

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»**
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК
КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

**ИНФОРМАЦИОННАЯ СИСТЕМА СТУДЕНЧЕСКОГО ГОРОДКА
«НИУ БЕЛГУ»**

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.02 Информационные
системы и технологии
очной формы обучения, группы 07001407
Пугача Максима Юрьевича

Научный руководитель
ст.пр.,
Гуль С.В.

БЕЛГОРОД 2018

РЕФЕРАТ

Информационная система студенческого городка «НИУ БелГУ» – Пугач Максим Юрьевич, выпускная квалификационная работа бакалавра, Белгород, Белгородский государственный национальный исследовательский университет (НИУ «БелГУ»), количество страниц 56, включая приложение 59, количество рисунков 41, количество использованных источников 20.

КЛЮЧЕВЫЕ СЛОВА: информационная система, база данных, веб-приложение, пользовательский интерфейс.

ОБЪЕКТ ИССЛЕДОВАНИЯ: процессы обработки и хранения данных для студенческого городка «НИУ БелГУ».

ПРЕДМЕТ ИССЛЕДОВАНИЯ: разработанная информационная система (ИС) для заведующих общежитиями в студенческом городке «НИУ БелГУ».

ЦЕЛЬ РАБОТЫ: упростить работу управляющих общежитиями по хранению и систематизации данных студенческого городка «НИУ БелГУ».

ЗАДАЧИ ИССЛЕДОВАНИЯ: провести анализ предметной области, рассмотреть существующие ИС студенческого городка, выбрать средства для разработки веб-приложения, разработать модели для проектирования, программно реализовать разработанные модели, провести тестирование разработанной ИС.

ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ: произведен анализ предметной области; ознакомлен с ИС студенческого городка; разработана и протестирована ИС студенческого городка «НИУ БелГУ».

СОДЕРЖАНИЕ

| | |
|--------------------------------------------------------------------------|----|
| ВВЕДЕНИЕ | 4 |
| 1.1 Анализ выбранной предметной области | 6 |
| 1.2 Сравнение ИС..... | 6 |
| 1.3 Структура информационной системы | 8 |
| 1.4 Средства разработки..... | 13 |
| 1.5 Базы данных | 16 |
| 1.6 Способ доступа к базе данных | 18 |
| 2 Проектирование ИС | 19 |
| 2.1 Формирование требований | 19 |
| 2.1.1 Требования к функциональности..... | 19 |
| 2.1.2 Требования к интерфейсу..... | 20 |
| 2.1.3 Технические требования..... | 20 |
| 2.2 Проектирование функциональной части приложения..... | 20 |
| 2.3 Проектирование базы данных | 27 |
| 3 Программная реализация информационной системы | 32 |
| 3.1 Разработка и тестирование ИС студенческого городка «НИУ БелГУ» | 32 |
| ЗАКЛЮЧЕНИЕ..... | 50 |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 52 |
| ПРИЛОЖЕНИЕ А..... | 55 |

ВВЕДЕНИЕ

В современном мире, в любой организации особое значение имеет человек, который управляет определённым видом деятельности, в том числе и общежитием. Управляющий общежитием – очень ответственная профессия, которая связана с обработкой большого количества данных.

Администрация большинства общежитий в настоящее время для управления поступающей информацией по-прежнему пользуется старым способом – записью на бумажных носителях, которая в связи с большим потоком данных часто теряет свою актуальность, еще не успев дойти до адресата. Передача необходимых документов через электронный почтовый ящик увеличивает скорость получения актуальной информации, но при этом дальнейший поиск этой информации в почтовом ящике или на компьютере пользователя не всегда удобен. Наличие специализированной информационной системы и базы данных, в которой хранилась бы вся информация, необходимая администрации общежитий, значительно упростила бы и ускорила обмен информацией и обработку данных.

Принимая во внимание все вышесказанное, целью данной бакалаврской работы является разработка информационной системы студенческого городка «НИУ БелГУ», при помощи которой будет происходить взаимодействие управляющего с различными информационными аспектами его деятельности.

Для достижения поставленной цели были выделены и выполнены следующие задачи:

- анализ предметной области и выбор средств, для разработки веб-приложения;
- проектирование информационной системы;
- реализация проекта с помощью программных средств.

Основная задача, проектируемой информационной системы является упрощенная и автоматизированная работа с информационной базой общежития.

Данная информационная система будет использоваться потенциальными управленцами общежитий. Вход в систему будет производиться под своими данными, которые им выдал главный администратор приложения, а в дальнейшем после авторизации пользователя непосредственно в самом приложении.

В качестве направлений информационной системы можно выделить такие аспекты:

- возможность отправки сообщений и файлов в реальном времени;
- автоматическая отправка сообщений о задолженностях за проживание на почту;
- автоматическое напоминание о запланированных мероприятиях при входе в приложение;
- удобство работы с системой;
- отправка заявок ремонтным служащим для уведомления о назначенной работе.

Заведующий общежитием получит возможность управления, всеми аспектами общежития, не отходя от своего рабочего места. Это позволит заведующему освободить больше времени для других немаловажных дел, связанных с общежитием, а также упростить взаимосвязь сотрудников и студентов.

1 Описание информационной системы и выбор инструментальных средств

1.1 Анализ выбранной предметной области

В выпускной квалификационной работе рассматриваются вопросы проектирования и разработки информационной системы студенческого городка «НИУ БелГУ», предназначенного для упрощения работы заведующих общежитиями. Данная информационная система предназначена для безопасного хранения, большого объема данных со всей введенной информацией.

Цель работы – разработать веб-приложение, позволяющее сократить время работы на составление различного вида отчетов, быстрого поиска нужных данных и сделать работу управляющего общежитием более эффективным храня все данные в единой базе.

Система должна иметь удобный и многофункциональный интерфейс, который можно легко понять и быстро освоить.

1.2 Сравнение ИС

Сравнение информационных систем может производиться по финансовым и не финансовым показателям.

Если рассматривать финансовые показатели, то можно выделить такие аспекты, как затраты на обучение, покупку лицензий и покупку технических средств. При внедрении информационной системы в организацию или учреждение стоит необходимость обучения кадров, так как все новейшие информационные системы переполнены различными функциями, в которых необходимо разбираться. Также это сильно усложняет структуру пользовательского интерфейса. Но как показывает практика, такое обилие функций не всегда нужно пользователю. В разработанной информационной

системе собраны только необходимые функции. Это позволяет поддерживать интуитивно понятный и доступный интерфейс, что в свою очередь делает необходимость обучения кадров только формальностью и от неё можно отказаться.

Так же пользователю не придётся платить за лицензионный продукт для обеспечения непрерывной работы.

При просмотре не финансовых показателей обычно учитывают такие критерии: функциональность, удобство и привлекательность пользовательского интерфейса, быстродействие системы в условиях, приближенных к реальному использованию, возможность доработки, то есть возможность добавления в код программы новых функций.

Что касается функциональности, разработанная программа не отличается какими-то сильными нововведениями от всех современных приложений, но в ней содержатся все необходимые для управляющего общежитием функции. Они выбраны с учётом пожеланий заведующего и администратора общежития. Такая минимизация количества функций позволяет не загромождать программу и использовать только действительно нужное в работе.

Если рассматривать уникальность каждой функции в отдельности, то в разработанной мною системе имеются такие функции, которых нет в других информационных системах такого вида. Например, такие: функция автоматической отправки сообщения студентам о задолженности за проживание; функция ведения списка запланированных мероприятий и оповещения о них управляющему; функция отправки сообщений в реальном времени.

Удобство и привлекательность пользовательского интерфейса может показаться не важной частью информационной системы, но это не так. От этого зависит отношение пользователя к программе. В разработанной программе интерфейс подобран, так что бы у пользователя возникало желание

работать и заходить в приложение более удобно. В то же время он не слишком отвлекает и не бросается в глаза.

Все основные кнопки и функции доступны и их не нужно искать. Всё просто, доступно и привлекательно для любого пользователя.

Другие программы подобного рода не всегда отличаются такой привлекательностью и удобством рисунок 1.1.

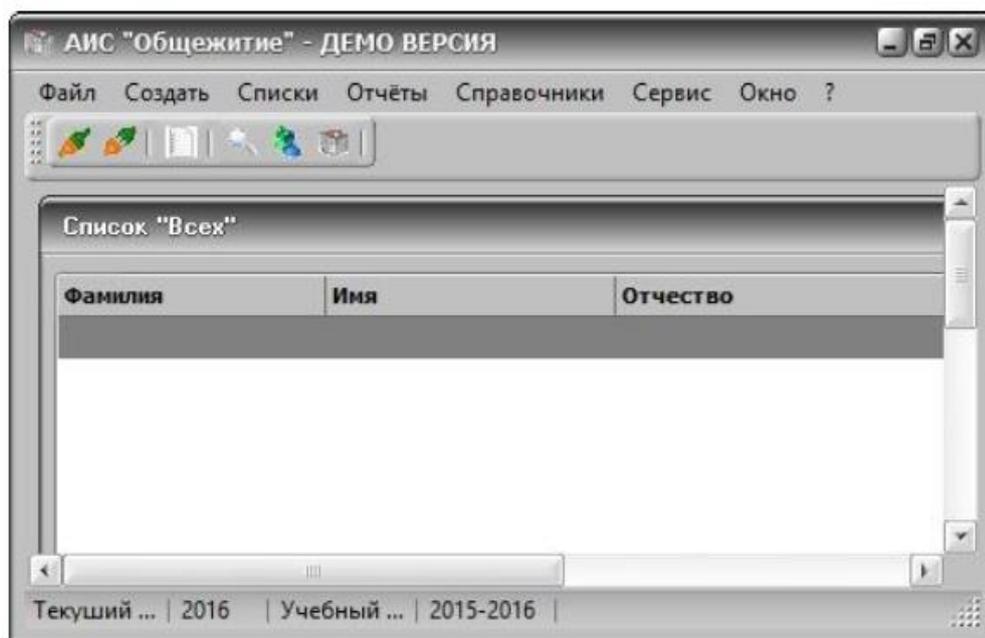


Рисунок 1.1 – Пример интерфейса АИС «Общежитие»

Данная программа не отличается привлекательным интерфейсом. Все функции скрыты во вкладках и их довольно трудно найти. Есть кнопки, которые не никак не обозначены и не ясно какую функцию они выполняют.

1.3 Структура информационной системы

Сама по себе информационная система – это прикладная программная система, целью которой является поиск, сбор, обработка, хранение информации и своевременное обеспечение соответствующих людей данными в рамках некой предметной области [1].

Структура информационных систем состоит из элементов, имеющих взаимосвязь и называемых «подсистемами».

Таким образом, подсистема является частью любой системы, в основе которой лежит общий признак, и которая обеспечивает эффективное функционирование.

Общую структуру подсистем можно проанализировать вне того, в какой предметной области она применяется. Исходя из этого, говорят о структурном признаке классификации, а сами подсистемы обычно называют обеспечивающими. Следовательно, структура любой информационной системы – это не что иное, как совокупность обеспечивающих подсистем рисунок 1.2.

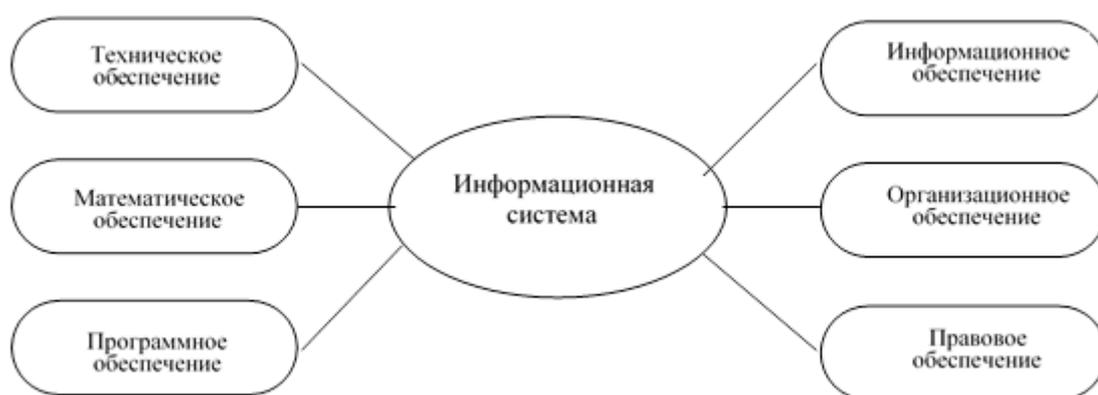


Рисунок 1.2 – Структура информационной системы как совокупность обеспечивающих подсистем

Как правильно, в состав данных подсистем входит техническое, информационное, математическое, программное, организационное и правовое обеспечение [2].

Целью создания подсистем информационного обеспечения является формирование и выдача некоторого объема информации максимально своевременно. Именно благодаря этому принимаются верные и безотлагательные управленческие решения.

Под термином «информационное обеспечение» понимается совокупность системы классификации и кодирования, а также комплекс схем

информационных потоков, унифицированных систем документации и методологий построения баз данных [2].

Чтобы создать информационную систему во внимание нужно брать такие аспекты как: изучение данных, непосредственно циркулирующие в организации; создание базы данных с целью обслуживания запросов, появляющихся в организации.

Маршруты движения информации, ее объемы, места возникновения и использования отражены в схемах потоков информации.

Анализируя данные схемы, можно сформировать меры, способствующие модернизации всей управленческой системы в целом.

Для того чтобы осуществить детальный анализ, необходимо построение схем информационных потоков, которые обеспечивают: устранение повторяющейся и не применённой информации, а также классификацию и целесообразное представление информации.

Существует два последовательных этапа, в которых представлены основные идеи методологии. Первый этап заключается в анализе функциональных отделов организации. Цель данного этапа состоит в том, чтобы познать особенности и структуру деятельности, установить информационные объекты и надлежащий комплекс параметров и характеристик, отображающий свойства и назначение.

Следующий этап основывается на создании информационно-логической модели данных для сферы деятельности, проверенной на начальном этапе. Значимость этого этапа состоит в том, что в модели должны быть зафиксированы и упорядочены все связи между всеми частями и их параметрами.

Основой для создания базы данных и есть данная модель. Требуется наличие многих пунктов, чтобы создать информационное обеспечение. Например, представление всей системы, ее функций, цели и задач; продумывание функционирования информации от зарождения до ее реализации; модернизация комплекса делопроизводства; существование и

применение системы классификации и кадрирования; умение оперировать методикой создания моделей, отражающих взаимосвязь информации в целом; существование технического обеспечения, а именно формирование массивов информации для различного вида носителей.

Под термином «техническое обеспечение» понимается совокупность технических средств, при помощи которых осуществляется бесперебойная работа информационных систем, а также соответствующая документация [2].

Комплекс технических средств, как правило, включает в себя компьютеры; устройства сбора, накопления, обработки, передачи и вывода информации; устройства передачи данных; линии связи; различную оргтехнику и устройства съёма информации, а также эксплуатационную документацию.

Документация позволяет оформлять начальный набор технических средств с их реализацией, процесс обработки данных и технологическое оснащение.

Множество математических моделей, алгоритмов, методов и программ, служащих для выполнения задач информационной системы и бесперебойной работы технических средств, содержит программное и математическое обеспечение [2].

В состав математического обеспечения входят методы математического программирования, статистики и теории массового обслуживания; средства моделирования процессов управления; типовые задачи управления.

Специальные общесистемные программные продукты и техническая документация являются неотъемлемой частью программного обеспечения.

Общесистемное программное обеспечение включает в себя комплексы, ориентированные на пользователей, а также цель которых – решение задач обработки данных. Данное программное обеспечение дает возможность повышать функциональность компьютера и управления процессов обработки данных.

Специальное программное обеспечение – это ничто иное как разработанные при создании информационных систем программы, реализующие модели варьирующейся степени тождественности, которые отражают работу реального объекта [2].

Техническая документация позволяет разрабатывать программные средства и состоит из описания задач и задания на алгоритмизацию, основных примеров и экономико-математической модели.

Организационное обеспечение – это, своего рода, комплекс методов и средств, цель которых – корректировка сотрудничества между работниками, в том числе с помощью технических средств [2].

Все эти процессы совершаются в ходе разработки и реализации информационных систем.

Функциями организационного обеспечения являются: исследование системы управления; выявление задач, подлежащих автоматизации; подготовку задач к решению на компьютере; разработка решений по управлению структурой организации, для увеличения эффективности.

Правовое обеспечение состоит из комплекса правовых норм, которые определяют юридический статус и реализацию информационных систем, регламентирующей установленный порядок получения, обработки и использования данных с целью упрочнения законности.

Правовое обеспечение включает в себя законы и приказы, указы и инструкции и другое. А также состоит из локальной и общей части. Целью общей части является контроль функционирования информационной системы, а локальной – контроль функционирования определённой системы. Правовое обеспечение создания информационной системы включает в себя нормативные акты, связанные с договорными связями разработчика или заказчика и правовым регулированием отклонений от заключенного договора.

Таким образом, правовое обеспечение этапов функционирования состоит из статуса информационной системы, прав, обязанностей персонала,

правового положения видов процесса управления и порядка создания и использования данных.

1.4 Средства разработки

В наше время информационные технологии активно развиваются и всё больше внедряются в различные сферы деятельности человека. Так же развиваются различные средства разработки приложений. Для веб разработок стандартом является такие технологии, как html, css, JavaScript и PHP.

HTML – это язык разметки гипертекста. Данный язык применяется для создания веб-страниц. Он обрабатывается браузером и отображается в виде документа в удобной для человека форме. HTML выступает как средство логической разметки страницы. В HTML существуют теги, которые помогают наделять содержимое страницы определенным смыслом [3].

CSS – это язык описания внешнего вида документа, написанного с использованием языка разметки. Язык CSS предназначен для того, чтобы придавать необходимый внешний вид HTML-документам [3].

Язык JavaScript – это бесплатный язык сценариев, исполняемых на стороне клиента, который позволяет создавать интерактивные HTML-страницы. JavaScript работает на стороне клиента, не используя загрузку сервера [4].

PHP – инструмент, который используется в веб-программировании на стороне сервера. Работа PHP в самом простом варианте сводится к обработке http запроса клиента. Обработка запроса, в свою очередь, заключается в программном формировании гипертекста в зависимости с параметрами запроса, после чего полученная разметка возвращается клиенту [5].

В качестве среды разработки проекта был выбран JetBrains PhpStorm, разработан компанией JetBrains на основе платформы IntelliJ IDEA (рисунок 1.3).

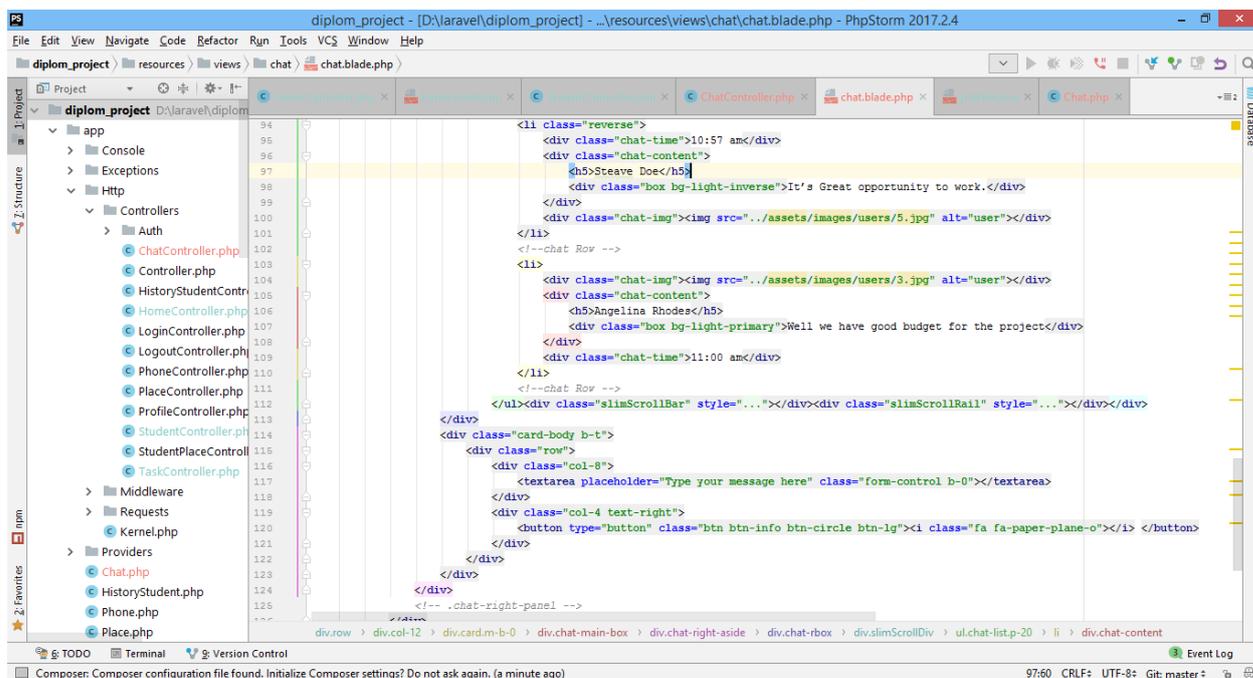


Рисунок 1.3 – Интерфейс PhpStorm

Программное обеспечение JetBrains PhpStorm представляет собой специализированное средство web-разработки, ориентированные на web-приложения и другие виды программ, которые можно создавать с помощью языка PHP и с использованием HTML, JavaScript и CSS. Решение PhpStorm выполняется развертывание и синхронизацию проектов через протокол FTP. Среда PhpStorm предлагает функции автоматического завершения языковых конструкций PHP в коде, инспектирование кода, различные алгоритмы упрощения функций кода и быструю навигацию по коду [6].

Еще одним программным средством разработки стал фреймворк – Laravel. Фреймворк (в информационных системах) – структура программной системы; программное обеспечение, упрощающее процесс разработки и объединения разных компонентов программного проекта. Фреймворк содержит большое количество разных по назначению библиотек, в отличие от обычных библиотек, объединяющих подпрограммы, которые близки по функциональности [7].

Laravel – один из лучших, веб-фреймворк с открытым кодом, предназначенный для разработки с использованием архитектурной модели MVC (англ. Model View Controller – модель – представление – контроллер).

Ключевые особенности, лежащие в основе архитектуры Laravel:

- пакеты;
- Eloquent ORM;
- Логика приложения;
- обратная маршрутизация;
- REST-контроллеры;
- автозагрузка классов;
- составители представлений;
- инверсия управления;
- миграции;
- модульное тестирование (юнит-тесты);
- страничный вывод.

В Laravel существуют пакеты, которые позволяют создавать и подключать модули в формате Composer к приложению на Laravel. Многие дополнительные возможности уже доступны в виде таких модулей. Такая особенность как Eloquent ORM занимается реализацией шаблона проектирования ActiveRecord. Позволяет строго определить отношения между объектами базы данных.

Логика данного фреймворка, создается либо при помощи контроллеров, либо маршрутов (функций-замыканий).

Обратная маршрутизация связывает между собой генерируемые приложением ссылки и маршруты, позволяя изменять последние с автоматическим обновлением связанных ссылок. При создании ссылок с помощью именованных маршрутов Laravel автоматически генерирует конечные URL. Так же существует механизм автоматической подгрузки классов PHP.

Инверсия управления – позволяет получать экземпляры объектов по принципу обратного управления. Также может использоваться для создания и получения объектов-одиночек.

Системами управления версиями для баз данных являются миграции. Они позволяют связывать изменения в коде приложения с изменениями, которые требуется внести в структуру БД, что делает проще развёртывание и обновление приложения.

Для тестирования приложения шаблонов и контроллеров, существует модульное тестирование (юнит-тесты). Тестирование играет очень большую роль в Laravel, который сам по себе содержит большое число тестов для предотвращения регрессий (ошибок вследствие обновления кода или исправления других ошибок) [8].

1.5 Базы данных

Сейчас трудно создать информационную систему, не имеющую базы данных. Совершенствование систем автоматизированной обработки информации всегда было тесно связано с концепциями развития баз данных.

База данных является совокупностью связанных данных, которые организованы некоторыми правилами, предусматривающим общие принципы хранения, управления и описания, независимо от прикладных программ [9].

Реализация структуры базы данных происходит из соображений адекватности описываемого объекта (концептуальный и логический уровень), удобства использования, внедрения, учёта и использования данных (физический уровень).

При разработке проекта использовалась реляционная база данных.

Представленная база данных включает в себе множество связанных таблиц. Каждая из них содержит какую-то информацию об объектах данного вида. Любая строка содержит информацию о единственном объекте, к

примеру, о сотруднике, а столбцы содержат характеристики этих объектов, так называемые атрибуты, такие как: ФИО, номер телефона, дата рождения.

Строки в таблице называют записями. Каждая запись имеет одинаковую структуру. Она состоит из полей – элементов данных, которые содержат атрибуты объекта как показано на рисунке 1.4.

| | id | name | number | dormitory_number | created_at | updated_at |
|---|----|--------------------|-------------|------------------|-------------------|--------------------|
| 1 | 1 | Валентина Петровна | 78413472716 | 1 | 2018-05-31 22:... | 2018-05-31 22:5... |
| 2 | 2 | Слесарь | 79606262814 | 2 | 2018-05-31 22:... | 2018-05-31 22:5... |
| 3 | 3 | Сергей Игоревич | 79511223841 | 3 | 2018-05-31 22:... | 2018-05-31 22:5... |
| 4 | 4 | Плотник | 79517862731 | 1 | 2018-05-31 22:... | 2018-05-31 22:5... |

запись или строка

поле или столбец

Рисунок 1.4 – Название объектов в таблице

В каждом из полей содержится одна характеристика данного объекта [10].

В реляционных базах данных таблицы имеют ряд свойств. Основными, являются следующие: таблица не содержит двух идентичных строк, столбцы распределяются в определённом порядке. Этот порядок задаётся при создании таблицы. В ней обязательно должен быть хотя бы один столбец. Столбец всегда имеет уникальное имя в пределах таблицы. Все значения одного столбца имеют один тип, к примеру, число, текст или дата. На пересечении столбца и строки всегда находится атомарное значение – такое значение, не состоящее из группы значений. Таблицы, которые подходят под это условие, называются нормализованными [11].

Наиболее известными программными продуктами для формирования и управления реляционными базами данных, которые будут использованы приложением, являются Microsoft SQL Server, Microsoft Access и Oracle Database и др. Также существует способ разработки баз данных с помощью

создания зашифрованных файлов, хранящихся на компьютере, где непосредственно и будет содержаться определенная информация.

Преимущество этого способа в том, что установка приложения затрачивает меньшее количество времени за счёт того, что не устанавливается никакого стороннего программного обеспечения. Также при создании самого приложения упрощается реализация связи с данными.

1.6 Способ доступа к базе данных

В большинстве современных программ для доступа к базе данных используется СУБД.

Система управления базами данных, то есть СУБД – это специальное программное обеспечение, при использовании которого пользователи имеют возможность разрабатывать и контролировать базу данных, а также существует возможность осуществления к ней контролируемого доступа [12].

Laravel делает работу с БД чрезвычайно простой и удобной. Если необходимо настроить отдельные подключения для чтения (SELECT) и изменения данных (INSERT, UPDATE, и DELETE), то Laravel позволяет это сделать на одном дыхании. Соответствующее подключение будет автоматически использоваться при работе с БД любым из способов: чистый SQL, конструктор запросов, объектные модели Eloquent ORM.

Так же после настройки подключения к БД можно делать запросы, используя фасад DB. Фасад предоставляет методы для каждого типа запроса: `select`, `update`, `insert`, `delete`, и `statement`.

2 Проектирование ИС

2.1 Формирование требований

В ходе проектирования информационной системы управления общежитием, были сформированы следующие задачи:

- упростить процесс управления общежитием;
- упростить процесс взаимодействия управляющего общежитием с сотрудниками;
- упростить процесс взаимодействия с проживающими в общежитии;
- упростить процесс общения и передачу информации с другими работниками;
- упростить процесс извлечения и нахождения нужной информации.

2.1.1 Требования к функциональности

Требования к функциональности:

- создать инструмент отслеживания оплаты за проживание;
- реализовать «канал» оповещения студентов о задолженности за проживание;
- создать инструмент управления комнатами общежитий (просмотр имущества комнаты, просмотр жильцов комнаты);
- создать в приложении механизм переселения студента в другое общежитие;
- реализовать поиск по полному и частичному совпадению;
- создать инструмент перевода информации из любой таблицы в файл;
- создать инструмент создания плана мероприятий;
- реализовать механизм оповещения о запланированном мероприятии;
- создать телефонный справочник;

- реализовать процесс сохранения истории о действиях над студентами;
- создать средство обмена сообщениями и файлами в реальном времени между заведующими и работниками общежитий;
- реализовать контролируемый доступ в приложение;
- реализовать инструмент для восстановления пароля;
- создать механизм оповещения и вызова сотрудников;
- реализовать инструмент для навигации по приложению.

2.1.2 Требования к интерфейсу

Интерфейс должен удовлетворять таким требованиям:

- должен быть интуитивно понятным;
- с минимальной загруженностью;
- привлекательным;
- эффективным;
- адаптивным.

2.1.3 Технические требования

Технические требования ко всему проекту:

- использовать инструменты для разработки html, css, php и javascript;
- быстрая загрузка и различные действия на странице;
- проверка на введение корректности данных;
- отображение страницы для любых типов экрана.

2.2 Проектирование функциональной части приложения

Зависимо от области применения, информационные системы обычно различаются по своим функциональным особенностям, архитектуре

построения или реализации. Однако они обладают некоторыми общими свойствами. Первая из таких свойств является, любая из информационных систем предназначена для своевременного сбора, обработки и хранения информации. Следствием этого является то, что в основе всех таких систем лежат среды хранения и доступа к данным. Следующая информационная система ориентирована на конечного пользователя, который не обладает достаточной квалификацией или нужным умением в области применяемой вычислительной техники. Исходя из этого, созданное приложение должно обладать простым, наиболее удобным и легко осваиваемым пользовательским интерфейсом. Интерфейс обязан предоставить конечному пользователю всё необходимое для осуществления работы с функциями, но в это же время не должен давать ему возможность выполнять лишние действия [13].

Исходя из вышеописанных свойств информационных систем, помимо программной составляющей, для проекта необходимо:

- спроектированная база данных;
- разработанный графический интерфейс пользователя.

Для раскрытия функций системы была начата разработка диаграммы вариантов использования, также называемая Use-case диаграммой. Данная диаграмма рассматривается как одно из главных средств, для начального этапа моделирования динамики системы. Её используют для подробного выяснения всех требований заказчика к системе и дальнейшей фиксации этих требований в некой форме, позволяющей проводить дальнейшую разработку.

Use-case диаграмма описывает отношения и зависимости между группами вариантов использования и действующими лицами, которые участвуют в процессе [14].

Диаграмма вариантов использования не подходит для отображения проекта и описания внутреннего устройства информационной системы. Она предназначена для упрощения взаимодействия с будущими пользователями и клиентами системы. И особенно она пригодится для определения необходимых и нужных заказчику характеристик создаваемой системы. То

есть, другими словами, Use-case диаграмма говорит, что система должна делать и не указывает на применяемые при этом методы.

Вариант использования создаёт описание группы действий в системе, приводящих к конкретному конечному результату. Делается это с точки зрения действующего лица [14].

Варианты использования – это описания типичных взаимодействий пользователей системы с самой системой. Также они отображают внешний интерфейс всей системы и указывают на форму того, что система обязана сделать [14].

Смысловая архитектура диаграммы вариантов использования заключается в том, что проектируемая информационная система отображается в виде большого количества сущностей или актеров, взаимодействующих с системой при помощи различных вариантов использования.

Актер или действующее лицо – это любая сущность, которая взаимодействует с данной системой извне.

Система предоставляет актёру сервисы, описанием которых, также занимаются варианты использования. Варианты использования системы управления, прежде всего, основываются на определенных видах действий [14].

Действующее лицо – это внешний источник, то есть оно не является элементом системы. Взаимодействие с системой происходит через вариант использования. Действующими лицами не обязательно должны быть реальные люди. Ими могут быть как компьютерные системы, так и любые внешние события.

В результате согласования с представителем организации, была создана диаграмма вариантов использования и показана на рисунке 2.1.



Рисунок 2.1 – Use-case диаграмма использования

Ещё очень важным является то, что действующие лица представляют не физических людей или системы, а всего лишь их роли. Это значит, что при взаимодействии с системой какими-либо способами человек отображается несколькими действующими лицами.

Существует множество методологий моделирования и представления бизнес-процессов. Самые популярные из них IDEF0, IDEF3 и DFD.

Моделирование бизнес-процессов (Business Process Modeling). Больше применение на данный момент имеет стандарт IDEF0, позволяющий описать бизнес-процесс. Предназначение модели в нотации IDEF0 состоит в высокоуровневом описании бизнеса организации в функциональном аспекте [15].

Описание потоков работ (Work Flow Modeling). Основой Стандарта IDEF3 является описание действующих процессов и подобен алгоритмическим методам создания блок-схем [15].

Описание потоков данных (Data Flow Modeling). Существует нотация DFD (Data Flow Diagramming), отображающая очередность работ, выполненных в ходе данного процесса и объем информации, циркулирующей между текущими работами [15].

В качестве методологии функционального проектирования был выбран стандарт IDEF0, так как эта методология сейчас имеет наибольшее распространение и в данном случае она более подходящая.

Стандарт IDEF0 отображает организацию как совокупность модулей. В нём существует правило – наиболее доминирующая функция располагается вверху диаграммы в левом углу, а наименее доминирующая – в правом нижнем. Помимо этого, существует правило стороны:

- стрелка входа;
- стрелка управления;
- стрелка механизма;
- стрелка выхода.

Существует 5 типов стрелок в методологии IDEF0:

- вход;
- управление;
- выход;
- механизм;
- вызов.

На вход поступает множество объектов, которые применяются и преобразуются работой с целью достижения какого-либо результата (выхода). Возможен и тот вариант, что в работе не существует ни одной стрелки, обозначающей вход. Стрелка входа рисуется как примыкающая к левой грани работы.

Стрелка управление – объем информации, регулирующий действия работы. Также управляющие стрелки содержат информацию, указывающую действия для выполнения работе. Любой работе требуется как минимум одна стрелка управления, которая рисуется как примыкающая к верхней грани работы;

На выход следуют те объекты, в которые трансформируются во входы. Любой работе необходимо как минимум одну такую стрелку, которая рисуется как выходящая из правой грани работы;

Механизм это те ресурсы, за счет которых происходит выполнение работы. Данная стрелка представляется как примыкающая в нижнюю грань работы. Изображение стрелок механизма является не обязательным;

Вызов действует стрелка, имеющая особое предназначение – указывать на иную модель работы. Данная стрелка представляется как выходящая из нижней части работы и применяющаяся с целью указания на некоторую работу, которая выполняется вне моделируемой системы [16].

Как уже стало понятно, описание представляет собой «чёрный ящик», содержащий в себе входы, выходы, управление и механизм, который со временем конкретизируется до нужного уровня. Помимо этого, существуют словари описания активностей и стрелок, где даны описания того, какой смысл вложен в заданную активность или стрелку.

В дополнение демонстрируются сигналы управления, которые на DFD не были отображены. Такая модель применяется при создании бизнес-процессов и проектов, базирующихся на моделировании как административных, так и организационных процессов.

При создании информационной системы основной задачей является анализ работы предприятия, для которого создается данная система. Требуется создать модель, которая описывает процесс работы в организации, в соответствии с заданной тематикой. Наряду с этим, модель должна содержать весь спектр нужной информации о функциях бизнес-процессов и принципах работы организации.

Процесс моделирования системы базируется на формировании контекстной диаграммы абстрактного уровня описания системы, точно определяющей элементы моделирования, ее миссию и точки зрения на эту модель (рисунок 2.1).



Рисунок 2.2 – Контекстная диаграмма

Она представляет обобщенное описание системы и взаимное сотрудничество системы и внешней среды.

Данная контекстная диаграмма демонстрирует главный процесс «Информационная система», обозначенный в блоке прямоугольного вида. Входной информацией для данного процесса являются: «Данные о комнатах», «Данные о студентах», «Данные о телефонных номерах», «Данные о мероприятиях», «Данные о хозяйственном учете» и «Данные для поиска данных», вводимые в систему управляющим, полученный результат работы процесса - выходная информация: «Статистика свободных комнат», «Статистика проживающих», «Отчёт об оплате проживания», «Отчёт о хозяйственном учете», «Отчёт о жильцах», «Отчёт о выселенных», «Список запланированных мероприятий», «Отчет о переселенных», «Отчёт о заселенных», «Справочник телефонных номеров», «Список заявок на ремонт». Также были выделены направления «Устав общежития» и «Законодательство РФ», на базе которых осуществляются все выходные процессы, и механизм «Управляющий общежитием», занимающийся основной работой этими процессами.

В конечном счете, всякий фрагмент системы отображается на специальной диаграмме декомпозиции (рисунок 2.3).

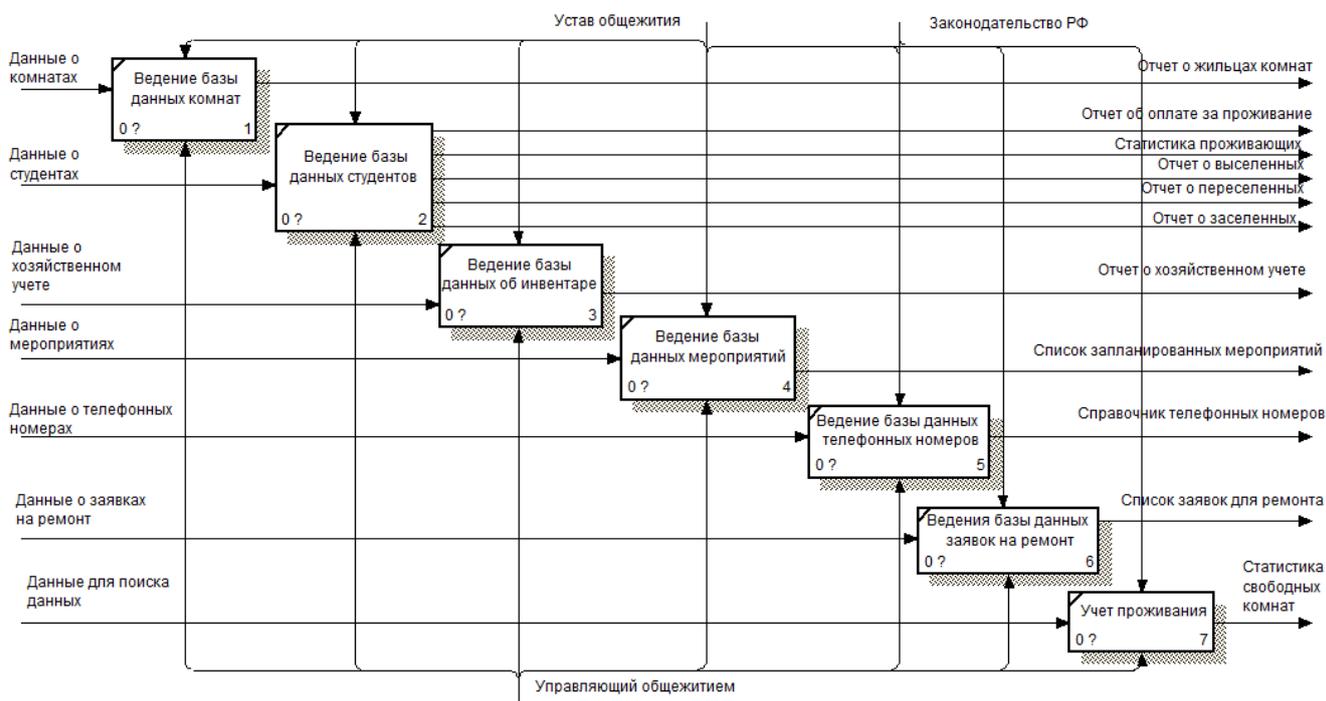


Рисунок 2.3 – Диаграмма декомпозиции

Позднее, как произошло описание контекстной диаграммы, осуществляется функциональная декомпозиция. Она предполагает деление на подсистемы. Далее каждая подсистема описывается индивидуально. Затем, если таковое требуется, происходит деление ещё на более мелкие составляющие и так далее до того момента, пока не будет достигнут нужный уровень подробности.

2.3 Проектирование базы данных

Проектирование базы данных – процесс распознавания данных, их соотношения и размещение итогов в некоторую компьютерную программу. Данное проектирование существует вне зависимости от вида управления.

Этапы проектирования бывают:

- концептуальное проектирование;

- логическое проектирование;
- физическое проектирование.

Концептуальное проектирование это процесс, в котором осуществляется сбор, исследование и редактирование требований, обращенных к данным. Существуют такие мероприятия: исследование предметной области, выявление фрагментов пользовательского представления (каждый характеризуется информационными объектами и взаимосвязями), моделирование и разделение всех представлений. После этого образовывается концептуальную модель, соразмерную структуре базы данных.

Так же существует логическое проектирование это изменение требований к данным в структуры данных. В итоге имеем СУБД-упорядоченную структуру базы данных и спецификации прикладных программ.

Физическое проектирование – определение основных особенностей хранения данных и методов доступа [17].

Правила реляционной модели контролируют организацию информации в таблицах, а также их взаимосвязь.

Для ведения базы и организации поиска требуемой информации в базе должны храниться сведения обо всех студентах, жильцах отдельных комнат, о хозяйственном учете, комнатах, телефонах, заявок на ремонт, имуществе комнат и мероприятиях.

Исходя из сказанного выше, будет создано четырнадцать таблиц, в каждой из которых будет храниться определённая информация, результат можно представить в виде схемы базы данных (рисунок 2.4).

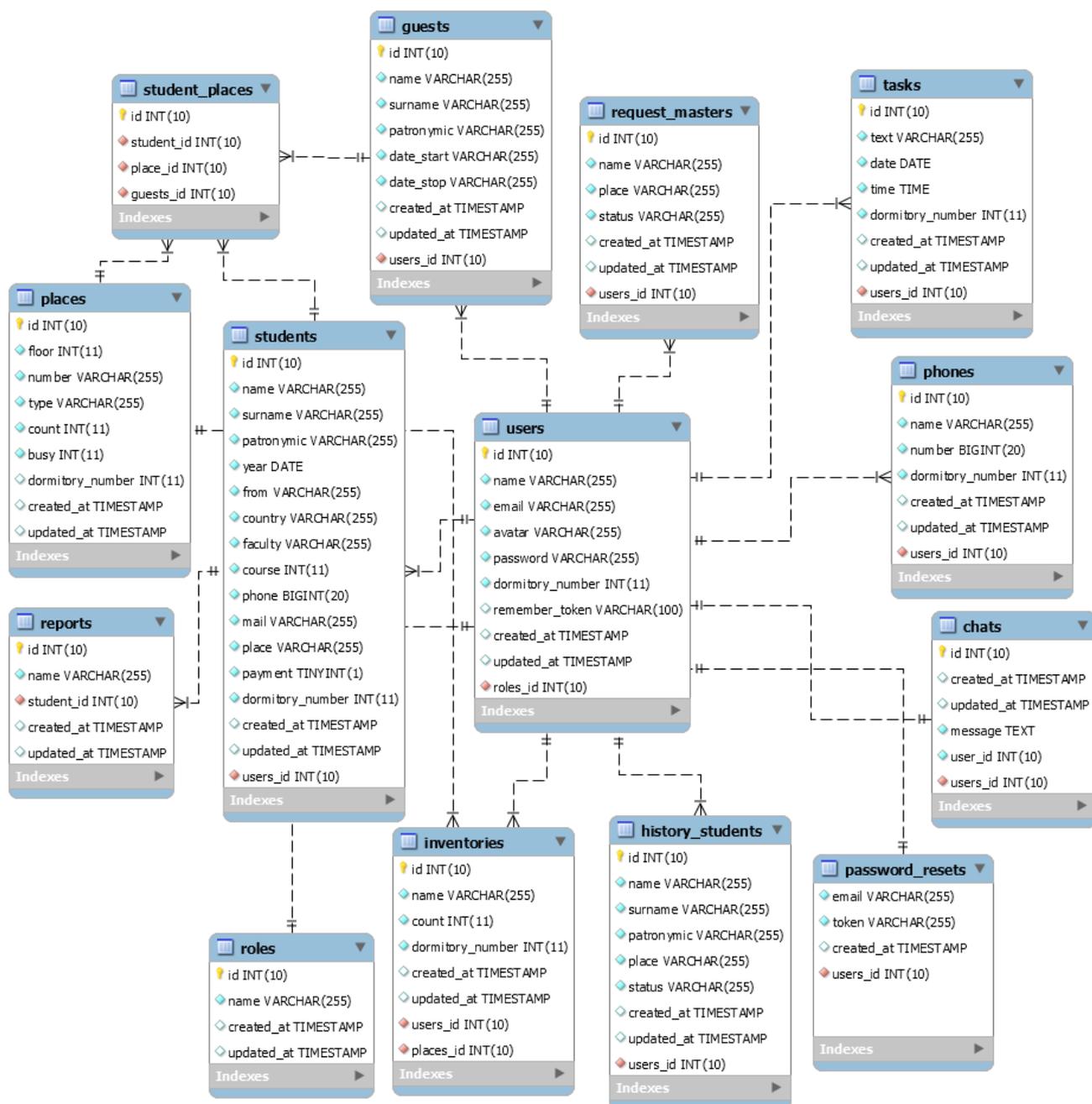


Рисунок 2.4 – Схема базы данных

Первая таблица – это таблица, содержащая всех проживающих в общежитие студентов – «Студенты». В ней, в обязательном порядке, должна быть такая информация, как: ФИО, институт, курс, номер телефона, дата рождения, адрес прописки, электронная почта, номер комнаты, дата заселения, оплата за проживание и номер общежития. Вся информация будет вводиться вручную кроме номера общежития и выбора информации из предложенного списка.

Вторая таблица – это таблица комнат – «Комнаты». В ней будут содержаться номера самих комнат, этаж, тип, вместимость, количество свободных мест и номер общежития. Комнаты будут добавляться вручную. Количество свободных мест в комнате будет высчитываться исходя из вместимости и проживающих в ней людей. Имущество будет добавляться через другую таблицу, таблицу «Имущество». Проживающих можно будет просмотреть исходя из таблицы смежной таблицы «СтудентМесто».

Таблица «Имущество», то есть третья таблица должна содержать наименование имущества, количество и номер общежития. Вся информация вводится вручную кроме номера общежития.

«СтудентМесто» – четвёртая таблица. Она должна содержать номера студентов и номера комнат.

Пятая таблица «Гости». Она содержит поля: ФИО, дата заселения и дата выселения гостя. Информация вводится вручную.

«История студентов» – таблица шесть. Она содержит ФИО, номер комнаты и статус студента. Информация выводится автоматически исходя из действий над студентом.

Таблица номер семь – это таблица «Телефоны». Она должна содержать такую информацию: ФИО или придуманное название телефонной записи, номер телефона и номер общежития. Вся информация вводится вручную кроме колонки номер общежития.

Восьмая таблица – это таблица «Мероприятия». Она должна содержать название мероприятия, номер общежития, дату проведения и конкретное время. Вся информация, так же, как и в предыдущей таблице, будет вводиться вручную.

В девятой таблице «Чат», содержится информационные поля как: Сообщение и номер пользователя.

Десятая таблица «Заявки мастеру». Она содержит название, номер комнаты, номер общежития и статус. Вся информация вводится вручную.

Таблица номер одиннадцать «Объяснительные», таблица содержит поля: название и номер студента. Вся информация вводится вручную.

«Пользователь» – это двенадцатая таблица. Она должна содержать такие поля как: Название или ФИО, почта, колонка фотография пользователя, пароль, номер общежития, запоминающийся номер (для восстановления пароля). Данная информация заполняется главным администратором приложения.

Тринадцатая таблица – это таблица, содержащая все роли пользователей в приложении – «Роли». Она содержит поле название.

Последняя таблица это «Восстановление пароля». Таблица содержит поля: почта и запоминающийся номер.

Таким образом, в практической части была спроектирована данная информационная система студенческого городка. Так же были описаны способы проектирования той или иной задачи для данного приложения и поставлены требования, которым должны придерживаться. Создав все нужные схемы для разработки, можно приступить к реализации самого веб-приложения.

3 Программная реализация информационной системы

3.1 Разработка и тестирование ИС студенческого городка «НИУ БелГУ»

В данной главе приводится описание и правило пользования разработанным приложением.

При входе на сайт открывается страница авторизации (рисунок 3.1).

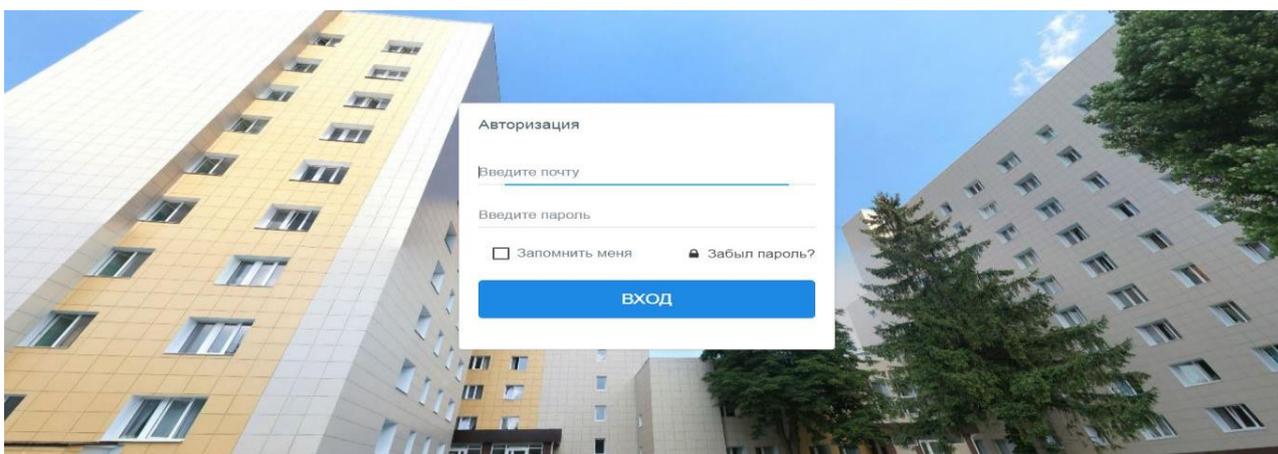


Рисунок 3.1 – Страница «Авторизация»

Чтобы войти на сайт, пользователю необходимо пройти авторизацию. Для этого, в открывшемся окне авторизации предлагается ввести наименование почты пользователя и пароль. В том случае, если пользователь забыл почту или пароль, то он может их восстановить при помощи страницы восстановления пароля (рисунок 3.2).

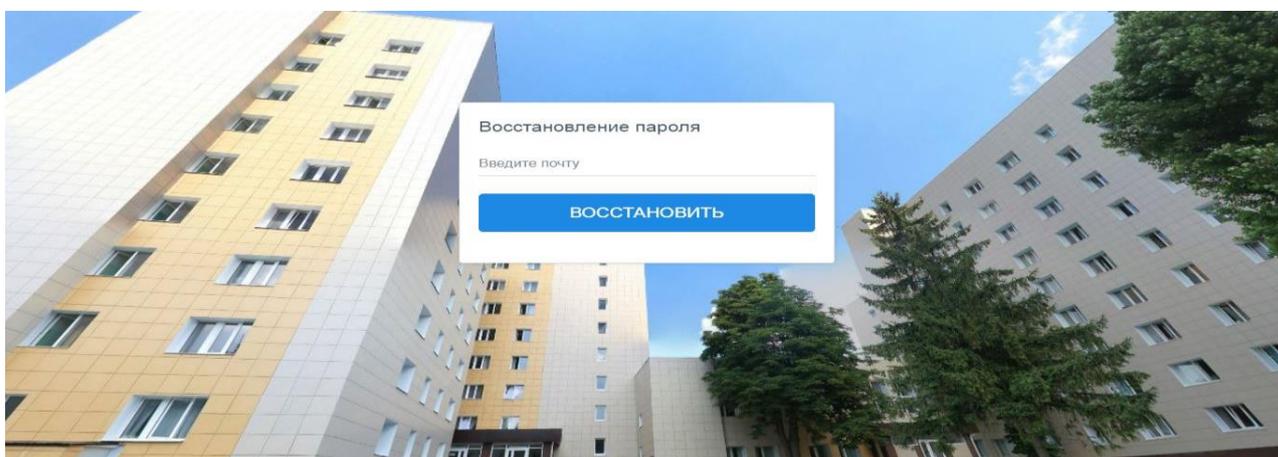


Рисунок 3.2 – Страница «Восстановление пароля»

В появившемся окне «Восстановление пароля» требуется ввести адрес своей электронной почты и нажать кнопку «Восстановить». После чего, на указанную почту пользователю приходит новый пароль. При необходимости, полученный пароль можно поменять в личном кабинете на сайте.

Если данные введены корректно, то происходит переход на главную страницу сайта, показано на рисунке 3.3.

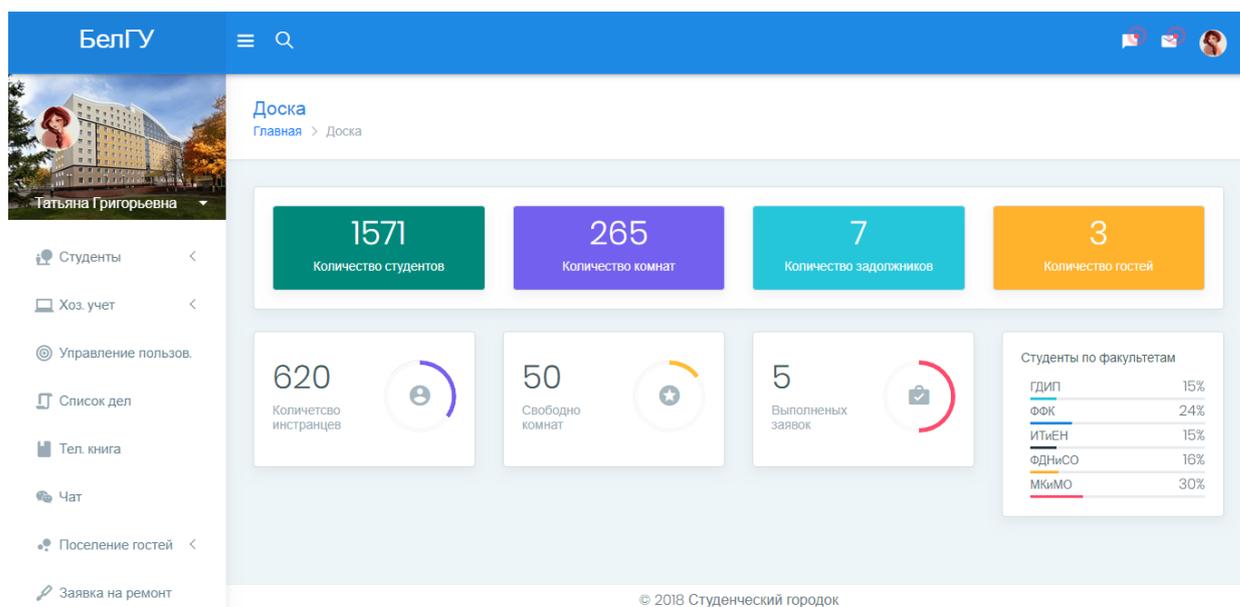


Рисунок 3.3 – Главная страница сайта

Если происходит вход на сайт под данными заведующего общежитием, то на главной странице предоставляется вся необходимая информация:

- количество студентов;
- количество комнат;
- количество задолжников;
- количество гостей;
- количество иностранцев;
- количество свободных комнат;
- количество выполненных заявок;
- статистика по факультету.

На сайте существует удобная система уведомления, касающаяся запланированных мероприятий и новых сообщений (рисунок 3.4).

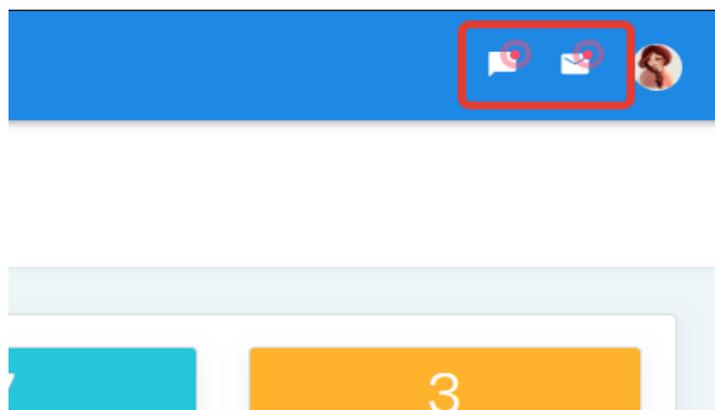


Рисунок 3.4 – Значки уведомлений

Если на данный момент у пользователя есть уведомления, которые он еще не просмотрел, то иконки сверху будут обозначаться красным сигналом. Для того чтобы посмотреть появившиеся уведомления, необходимо на них нажать. При нажатии на левый значок уведомлений появляется список задач на сегодня (рисунок 3.5). В правом значке отображается список непрочитанных сообщений (рисунок 3.6).

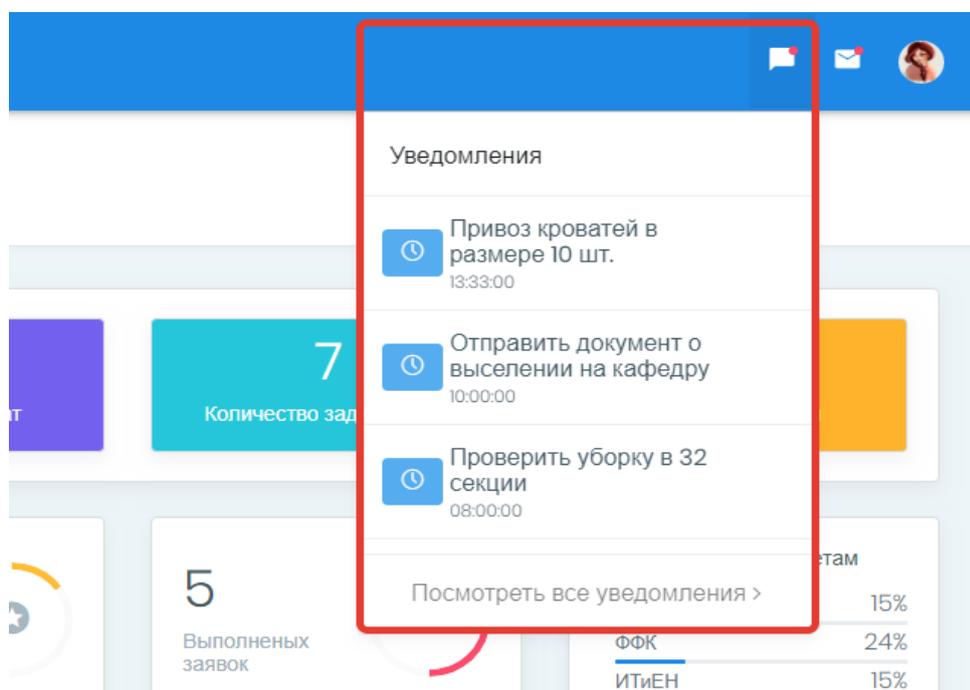


Рисунок 3.5 – Вызванный список мероприятий

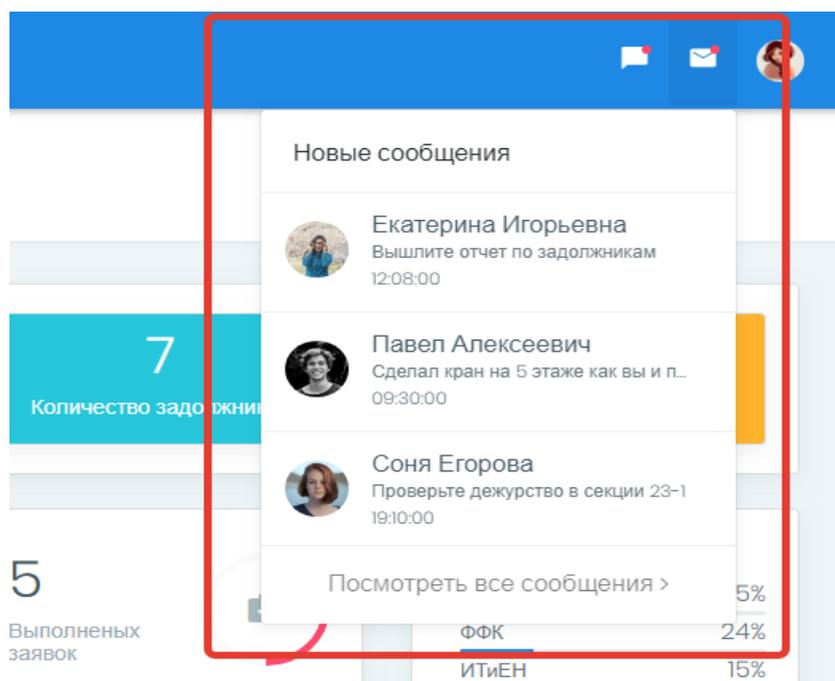


Рисунок 3.6 – Вызванный список новых сообщений

При нажатии кнопки «Посмотреть все уведомления» или «Посмотреть все сообщения», осуществляется переход на подробную информацию. Если уведомление или сообщение требуется удалить из списка, то это можно сделать при помощи одинарного нажатия левой кнопки мыши на нужную запись или сообщение.

С лева на главной странице представлен раздел «Студенты» (рисунок 3.7).

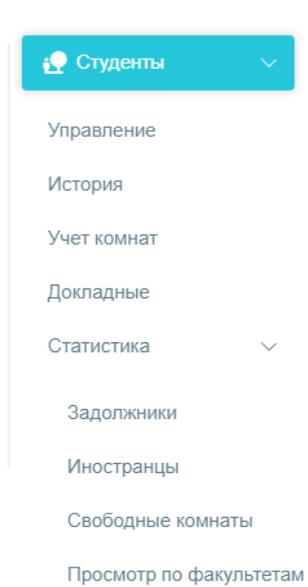


Рисунок 3.7 – Раздел «Студенты»

В раздел «Студенты» входят пять подразделов:

- управление;
- история;
- учет комнат;
- докладные;
- статистика.

В подразделах отображены действия, которыми может воспользоваться пользователь данного веб-приложения.

Подраздел «Управление» приведен на рисунке 3.8.

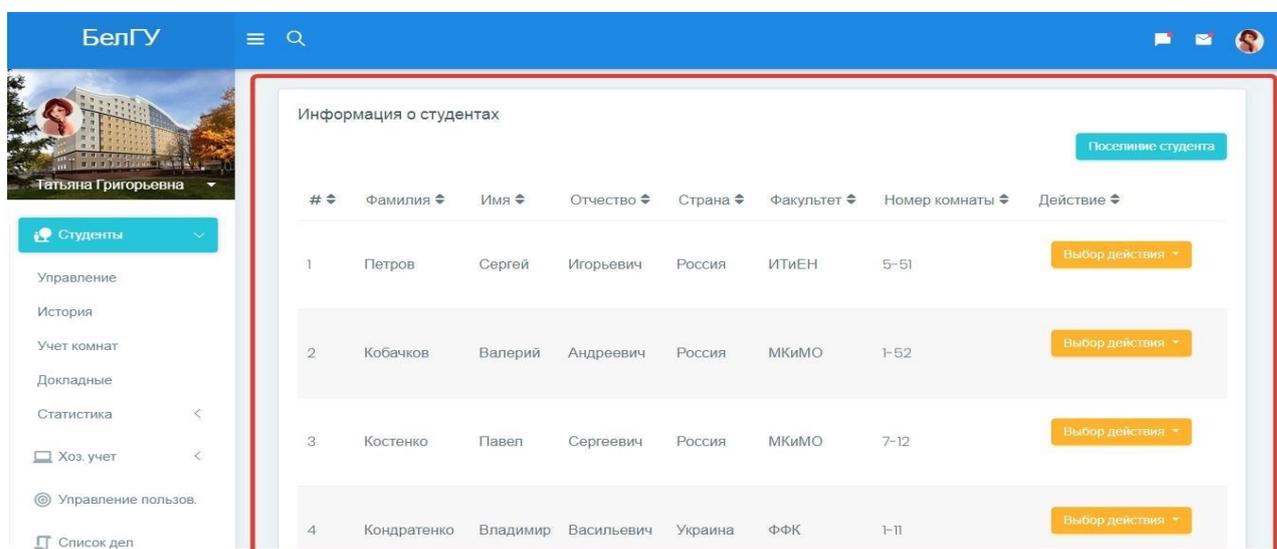


Рисунок 3.8 – Подраздел «Управление»

При выборе подраздела «Управление» отображается список всех проживающих студентов в общежитии, их комнаты и факультеты, на котором они учатся. Справа, напротив каждого студента, имеется кнопка «Выбор действия». Эта кнопка дает возможность совершать различные действия над записями. Если требуется найти в списке конкретного студента, то можно воспользоваться поисковиком, который находится в верхней части страницы, как показано на рисунке 3.9. В поиск вписываем ФИО нужного студента и нажимаем кнопку ввода (рисунок 3.10).



Рисунок 3.9 – Кнопка поиска

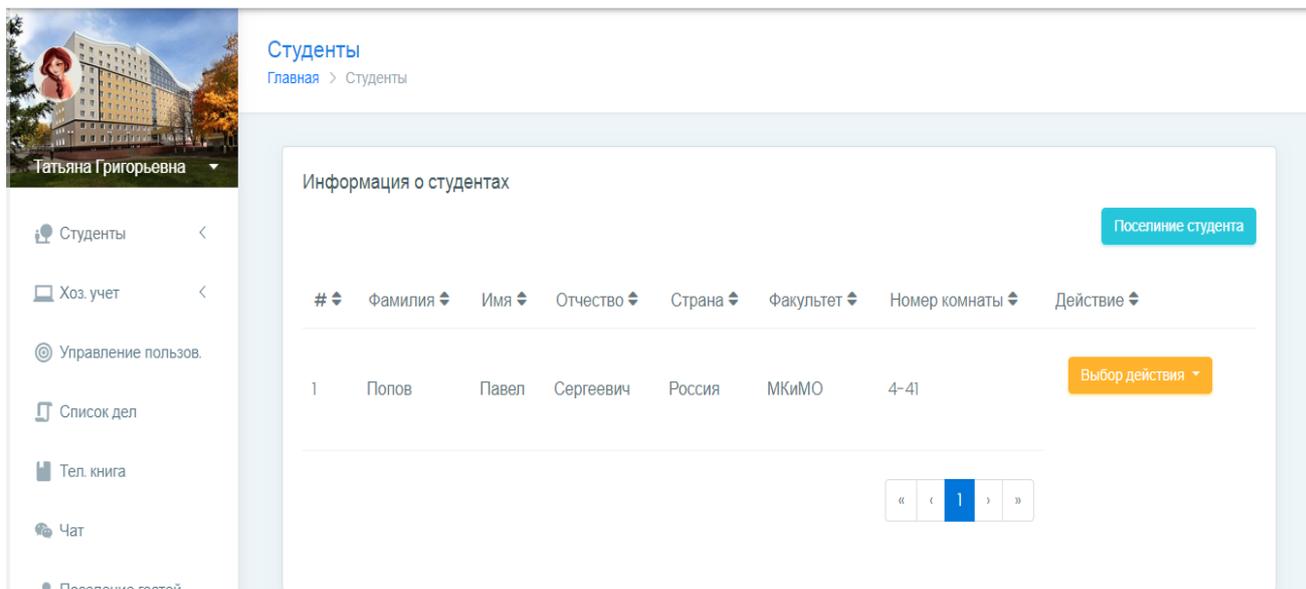


Рисунок 3.10 – Пример поиска

Вся информация для удобства пользователя будет выводиться в подразделе «Управление», где обычно показывается список всех студентов.

Если требуется поселить студента в общежитие, в правом верхнем углу нужно нажать кнопку «Поселение студента» как показано на рисунке 3.11.

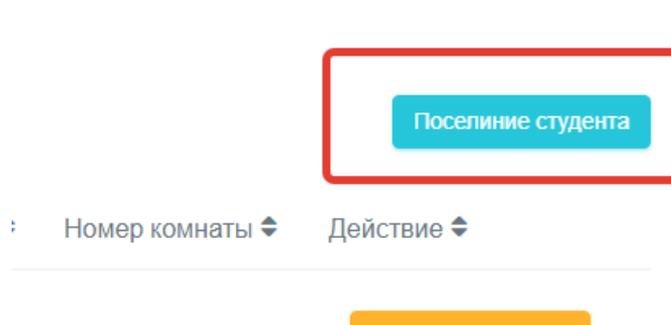


Рисунок 3.11 – кнопка «Поселение студента»

Далее на этой же странице открывается форма заполнения данных студента (рисунок 3.12).

Рисунок 3.12 – Форма заполнения данных студента

Данные студента вводятся удобным способом. В некоторых полях, для упрощения пользования, использован выбор из списка предложенных. Все поля необходимо заполнить точно, так как стоит проверка на корректность данных. Если они будут введены не правильно, то высветятся сведения об ошибках, - в каких полях произошел неправильный ввод.

После заполнения всех данных, нажимаем кнопку «Добавить», и студент будет спешно внесен в список проживающих в общежитии.

В общем списке проживающих в общежитии, напротив каждого студента есть кнопка «Выбор действия». При ее нажатии отражается следующий список действий: изменить, подробнее, выселить, переселить, докладная (рисунок 3.13).

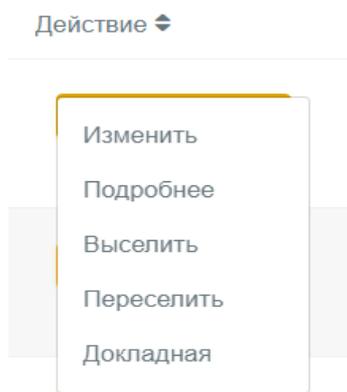


Рисунок 3.13 – Список действий над студентом

При нажатии пункта «Изменить» (рисунок 3.14) выводится окно изменения записи данных студента.

| Изменение записи | | |
|-------------------------------------------|----------------------------------------------------------|------------------------------------------|
| Имя | Фамилия | Отчество |
| <input type="text" value="Сергей"/> | <input type="text" value="Петров"/> | <input type="text" value="Игорьевич"/> |
| Год рождения | Место жительства | Страна |
| <input type="text" value="01.02.1995"/> | <input type="text" value="г. Челябинск ул.Пушкина дом"/> | <input type="text" value="Россия"/> |
| Институт | Курс | Телефон |
| <input type="text" value="ИТиЕН"/> | <input type="text" value="3"/> | <input type="text" value="79517882185"/> |
| Почта | Поселить | Оплата |
| <input type="text" value="serg@mail.ru"/> | <input type="text" value="Этаж: 4 Комната: 5 Мест: 1"/> | <input type="text" value="Оплачено"/> |
| <input type="button" value="Изменить"/> | | |

Рисунок 3.14 – Форма изменения записи

Теперь пользователь может легко изменить данные любого студента.

При выборе действия «Подробнее» выводится данное окно рисунок 3.15.

| Информация о студенте | | | |
|-----------------------------------------|--------------|-----------------------|--------------|
| Имя | Фамилия | Отчество | Год рождения |
| Сергей | Петров | Игорьевич | 1995-02-01 |
| Место жительства | Страна | Факультет | Курс |
| г. Челябинск ул.Пушкина дом 11 кв.21 | Россия | ИТиЕН | 3 |
| Телефон | Почта | Место жительства | Оплата |
| 79517882185 | serg@mail.ru | Этаж: 5 Комната: 5-51 | Оплачено |

Рисунок 3.15 – Форма просмотра записи

В данной форме представлена подробная информация о любом студенте.

При нажатии кнопки «Переселение», появляется окно формы переселения (рисунок 3.16).

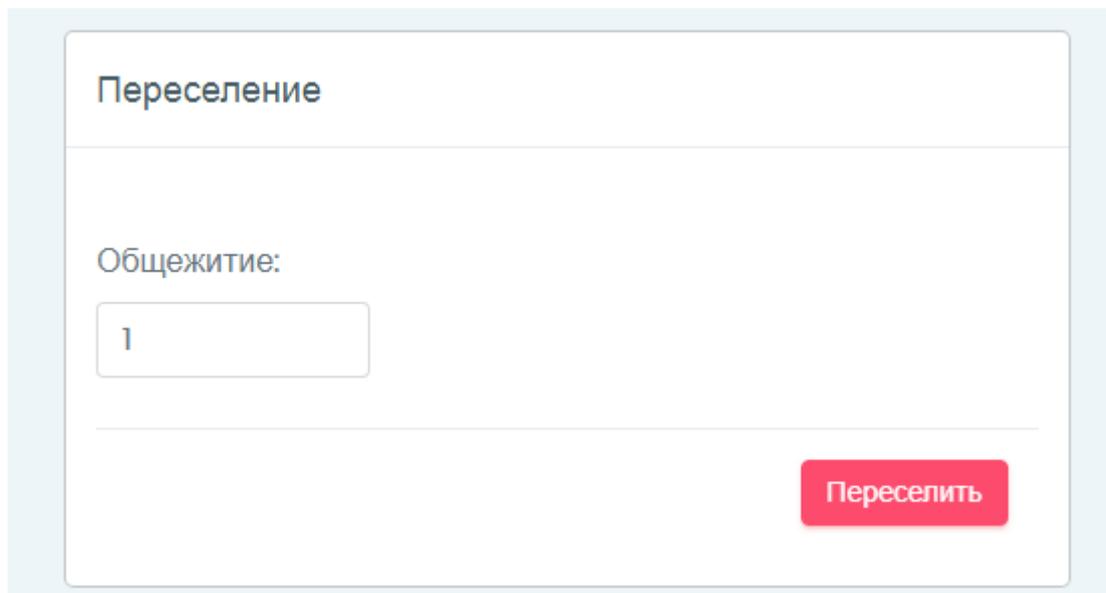


Рисунок 3.16 – Форма для переселения студента

При помощи этой формы можно переселить студента в другое общежитие. Его данные будут автоматически переброшены в список студентов другого общежития, в которое его переселили.

При выборе действия «Докладная» выводится форма, как показано на рисунке 3.17.

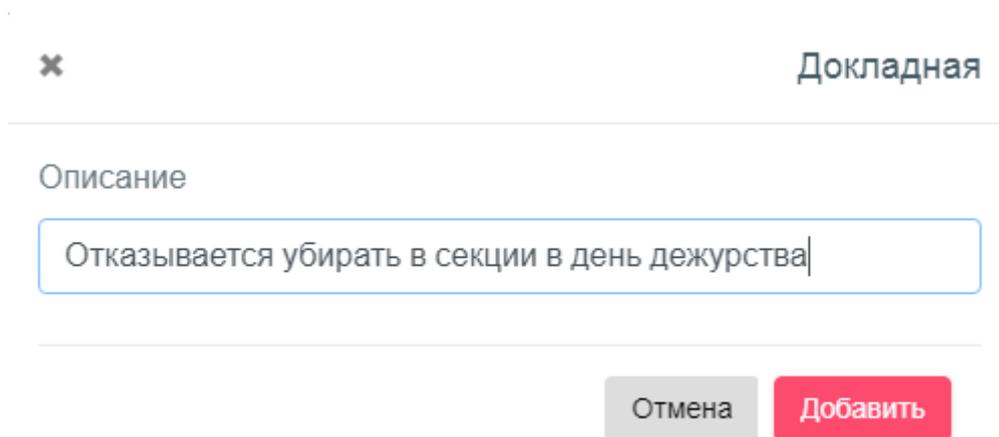


Рисунок 3.17 – Форма для написания докладной

При вводе докладной на студента, информация заносится в главную форму, где можно посмотреть полный список всех докладных в разделе «Докладные» как показан она рисунке 3.18.

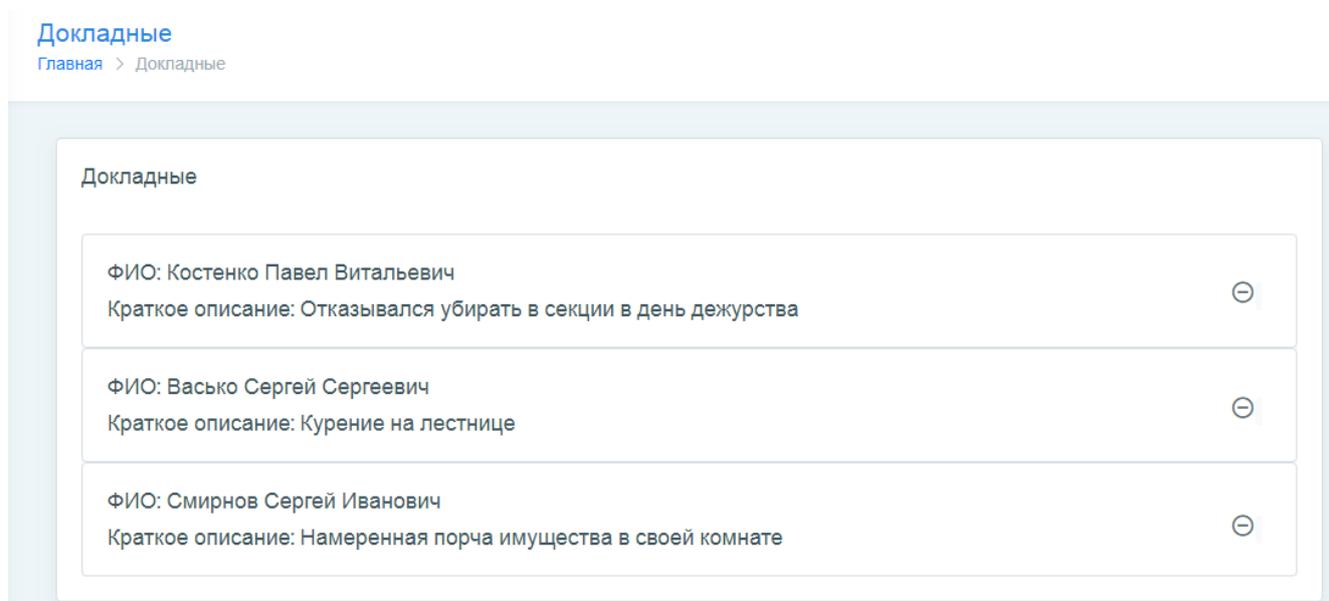


Рисунок 3.18 – Список всех докладных

Данный список легко просматривается. При необходимости, эти данные можно отправить на факультет, или удалить за отсутствием актуальности.

Для того, чтобы выселить студента из общежития, необходимо нажать кнопку «Выселить». После чего выводится статус об успешном удалении из базы данных (рисунок 3.19).

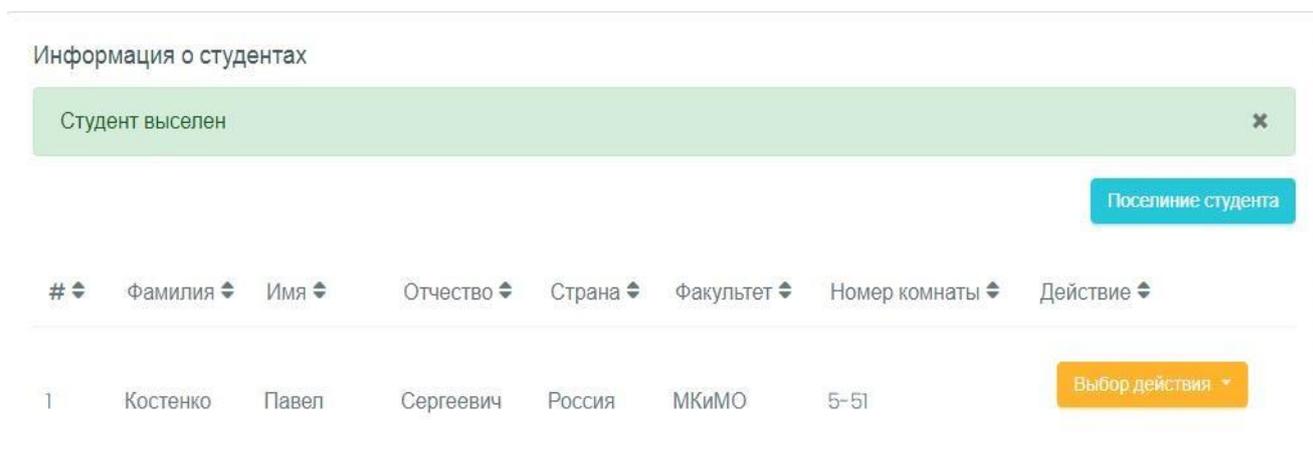


Рисунок 3.19 – Выселение студента

В разделе «История» можно посмотреть список произведенных действий над студентом (рисунок 3.20).

| # | Фамилия | Имя | Отчество | Комната | Статус | Дата |
|---|-------------|----------|------------|---------|-----------|---------------------|
| 1 | Смирнов | Сергей | Иванович | 5-51 | Поселен | 2018-06-04 21:01:16 |
| 2 | Кобачков | Валерий | Андреевич | 1-52 | Выселен | 2018-01-01 18:11:21 |
| 3 | Петров | Сергей | Игорьевич | 3-12 | Выселен | 2018-02-15 01:11:39 |
| 4 | Кондратенко | Владимир | Васильевич | 1-11 | Переселен | 2018-04-04 11:32:57 |

Рисунок 3.20 – Раздел «История»

В разделе «История» подробно отображаются данные, которые мы добавляли, переносили или удаляли из базы данных и дата, когда это происходило.

Для просмотра и манипуляции с комнатами в общежитии необходимо перейти в раздел «Учет комнат», где показаны все имеющиеся комнаты (рисунок 3.21).

| Этаж | Комната | Количество мест | Свободно | Тип |
|------|---------|-----------------|----------|----------|
| 4 | 4-21 | 2 | 1 | Общий |
| 5 | 5-51 | 3 | 1 | Общий |
| 9 | 9-11 | 6 | 0 | Гостевой |

Рисунок 3.21 – Раздел «Учет комнат»

Для того что бы вписать в базу данных комнату, где будут жить студенты или гости, нужно нажать на кнопку «Добавить». Затем указать этаж, комнату, количество мест и тип комнаты (студенты или гости).

В разделе «Статистика» существуют следующие подразделы:

- задолжники;
- иностранцы;
- свободные комнаты;
- просмотр по факультетам.

При нажатии подраздела «Задолжники», вы перейдете на страницу, где будет показан список студентов, у которых имеются какие-либо задолженности по общежитию (рисунок 3.22).

Студенты
Главная > Студенты

Задолжники Сохранить в Excel Добавить запись

| # | Фамилия | Имя | Отчество | Комната |
|---|-------------|----------|------------|---------|
| 1 | Смирнов | Сергей | Иванович | 5-51 |
| 2 | Кобачков | Валерий | Андреевич | 1-52 |
| 3 | Кондратенко | Владимир | Васильевич | 1-11 |

Рисунок 3.22 – Подраздел «Задолжники»

В данном подразделе показаны такие поля как: ФИО и комната. Данные можно добавлять, удалять или сохранить в отчет Excel.

Подраздел «Иностранцы» (рисунок 3.23).

Студенты
Главная > Студенты

Иностранцы Сохранить в Excel

| # | Фамилия | Имя | Отчество | Страна |
|---|---------|----------|------------|---------|
| 1 | Стамби | Арно | Ирман | Панама |
| 2 | Мари | Валерио | Доменикан | Италия |
| 3 | Ламберт | Августин | Аликсandre | Франция |
| 4 | Ревард | Рауль | Бернанд | Франция |

Рисунок 3.23 – Подраздел «Иностранцы»

Данные подраздел показывает список всех иностранцев с данными: ФИО и страна, из которой студент приехал. Данные можно сохранить в отчет Excel.

Подраздел «Свободные комнаты» (рисунок 3.24).

Студенты
Главная > Студенты

| # | Этаж | Номер комнаты | Всего мест | Занято |
|---|------|---------------|------------|--------|
| 1 | 7 | 7-21 | 5 | 4 |
| 2 | 2 | 2-2 | 2 | 1 |
| 3 | 3 | 3-13 | 3 | 1 |
| 4 | 8 | 8-2 | 3 | 2 |

Рисунок 3.24 – Подраздел «Свободные комнаты»

В данном пункте отображаются все свободные комнаты, которыми располагает общежитие: этаж, номер, общее количество мест и сколько занято. Данные можно сохранить в отчет Excel.

Подраздел «Просмотр по факультетам» (рисунок 3.25).

| # | Фамилия | Имя | Отчество | Комната | Факультет |
|---|------------|----------|---------------|---------|-----------|
| 1 | Баскакова | Евдокия | Иосифовна | 4-21 | ФФК |
| 2 | Шувалова | Каролина | Федоровна | 1-12 | ФФК |
| 3 | Бикеев | Аркадий | Артемович | 1-5 | ФФК |
| 4 | Колотушкин | Геннадий | Святославович | 2-8 | ФФК |

Рисунок 3.25 – Подраздел «Просмотр по факультетам»

В данном подразделе используются поля: ФИО, комната и факультет. При помощи списка можно выбирать определенные записи по факультетам и сохранять их в отчете Excel.

Раздел «Хоз. учет», подраздел «Учет по комнатам» представлен на рисунке 3.26.

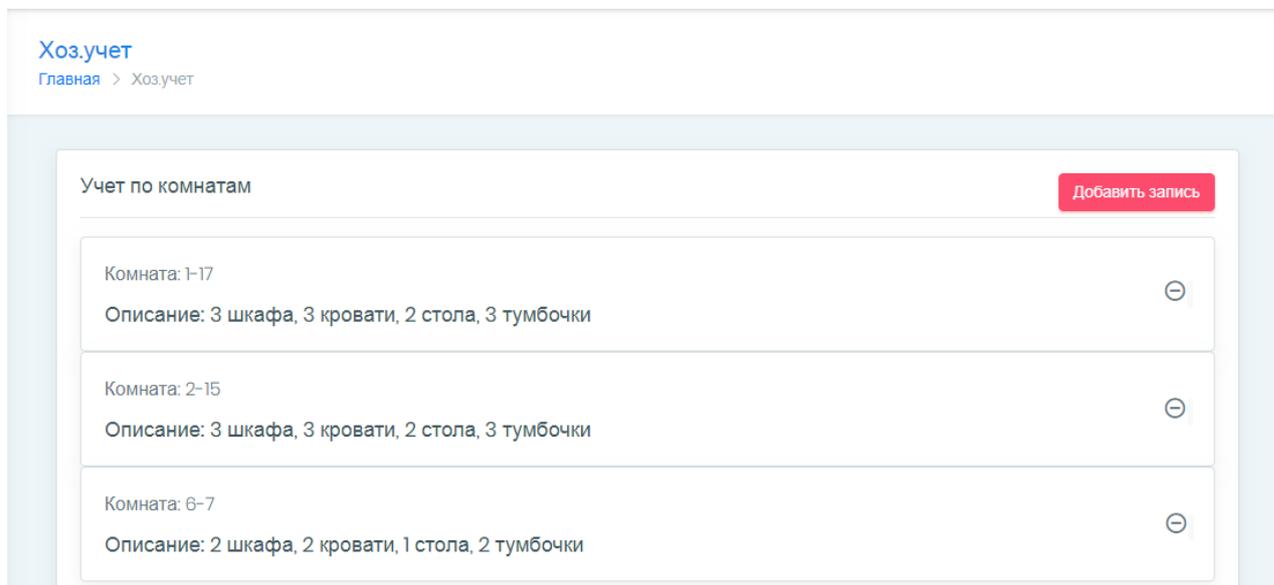


Рисунок 3.26 – Подраздел «Учет по комнатам»

В данном разделе показан весь учет мебели по комнатам, данные можно легко удалять и добавлять с помощью кнопок.

Раздел «Хоз. учет», подраздел «Общий учет» представлен на рисунке 3.27.

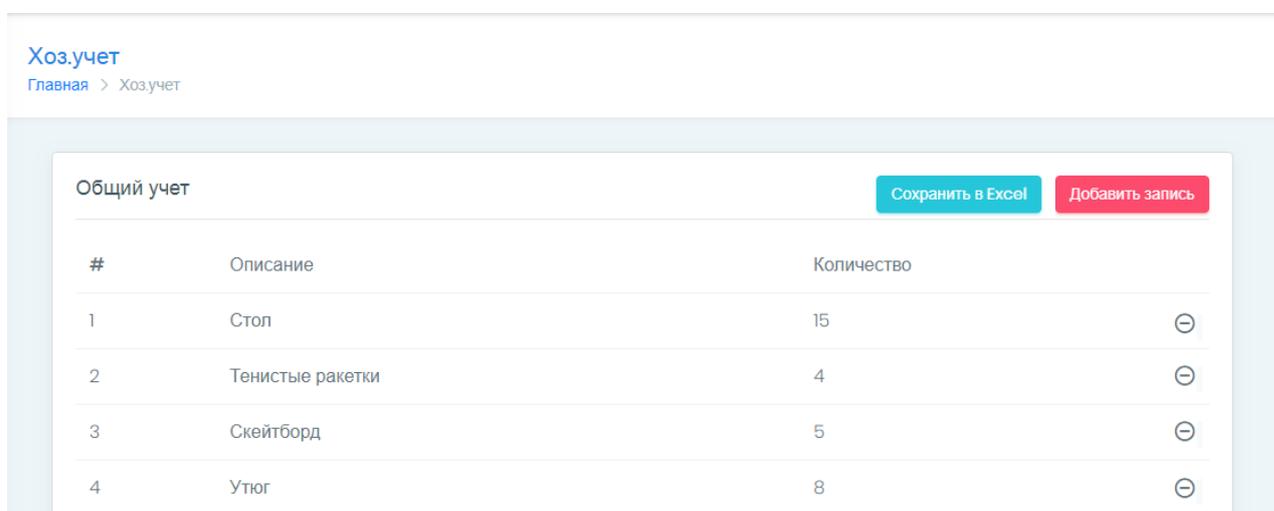


Рисунок 3.27 – Подраздел «Общий учет»

Данный подраздел описывает весь инвентарь, которым располагает общежитие, и в каком количестве он имеется. При необходимости можно добавить инвентарь, нажав розовую кнопку «Добавить запись» в правом верхнем углу. Данные можно удалять, добавлять или сохранять в файле.

Раздел «Список дел» показан на рисунке 3.28.

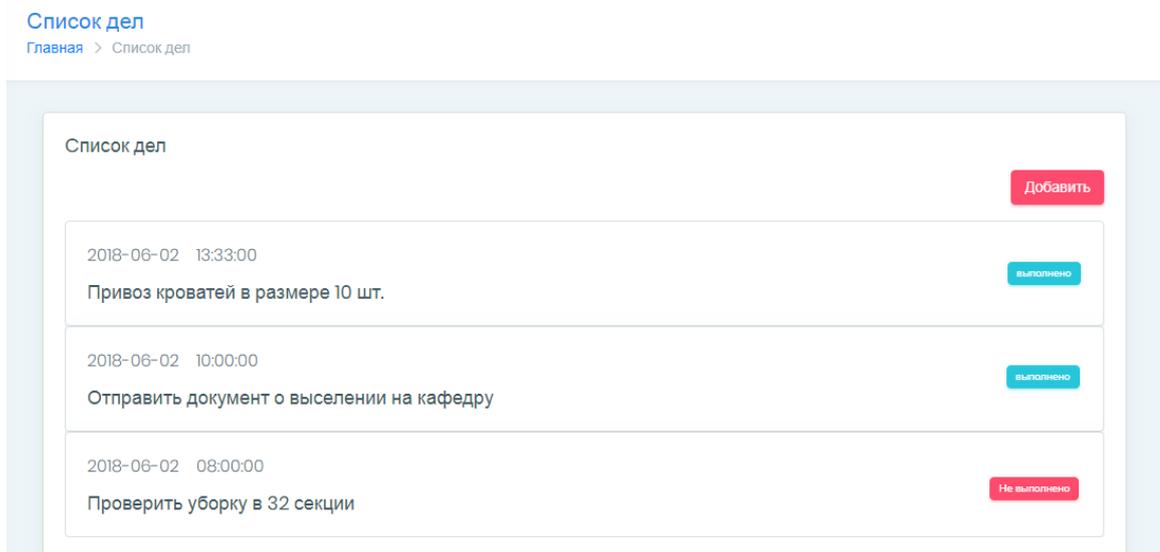


Рисунок 3.28 – Раздел «Список дел»

На данной странице показаны все мероприятия с датой и временем, которые будут происходить в общежитии. При надобности их можно добавить, нажав соответствующую кнопку «Добавить». Для каждой записи так же можно применять статусы «Выполнено» и «Не выполнено» в соответствии со статусом мероприятия.

Раздел «Телефонная книга» показан на рисунке 3.29.

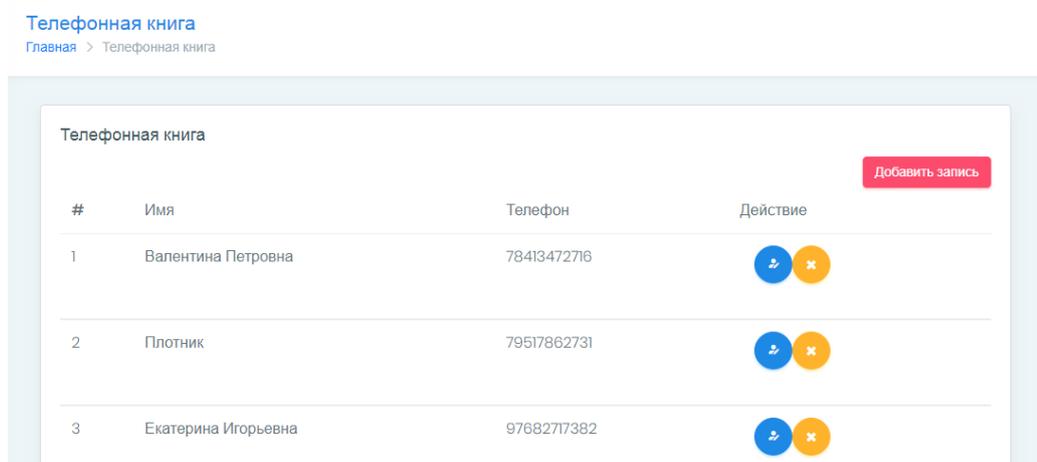


Рисунок 3.29 – Раздел «Телефонная книга»

Раздел отвечает за данные о любых введенных заведующей телефонах. Данные можно изменять, удалять и добавлять, при помощи кнопок, расположенных напротив каждого номера справа.

Раздел «Управления пользователями» показан на рисунке 2.30.

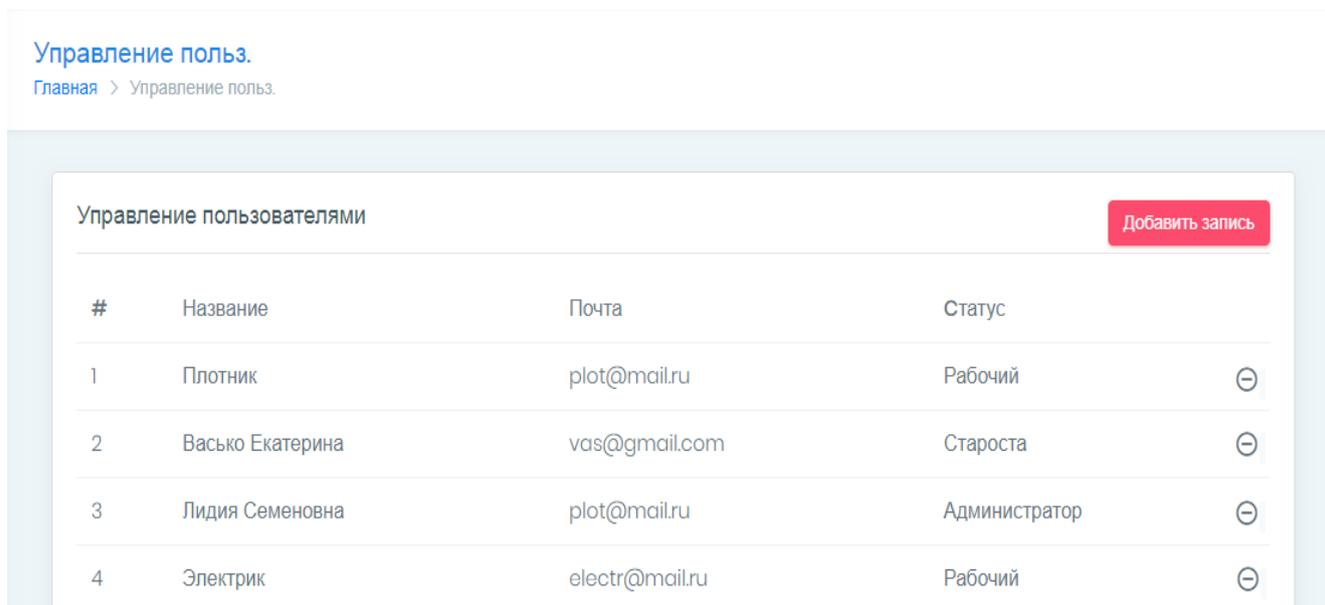


Рисунок 3.30 – Раздел «Управления пользователями»

Данный раздел служит для осуществления действий над пользователями: добавлять и удалять. У каждого пользователя существует своя роль в приложении. У пользователей со статусом «Староста», есть возможность помогать в добавлении записей, а так же использовать чат для общения с администратором или заведующим.

Статус «Рабочий» может просматривать заявки, оставленные заведующим, и использовать чат с администратором, заведующим или другими рабочими.

При использовании статуса «Администратор», можно полностью пользоваться всем функционалом, кроме данного раздела.

Раздел «Чат», показан на рисунке 2.31.

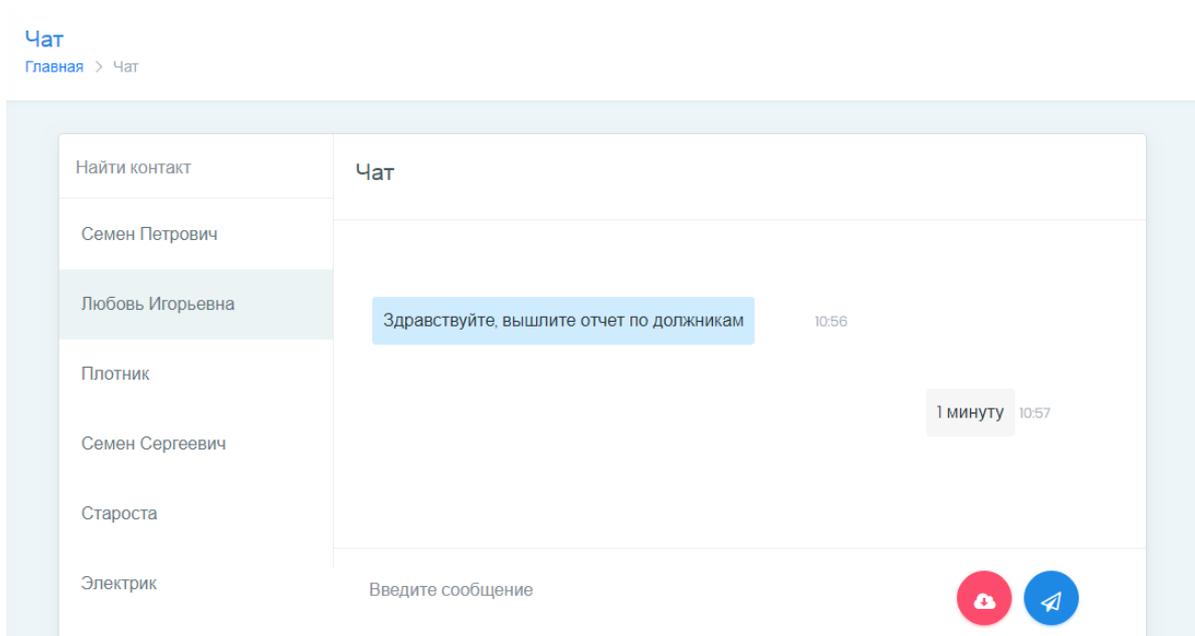


Рисунок 3.31 – Раздел «Чат»

В разделе «Чат» можно общаться и обмениваться данными с пользователями, которые есть в главной базе данных в реальном времени. Для этого нужно слева выбрать пользователя, которому хотите отправить сообщение. В поле «Введите сообщение», написать необходимый текст и нажать соответствующую кнопку ввода.

Раздел «Гости», показан на рисунке 2.32.

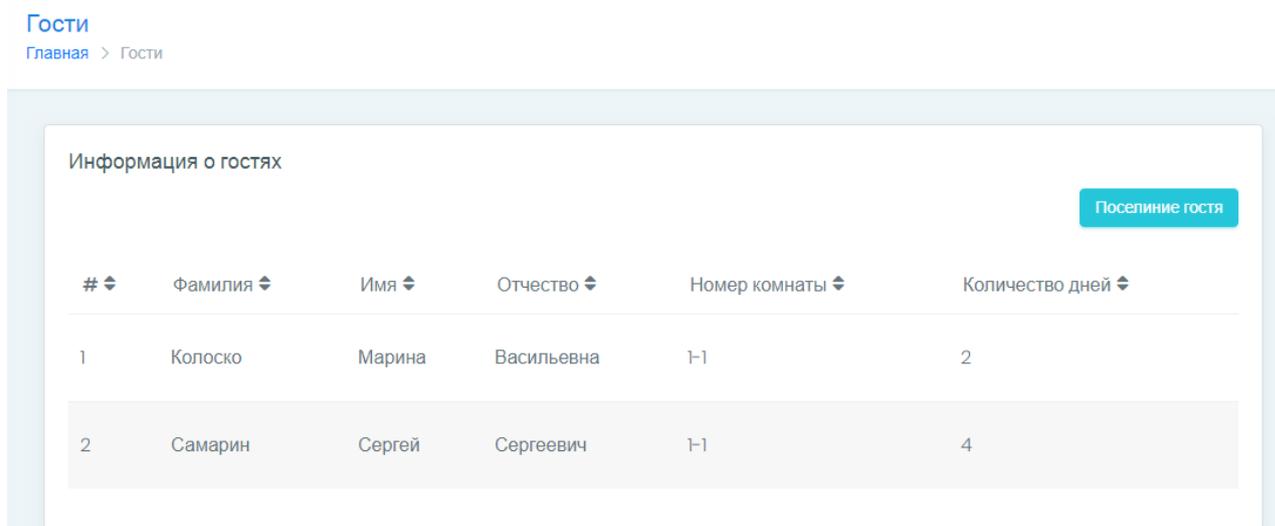


Рисунок 3.32 – Раздел «Гости»

Данный раздел включает информацию обо всех гостях общежития, которые на данный момент снимают комнату. Данные можно добавлять или удалять из базы данных. Поле «Количество дней» считается автоматически, начиная от даты внесения записи в базу данных.

Раздел «Заявки на ремонт», показан на рисунке 2.33.

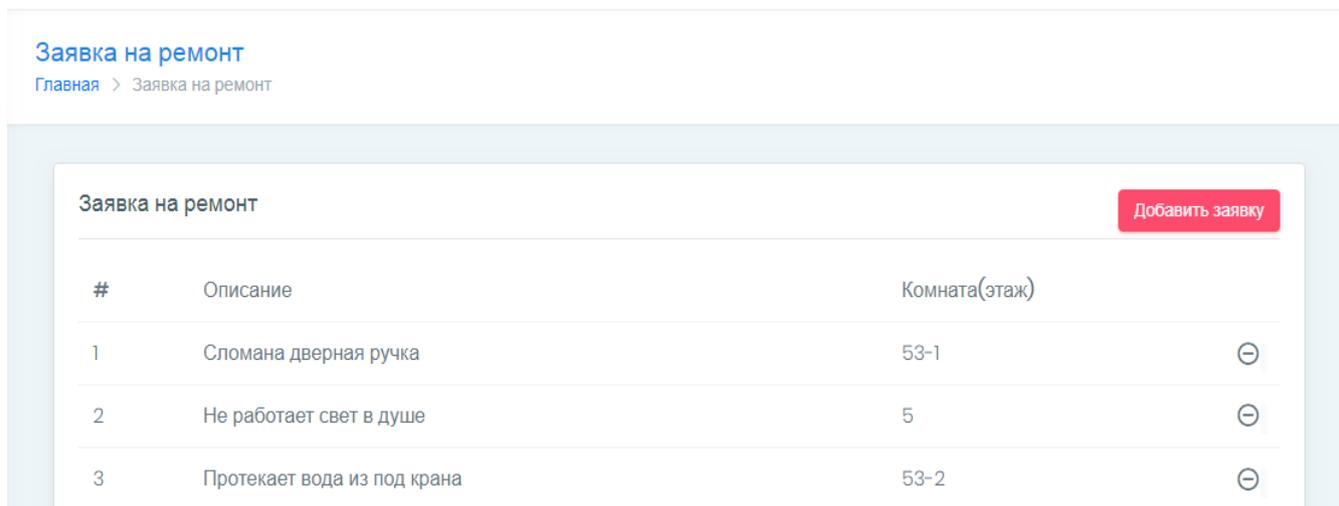


Рисунок 3.33 – Раздел «Заявка на ремонт»

Последний раздел «Заявка на ремонт» служит для внесения в таблицу заявки, которую нужно выполнить пользователю со статусом «Рабочий». Рабочий общежития заходит на сайт под своими данными. В верхнем углу в соответствующей иконке будут появляться новые уведомления в виде заявок, которые нужно выполнить. При переходе в данный раздел отображается полный список заявок на ремонт.

Таким образом, мы рассмотрели основные функции, имеющиеся на данном сайте, которые помогут всем заведующим студенческого городка облегчить управление общежитием и производить систематизацию данных. Профессия заведующей, очень ответственная работа, при этом были выполнены работы по корректному введению данных. Информационная система имеет удобный и многофункциональный интерфейс, который можно легко понять и быстро освоить различного типа пользователей.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была достигнута поставленная цель информационной системы студенческого городка «НИУ БелГУ» для упрощения управления общежитием.

В качестве основных особенностей информационной системы были выделены такие аспекты:

- возможность отправки сообщений и файлов в реальном времени;
- автоматическая отправка сообщений о задолженностях за проживание на почту;
- автоматическое напоминание о запланированных мероприятиях при входе в приложение;
- удобство работы с системой;
- отправка заявок ремонтным служащим для уведомления о назначенной работе.

В данной работе представлен процесс проектирования системы. В начале проектирования системы были поставлены определенные задачи. Их поэтапное решение привело к реализации системы.

При проектировании системы были определены средства разработки. Анализ этих средств помог установить преимущества разработки и определить особенности подобных средств.

При выполнении следующего этапа проектирования системы были сформулированы требования к проекту. Формулировка требований и их утверждение привели к этапу построения системы.

Последним этапом стало описание системы. На этом этапе описывается сценарий действий управляющего при взаимодействии с информационной системой.

Данные задачи были выполнены, а именно:

- проанализирована предметная область, произведен выбор средств, для разработки веб-приложения;

- спроектирована информационная система;
- разработана и протестирована ИС студенческого городка «НИУ БелГУ».

Реализованная информационная система поможет сократить время работы и повысить производительность труда управляющему общежитием.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Свободная энциклопедия «Википедия» [Электронный ресурс] / Информационная система. – Режим доступа: https://ru.wikipedia.org/wiki/Информационная_система. – Дата доступа: 20.04.2018.
2. Студенческий информационный ресурс [Электронный ресурс] / Структура информационной системы. – Режим доступа: http://www.xsieit.ru/download/design_of_information_systems/lectures/873.html – Дата доступа: 25.04.2018.
3. Справочные материалы по веб-разработке. [Электронный ресурс] / Средства веб-разработки. – Режим доступа: <http://codeharmony.ru/materials/125> – Дата доступа: 25.04.2018.
4. Руководства, инструменты и сообщества для разработчиков [Электронный ресурс] / JavaScript основы. – Режим доступа: <https://www.ibm.com/developerworks/ru/library/wa-javascriptstart/index.html>. – Дата доступа: 25.05.2018.
5. Информационный ресурс [Электронный ресурс] / Программирование на PHP. – Режим доступа: <http://codingcraft.ru/php.php>. – Дата доступа: 27.05.2018.
6. Описание продукта [Электронный ресурс] / Описание JetBrains PhpStorm. – Режим доступа: <https://itpro.ua/product/jetbrains-phpstorm/?tab=description>. – Дата доступа: 15.05.2018.
7. Свободная энциклопедия «Википедия» [Электронный ресурс] / Фреймворк. – Режим доступа: <https://ru.wikipedia.org/wiki/Фреймворк/> – Дата доступа: 10.05.2018.
8. Свободная энциклопедия «Википедия» [Электронный ресурс] / Laravel. – Режим доступа: <https://ru.wikipedia.org/wiki/Laravel>. – Дата доступа: 27.05.2018.

9. Национальный Открытый Университет [Электронный ресурс] / База данных и СУБД. Внедрение в SQL. – Режим доступа: <http://www.intuit.ru/studies/courses/42/42/lecture/27193>. – Дата доступа: 28.05.2018.

10. Научная библиотека избранных естественнонаучных изданий [Электронный ресурс] / Проектирование баз данных. – Режим доступа: http://sernam.ru/book_cbd.php?id=2. – Дата доступа: 28.05.2018.

11. Создаём и раскручиваем сайт самостоятельно [Электронный ресурс] / Урок 3. Реляционные базы данных. – Режим доступа: <http://www.site-do.ru/db/db3.php>. – Дата доступа: 10.05.2018.

12. Национальный Открытый Университет [Электронный ресурс] / База данных: модели, разработка, реализация. – Режим доступа: <http://www.intuit.ru/studies/courses/1001/297/lecture/7401>. – Дата доступа: 05.05.2018.

13. Национальный Открытый Университет [Электронный ресурс] / Категории информационных систем. – Режим доступа: <http://www.intuit.ru/studies/courses/1055/271/lecture/6876>. – Дата доступа: 11.05.2018.

14. Ресурс для IT-специалистов [Электронный ресурс] / UML-диаграмма вариантов использования (use case diagram). – Режим доступа: <https://habrahabr.ru/post/47940/>. – Дата доступа: 29.04.2018.

15. Консалтинг и автоматизация [Электронный ресурс] / Основные типы методологий моделирования и анализа бизнес-процессов. – Режим доступа: <http://www.citycg.ru/services/business-process/metody-opisaniya-processov/>. – Дата доступа: 28.04.2018.

16. Справочные материалы по информационным технологиям. [Электронный ресурс] / Методология IDEF0. – Режим доступа: <http://itteach.ru/bpwin/metodologiya-idef0/all-pages>. – Дата доступа: 15.05.2018.

17. Информационный ресурс [Электронный ресурс] / Этапы проектирования БД. Цели и виды работ на этапах концептуального,

логического и физического проектирования. – Режим доступа:
<http://studopedia.org/8-228774.html>. – Дата доступа: 28.06.2018.

18. Репин, В.В. Бизнес-процессы: построение, анализ, регламентация. / В.В. Репин. – М.: РИА «Стандарты и качество», 2007. – 240 с

19. Маркин, А. В. Основы Web-программирования на PHP. / А.В. Репин. – СПб.: Диалог–МИФИ -- , 2015. – 256 с.

20. Симдянов, И.В. MySQL / И.В.Симдянов, М.В. Кузнецов. – Москва. – 2016. –747 с.

ПРИЛОЖЕНИЕ А

Листинг основного класса «Students»

```
<?php
namespace App\Http\Controllers;

use App\HistoryStudent;
use App\Http\Requests\StudentRequest;
use App\Place;
use App\Student;
use App\Task;

class StudentController extends Controller
{
    public function index()
    {
        $students = Student::query()->where('dormitory_number',auth()->user()->dormitory_number)
            ->get();
        $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
            ->get();
        $places = Place::query()->where('dormitory_number',auth()->user()->dormitory_number)
            ->whereColumn('count','>','busy')
            ->get();
        return view('students.index', compact('students','tasks','places'));
    }

    public function show(Student $student)
    {
        $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
            ->get();
        return view('students.show', compact('student','tasks'));
    }

    public function create()
    {
        $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
            ->get();
        return view('students.create',compact('tasks'));
    }

    public function store(StudentRequest $request)
    {
        $data = $request->validated();
        $data['dormitory_number'] = auth()->user()->dormitory_number;
        $students = Student::query()->create($data);
        $place = Place::find($request->get('place'));

        HistoryStudent::query()->create([
            'name' => $request->name,
            'surname' => $request->surname,
            'patronymic' => $request->patronymic,
            'place' => $place->number,
            'status' => 'Поселен'
        ]);

        $students->places()
            ->attach($place);

        Place::query()->where('id',$place->id)
```

```

        ->increment('busy');

return redirect()->route('students.index')
        ->with('status', 'Студент успешно внесен в базу');
}

public function edit($id)
{
    $tasks = Task::query()->get();
    $places = Place::query()->where('dormitory_number', auth()->user()->dormitory_number)
        ->whereColumn('count','>','busy')
        ->get();
    $student = Student::find($id);
    return view('students.edit',compact('student','tasks','places'));
}

public function relocEdit($id)
{
    $tasks = Task::query()->get();
    $student = Student::find($id);
    return view('students.reloc',compact('student','tasks'));
}

public function update($id, StudentRequest $request)
{
    Student::find($id)->update($request->validated());
    return redirect()->route('students.index');
}

public function reloc($id)
{
    $student = Student::find($id);
    HistoryStudent::query()->create([
        'name' => $student->name,
        'surname' => $student->surname,
        'patronymic' => $student->patronymic,
        'place' => $student->place,
        'status' => 'Переселен'
    ]);
    $student->update(['dormitory_number' => request()->dormitory_number]);
    return redirect()->route('students.index');
}

public function destroy($id)
{
    $student = Student::find($id);
    $place = Place::find($student->place);

    HistoryStudent::query()->create([
        'name' => $student->name,
        'surname' => $student->surname,
        'patronymic' => $student->patronymic,
        'place' => $student->place,
        'status' => 'Выселен'
    ]);

    Place::query()->where('id',$place->id)
        ->decrement('busy');

    $student->places()
        ->detach($place);

    $student->delete();
}

```

```

    session()->flash('status', 'Студент выселен');
    return redirect()->back();
}

public function ino()
{
    $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
        ->get();
    return view('statistics.ino',compact('tasks'));
}

public function payment()
{
    $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
        ->get();
    return view('statistics.payment',compact('tasks'));
}

public function placeC()
{
    $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
        ->get();
    return view('statistics.placeC',compact('tasks'));
}

public function searchFaculty()
{
    $tasks = Task::query()->where('dormitory_number',auth()->user()->dormitory_number)
        ->get();
    return view('statistics.searchFaculty',compact('tasks'));
}
}

```


Выпускная квалификационная работа выполнена мной совершенно самостоятельно. Все использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют ссылки на них.

« ____ » _____ Г.

(подпись)

(Ф.И.О.)