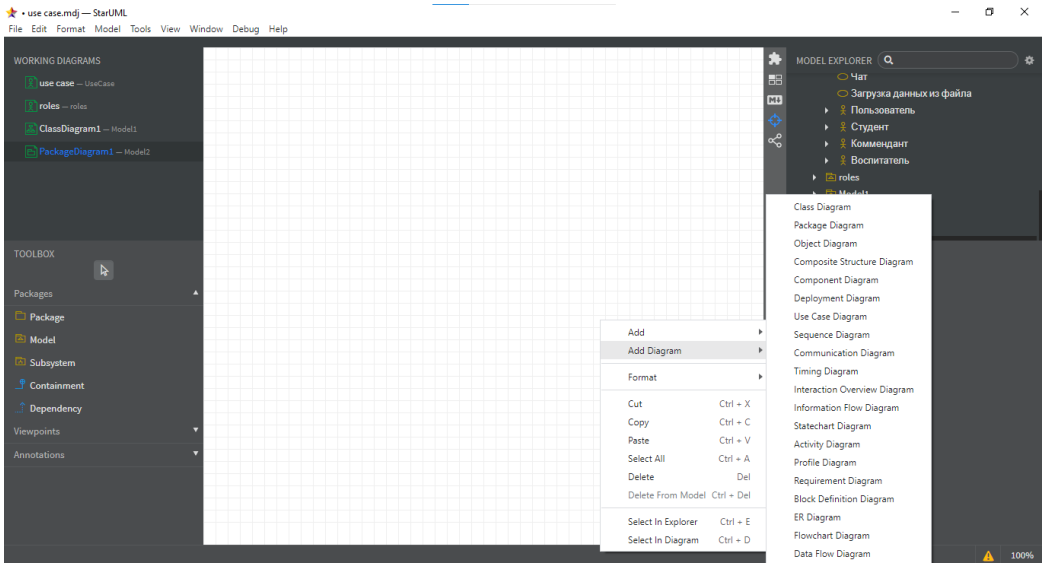

Рисунок 1.2 – Интерфейс главного окна StarUml

В StarUML можно создавать различные типы диаграмм, включая:

1. Диаграммы классов – используются для описания структуры классов и их отношений;
2. Диаграммы последовательностей – используются для описания взаимодействия между объектами и процессов, происходящих во времени;
3. Диаграммы состояний – используются для описания жизненного цикла объекта и его состояний;
4. Диаграммы компонентов – используются для описания структуры и отношений между компонентами системы;
5. Диаграммы развертывания – используются для описания физического размещения компонентов и системы в целом;
6. Диаграммы активностей – используются для описания последовательности действий и процессов в системе.

На рисунке 1.2 в виде выпадающего меню представлен весь список возможных диаграмм.

StarUML поддерживает различные языки моделирования, включая UML 2.x, SysML, ERD, BPMN, SoaML и другие. Он также позволяет пользователям импортировать и экспортировать диаграммы в различных форматах файлов, включая XMI, SVG, PDF, PNG, JPEG и другие. Данное приложение имеет множество функций, которые облегчают создание и редактирование диаграмм, такие как автоматическое выравнивание элементов, возможность группировки элементов, подсветка синтаксиса, подсказки и многое другое. Кроме того, StarUML позволяет пользователю создавать свои собственные элементы, шаблоны и плагины, чтобы улучшить функциональность и адаптировать инструмент под свои потребности.

StarUML – это мощный и удобный инструмент для создания UML-диаграмм, который предоставляет широкий спектр возможностей для создания

и редактирования диаграмм, а также импорта и экспорта диаграмм в различных форматах файлов. Он также имеет множество функций, которые облегчают создание и редактирование диаграмм, и может быть расширен за счет пользовательских элементов, шаблонов и плагинов [8].

Postman – это инструмент для тестирования API, который позволяет разработчикам быстро и удобно отправлять запросы к API, тестировать их и анализировать ответы.

С помощью Postman можно отправлять запросы на сервер и получать ответы в разных форматах, таких как JSON, XML, HTML и другие. Также в Postman есть множество функциональных возможностей, таких как автоматическое генерирование документации, управление авторизацией, создание и выполнение тестовых сценариев и другие. Он поддерживает различные типы запросов, включая GET, POST, PUT, DELETE и многие другие. Вы можете отправлять запросы с параметрами, заголовками и телом запроса в формате JSON или форм-данных.

Один из самых удобных и полезных аспектов Postman – это коллекции запросов, которые можно создавать и организовывать для более эффективного управления тестированием API. Коллекции могут быть экспортированы и импортированы в различных форматах, таких как JSON и XML.

Postman имеет множество преимуществ, которые делают его популярным среди разработчиков, вот некоторые из них:

1. Удобный интерфейс. Интерфейс Postman интуитивно понятен и удобен в использовании. Для отправки запроса не нужно писать код, все нужные функции находятся в графическом интерфейсе.

2. Простота в использовании. Postman не требует от разработчика особых навыков, чтобы начать использовать его. Для создания запросов нужно просто заполнить соответствующие поля.

3. Расширяемость. Postman можно легко расширять, используя плагины. Например, с помощью плагина можно добавить поддержку авторизации на основе OAuth или создать пользовательский набор инструментов для тестирования API.

4. Автоматизация. Postman позволяет автоматизировать тестирование API. Это особенно полезно при наличии большого количества запросов, которые нужно тестировать.

5. Множество функций. Postman имеет множество функций, таких как создание коллекций, генерация документации и выполнение тестовых сценариев, что делает его полезным инструментом для разработки и тестирования приложений.

6. Бесплатность. Postman имеет бесплатную версию, которая покрывает большинство потребностей разработчиков, а также платную версию с дополнительными функциями.

Postman является полезным и удобным инструментом для тестирования API, который может существенно ускорить процесс разработки приложений [9].

1.5 Техническое задание для клиент-серверного программного продукта «Программный комплекс автоматизации обслуживания жилого фонда студенческого общежития»

Цель разработки: разработать приложение, предназначенное для автоматизации обслуживания жилого фонда студенческого общежития.

Данный программного комплекс предназначается для студентов и работников студенческого общежития.

Приложение позволит коменданту следующее:

– вести учет технического оборудования, которое ввозят студенты для личного пользования;

– вести учет количества свободных и занятых мест с возможность фильтрации за год или за месяц;

– для заполнения списка студентов комендант сможет импортировать файл формата Excel в базу данных;

– мониторинг оплаты за проживание.

Для студентов будет реализовано:

– личный кабинет, где они смогут следить за текущими новостями и событиями;

– чат с воспитателем;

– возможность создавать заявки на починку бытового оборудования.

Для воспитателей приложение предусматривает следующие функции:

– создание и ведение ленты новостей;

– создание и ведение ленты событий;

– начисление баллов студентам;

– создание чата со студентом или общей беседы;

– оставлять замечания в личном кабинете студента.

Приложение должно иметь следующую структуру и функциональность:

– работать как веб-приложение основанное на REST архитектуре;

– являться кроссплатформенным;

– являться кроссбраузерным;

– иметь принцип работы, аналогичный веб-приложениям (реализовывать REST архитектуру, быть доступным посредством веб-браузера, иметь пользовательский интерфейс);

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И АЛГОРИТМЫ

2.1 Анализ предметной области

Анализ предметной области «Автоматизация обслуживания жилого фонда студенческого общежития» позволяет выделить следующие основные аспекты:

1. Учет и распределение комнат. Для автоматизации данного процесса необходимо иметь возможность вести учет свободных и занятых комнат, а также осуществлять распределение студентов в свободные комнаты.
2. Регистрация и учет студентов. Необходимо иметь возможность регистрировать студентов и вести учет их проживания в общежитии, а также осуществлять контроль за своевременной оплатой за проживание.
3. Предоставление различной информации. Необходимо иметь возможность создавать и просматривать новостные ленты для того.
4. Связь воспитателей и студентов. Необходимо иметь возможность связываться с студентом для решения вопросов, связанных с проживанием в общежитии.

Для автоматизации этих процессов может быть использована специализированная информационная система, разработанная с учетом конкретных потребностей студенческого общежития. Она будет включать в себя базу данных для хранения информации о студентах и комнатах, модули для управления ресурсами и контроля доступа, а также возможности для сотрудников общежития добавлять и изменять информацию о конкретном студенте.

Таким образом, разработка автоматизированной системы позволит упростить и ускорить процессы управления жилым фондом студенческого общежития, повысить эффективность использования ресурсов и улучшить качество обслуживания студентов. Благодаря использованию базы данных, все данные будут храниться в одном месте и сотрудники общежития смогут быстро получать необходимую информацию о студентах и комнатах. А модули управления ресурсами и контроля доступа позволят более точно контролировать использование ресурсов и обеспечить безопасность в общежитии. Разработка системы также может способствовать улучшению учебного процесса, так как облегчит жизнь студентам и позволит им сконцентрироваться на учебе, а не на решении бытовых вопросов.

2.2 Функциональная модель программного комплекса

Для того, чтобы создать качественное приложение, нужно четко понимать, какие роли у нас будут и какой функционал им будет доступен. Для этого нужно знать, какие объекты попадают в предметную область проектируемой системы и какие логические связи между ними существуют. Для формирования такого

понимания используются логические модели предметной области. Целью построения логической модели является получение графического представления логической структуры исследуемой предметной области. Для стабильной работы программного обеспечения необходимо чёткое распределение на роли, на основе которых будут формироваться функции взаимодействия со внутренней средой приложения.

Актер – множество логически связанных ролей в UML, исполняемых при взаимодействии с прецедентами или сущностями (система, подсистема или класс). Актером может быть человек или другая система, подсистема или класс, которые представляют нечто вне сущности.

Прецеденты представляют действия, выполняемые системой в интересах актеров. Проще говоря, прецедент – это описание последовательности действий (или нескольких последовательностей), которые выполняются системой и производят для отдельного актера видимый результат. Один актер может использовать несколько элементов прецедентов, и наоборот, один прецедент может иметь несколько актеров, использующих его. Каждый прецедент задает определенный путь использования системы. Набор всех прецедентов определяет полные функциональные возможности системы.

Приложение должно реализовывать три роли: комендант, воспитатель, студент.

Основные функции для роли «Комендант»:

- учет личного технического оборудования;
- учет количества свободных и занятых мест;
- возможность импортировать данные о студентах формата Excel;
- производить мониторинг оплаты за проживание.

Основные функции для роли «Студент»:

- просмотр новостей и событий;
- чат с воспитателем;
- создавать заявки на починку бытового оборудования.

Основные функции для роли «Воспитатель»:

- создание и ведение ленты новостей и событий;
- начисление баллов студентам;
- создание чата со студентом;
- создание общей беседы;
- возможность оставлять замечания в личном кабинете студента

В качестве среды для разработки и отображения функциональной структуры программы будет использована StarUml. На рисунке 2.2 представлена Use Case диаграмма приложения.

Рисунок 2.1 – Диаграмма прецедентов и актеров

Далее приведено описание прецедентов для роли «Комендант».

Прецендент «Учет личного техники» предназначен для просмотра и управления личным техническим оборудованием студента.

Прецендент «Загрузка данных из файла» предназначен для выгрузки личных данных студентов в базу данных.

Прецендент «Мониторинг оплаты общежития» предназначен для просмотра задолженности перед общежитием, а также для информирования студента.

Прецендент «Просмотр занятых и свободных мест» предназначен для мониторинга занятости комнат.

Далее приведено описание прецендентов для роли «Воспитатель».

Прецендент «Чат» предназначен для общения с отдельным студентом или с неким количеством.

Прецендент «Просмотр списка студентов» предназначен для просмотра информации о студентах в целом или более подробно об отдельном студенте.

Прецендент «Начисление баллов» предназначен для начисления баллов отдельному студенту.

Прецендент «Оставлять замечания». Данный прецендент необходим для того, чтобы воспитатель мог оповестить студента о каком-то совершенном нарушении в его личном кабинете.

Прецендент «Ведение ленты новостей» предназначен для просмотра и создания новых новостей.

Прецендент «Ведение ленты мероприятий» предназначен для просмотра и создания с целю оповещения новых событий.

Диаграмма прецендентов и актеров отображает функциональность системы с точки зрения пользователей. Она показывает взаимодействие между пользователем и системой в рамках конкретной задачи или сценария использования.

2.3 Информационная модель программного комплекса

На основе анализа предметной области определим набор коллекций и их свойства. В MongoDB коллекция представляет собой группу документов, которые хранятся в базе данных. Коллекции в MongoDB являются аналогом таблиц в реляционных базах данных. Они содержат документы в формате JSON, которые могут иметь различную структуру и не обязательно должны иметь одинаковые поля.

Для отображения информационной модели разрабатываемого продукта были выделены следующие коллекции:

– коллекция «Студент». Хранит в себе информацию о студентах, а также о комнате, где он живет;

– коллекция «Аккаунт». Хранит в себе данные для входа в личный аккаунт пользователя;

– коллекция «Сотрудники». Хранит в себе личные данные сотрудников общежития;

– коллекция «Новость». Хранит в себе весь список новостей;

– коллекция «Чат». Хранит список всех чатов.

– коллекция «Заявки». Хранит информацию о заявке студента на починку бытового оборудования.

На картинке 2.2 приведена схема базы данных.

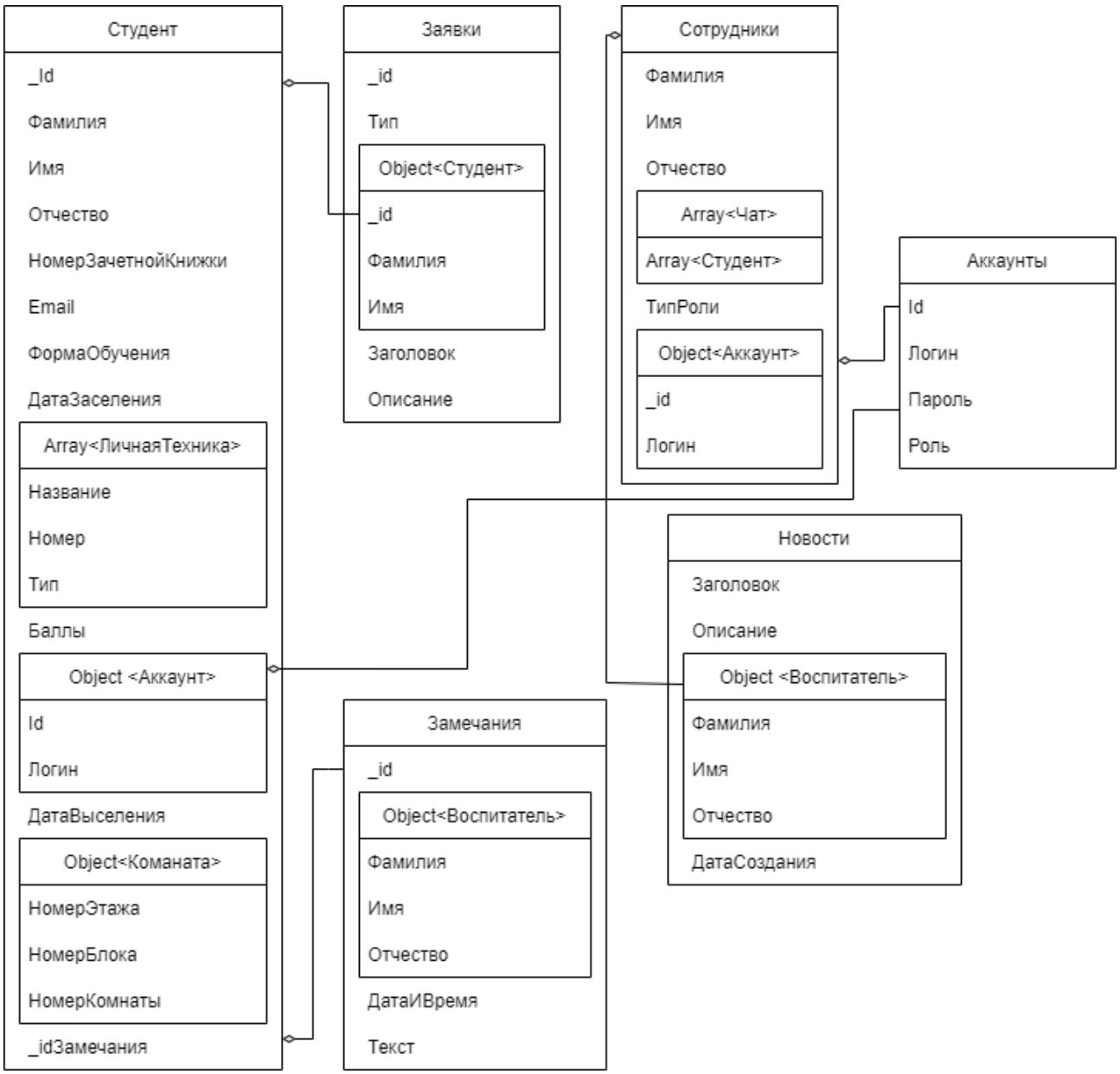

Рисунок 2.2 – Схема базы данных

В таблицах 2.1 – 2.6 представлено подробное описание полей в коллекциях.

Таблица 2.1 – Коллекция «Студент»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
firstName	string	Да	Нет

Продолжение таблицы 2.1

Название	Тип	Обязательное (required)	Уникальное (unique)
secondName	string	Да	Нет
middleName	string	Да	Нет
numberTest	string	Да	Да
email	string	Нет	Да
formEducation	union ("платное" | "бесплатное")	Да	Нет
dateEntry	string	Да	Нет
balls	number	Да	Нет
privateTechs	Array<Object>	Нет	Нет
privateTech.model	string	Нет	Нет
privateTech.number	string	Нет	Нет
privateTech.type	string	Нет	Нет
room	Object	Да	Нет
room.floor	number	Да	Нет
room.block	number	Да	Нет
room.apartament	number	Да	Нет
remarks	Array<Object>	Нет	Нет
remark.dateAndTime	string	Да	Нет
remark.header	string	Да	Нет
remark.status	string	Да	Нет
remark.text	string	Да	Нет
remark.mentor	Object	Да	Нет
remark.mentor.firstName	string	Да	Нет
remark.mentor.secondName	string	Да	Нет
remark.mentor.middleName	string	Да	Нет
account	Object	Да	Да
account.login	string	Да	Да

В коллекции, описанной в таблице 2.1 поле formEducation имет специальный тип union, предоставляемый TypeScript. Данный тип представляет объединение нескольких строк, которые разрешено присваивать данной переменной.

Таблица 2.2 – Коллекция «Новости»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
header	string	Да	Нет
description	string	Да	Нет
mentor	Object	Да	Нет
mentor.firstName	string	Да	Нет
mentor.secondName	string	Да	Нет
mentor.middelName	string	Да	Нет
dateCreate	date	Да	Нет

Как видно из таблицы 2.2, коллекция «Новости» хранит в себе данные воспитателя. Это необходимо для того, чтобы не осуществлять поиск по коллекции «Сотрудники» для установления данных создателя документа, что позволит нам сэкономить время загрузки данных.

Таблица 2.3 – Коллекция «Аккаунты»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
login	string	Да	Да
password	string	Да	Нет

В таблице 2.3 приведено описание полей коллекции «Аккаунты». Данная коллекция нужна для быстрого поиска пользователя в системе во время его авторизации.

Таблица 2.4 – Коллекция «Чат»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
name	String	Да	Нет
messages	Array<Object >	Да	Нет
messages.who	Object	Да	Нет
messages.who.firstName	String	Да	Нет
messages.who.secondName	String	Да	Нет
messages.who.middleName	String	Да	Нет
messages.when	Date	Да	Нет
messages.message	string	Да	Нет

Таблица 2.5 – Коллекция «Сотрудники»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
firstName	String	Да	Нет
secondName	string	Да	Нет
middleName	String	Да	Нет
Role	Union ("mentor" | "main")	Да	Нет
chats	Array<ObjectId>	Нет	Нет

В таблице 2.5 описаны поля для коллекции «Сотрудники». Все поля, кроме chats, обязательные. Поле chats хранит массив типа ObjectId. Данный массив заполняется id из коллекции «Чат», описанной в таблице 2.4.

Таблица 2.6 – Коллекция «Заявки»

Название	Тип	Обязательное (required)	Уникальное (unique)
_id	ObjectId	Да	Да
type	String	Да	Нет
student	Object	Да	Нет
student.firstName	string	Да	Нет

Продолжение таблицы 2.6

Название	Тип	Обязательное (required)	Уникальное (unique)
student.secondName	String	Да	Нет
student.room	number	Да	Нет
student.room.floor	Number	Да	Нет
student.room.block	Number	Да	Нет
student.room.apartament	Number	Да	Нет
header	String	Да	Нет
description	string	Да	Нет

Как можно заметить, каждая коллекция в MongoDB может иметь некоторые поля, которые могут быть обязательными или необязательными. Кроме того, поля могут быть уникальными или неуникальными в пределах коллекции. Поле, которое является уникальным, не позволяет вставлять документ с таким же значением, как уже имеющийся в коллекции. Поле, которое является обязательным, требует, чтобы его значение было указано при вставке документа. Если обязательное поле не указано, то операция вставки завершится неудачей. Коллекция также может иметь индексы, которые помогают ускорить выполнение запросов к базе данных. В целом, использование MongoDB позволяет эффективно хранить и управлять данными, предоставляя удобный интерфейс для доступа к ним.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Проектирование и разработка web-приложений. [Электронный ресурс] – Режим доступа: https://urait.ru/book/proektirovanie-i-razrabotka-web-prilozheniy-512113.
2. Что такое JavaScaript? [Электронный ресурс] – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/JavaScript/First_steps/What_is_JavaScript.
3. База данных MongoDb. [Электронный ресурс] – Режим доступа: https://itglobal.com/ru-by/company/glossary/mongodb/.
4. Express и NodeJs. [Электронный ресурс] – Режим доступа: https://developer.mozilla.org/ru/docs/Learn/Server-side/Express_Nodejs.
5. Фреймворк React. [Электронный ресурс] – Режим доступа: https://metanit.com/web/react/1.1.php.
6. IDE WebStorm. [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/WebStorm#:~:text=JetBrains%20WebStorm%20%20интегрированная%20среда%20разработки,на%20основе%20платформы%20IntelliJ%20IDEA.&text=WebStorm%20обеспечивает%20автодополнение%2C%20анализ%20кода,интеграцию%20с%20системами%20управления%20версиями.
7. Визуальная оболочка MongoDb Compass. [Электронный ресурс] – Режим доступа: https://www.hostland.ru/articles/mongodb-compass#:~:text=GUI%20MongoDB%20Compass&text=Это%20удобный%20Клиент%2C%20разработанный%20MongoDB,ОС%20Linux%2C%20Mac%20и%20Windows.
8. StarUml. Описание и как он устроен. [Электронный ресурс] – Режим доступа: https://soware.ru/products/staruml.
9. Как работает Postman. Общие принципы [Электронный ресурс] – Режим доступа: https://blog.skillfactory.ru/glossary/postman/.