|  |  |  |
| --- | --- | --- |
|  |  | УТВЕРЖДАЮ |
|  |  | Руководитель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

ТЕХНИЧЕСКИЙ ПРОЕКТ

«Информационная система для распределения заявок по ремонту в общежитии»

2024

Содержание

[1. Уточнение структур данных 2](#_Toc180239216)

[2. Формы для их представления 5](#_Toc180239217)

[3. Разработка алгоритма решения задачи 11](#_Toc180239218)

# Уточнение структур данных

В «Информационная система для распределения заявок по ремонту в общежитии» будет иметься база данных. База данных в виде логической и физической ER диаграмме изображены на рисунках 1 и 2.

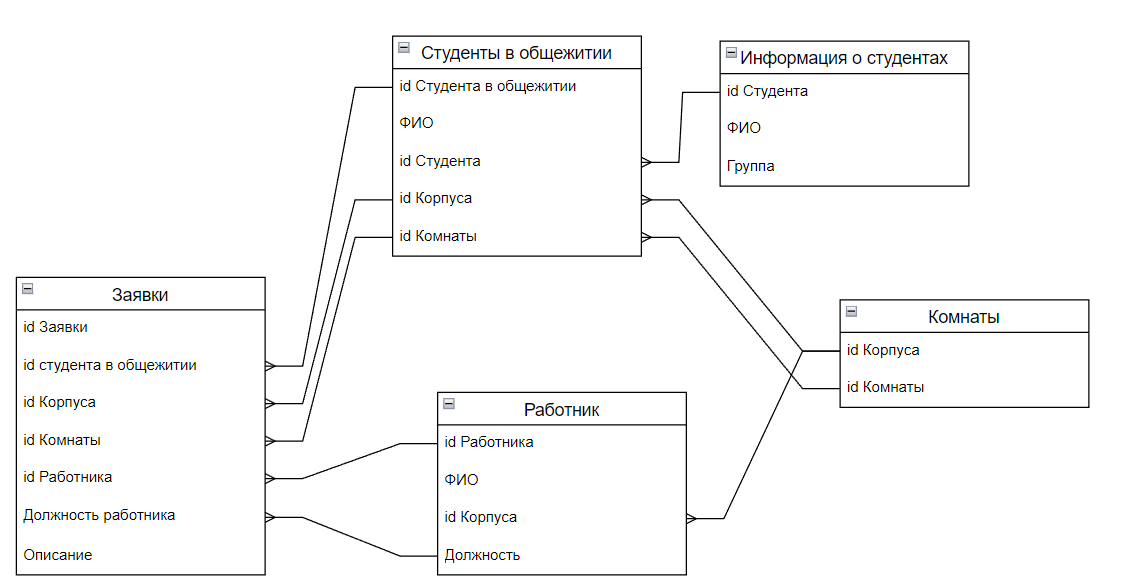


Рисунок 1 – Логическая ER диаграмма базы данных.

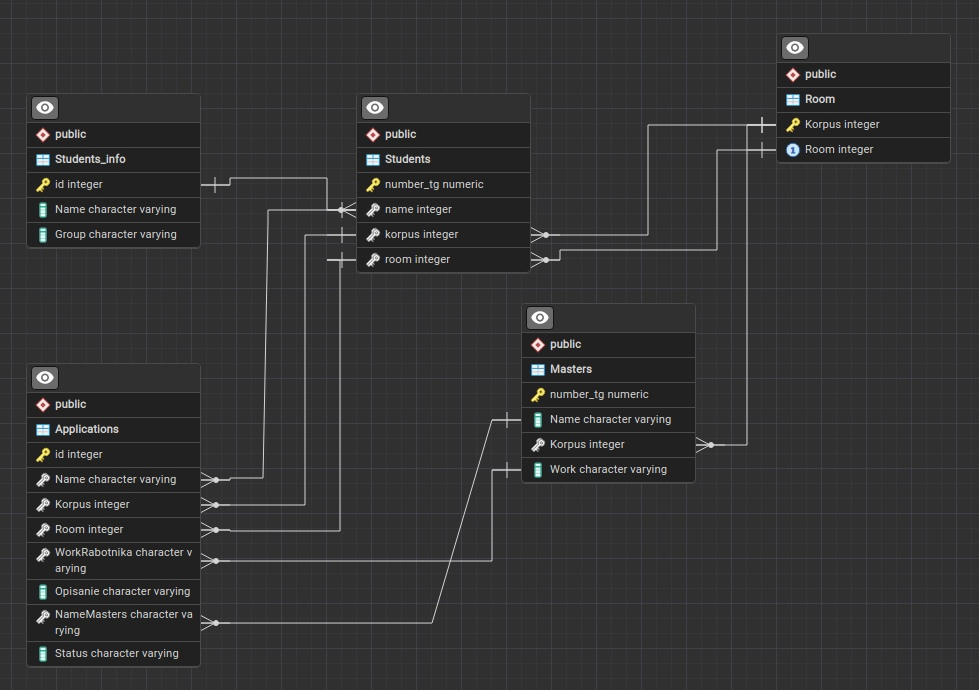


Рисунок 2 – Физическая ER диаграмма базы данных.

На ER диаграммах изображено 5 таблиц: Информация о студентах, Студенты в общежитии, Работники, Заявки, Комнаты.

В таблице «Информация о студентах» - содержатся данные о всех студентах, которые есть в учебном заведении. Таблица содержит следующие данные:

* id Студента;
* Его ФИО;
* Его группа.

В таблице «Студенты общежития» - содержатся данные о всех студентах проживающие в общежитии. Таблица содержит следующие данные:

* id Студента в общежитии;
* Его ФИО;
* Корпус общежития, в котором он живет;
* Комната, в которой он живет.

В таблице «Работники» - содержатся данные о всех работниках общежития. Таблица содержит следующие данные:

* id Работника;
* Его ФИО;
* id корпуса в котором он работает;
* Его должность.

В таблице «Комнаты» - содержатся список всех комнат и корпусов принадлежащие общежитию. Таблица содержит следующие данные:

* id Корпуса;
* id Комнаты.

В таблице «Заявки» - содержатся данные о всех заявка в общежитии. Таблица содержит следующие данные:

* Id заявки;
* id Студента в общежитии;
* id Корпуса;
* id Комнаты;
* id Работника;
* Должность работника;
* Описание заявки.

# Формы для их представления

Прототип экранной формы панели администратора (вкладка студенты) представлен на рисунке 3. На Экранной форме содержатся следующие элементы:

* Вкладка «Студенты», где будут указываться id, ФИО студентов, корпус и комната;
* Кнопка «Добавить» для добавления нового студента;
* Кнопка «Сохранить» для сохранения информации о студенте;
* Кнопка «Удалить» для удаления студента.

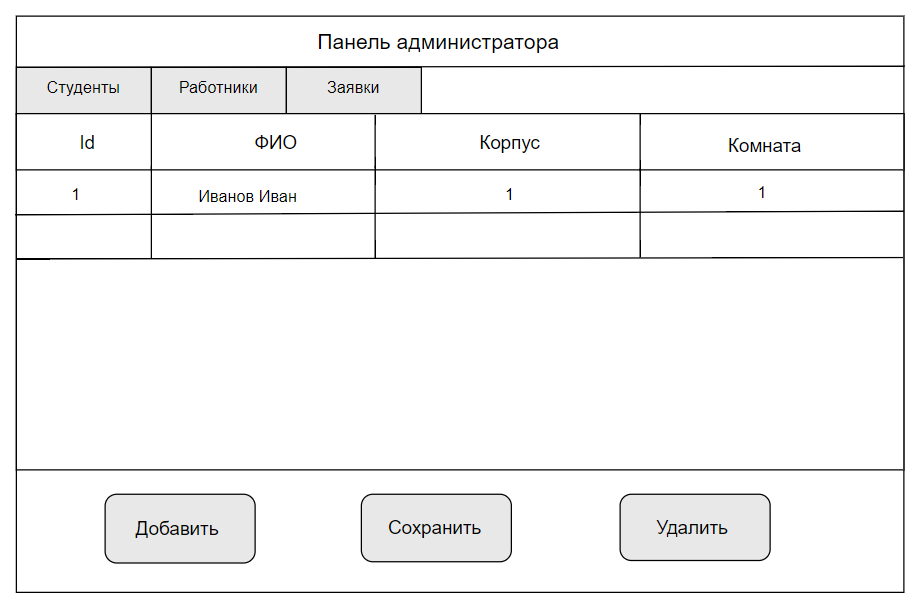


Рисунок 3 - Панель администратора (вкладка студенты).

Прототип экранной формы панели администратора (вкладка работники) представлен на рисунке 4. На Экранной форме содержатся следующие элементы:

* Вкладка «Работники», где будут указываться id, ФИО работника, должность и корпус;
* Кнопка «Добавить» для добавления нового работника;
* Кнопка «Сохранить» для сохранения информации о работнике;
* Кнопка «Удалить» для удаления работника.

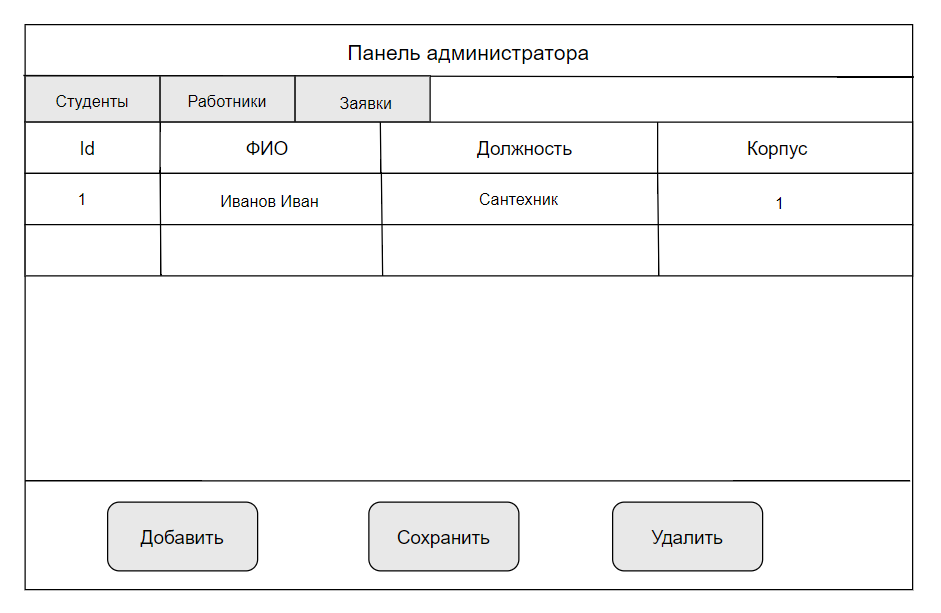


Рисунок 4 - Панель администратора (вкладка работники).

Прототип экранной формы панели администратора (вкладка заявки) представлен на рисунке 5. На Экранной форме содержатся следующие элементы:

* Вкладка «Заявки», где будут указываться: id, ФИО студента, корпус, ФИО работника, должность, а также описание и статус работы;
* Кнопка «Добавить» для добавления новой заявки;
* Кнопка «Сохранить» для сохранения информации о заявке;
* Кнопка «Удалить» для удаления заявки.



Рисунок 5 - Панель администратора (вкладка заявки).

Телеграмм бот «Сотрудники» будет функционировать следующим образом:

* При вводе команды \start бот запускается и будет готов к работе;
* После запуска, бот приветствует пользователя и запрашивает его номер телефона;
* После введения номера телефона, бот благодарит пользователя за его отправку;
* При нажатии на кнопку «Принять», бот принимает заявку;
* При нажатии на кнопку «Отклонить», бот отклоняет заявку и запрашивает причину отклонения;
* После того, как указывается причина, заявка отклоняется по данной причине.

Прототип экранной формы телеграмм бота «Сотрудники» представлен на рисунке 6.

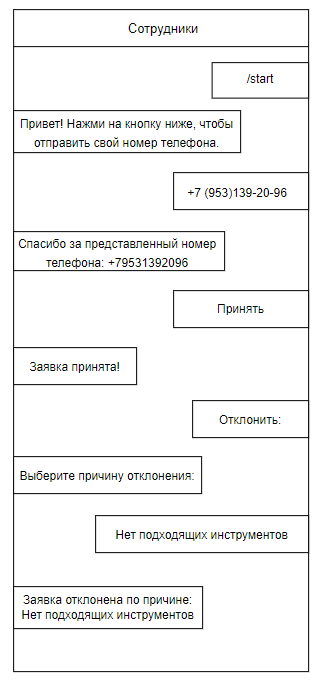


Рисунок 6 - телеграмм бот «Сотрудники»

Телеграмм бот «Студенты» будет функционировать следующим образом:

* При вводе команды \start бот запускается и будет готов к работе;
* После запуска, бот приветствует пользователя и запрашивает его номер телефона;
* После введения номера телефона, бот благодарит пользователя за его отправку;
* При нажатии на кнопку «Заполнить анкету», бот запрашивает должность работника;
* После ввода должности работника, бот выводит данную должность и далее запрашивает описание работы;
* После ввода описания работы, бот также выводит работу и благодарит пользователя за заполнение анкеты.

Прототип экранной формы представлен на рисунке 7.

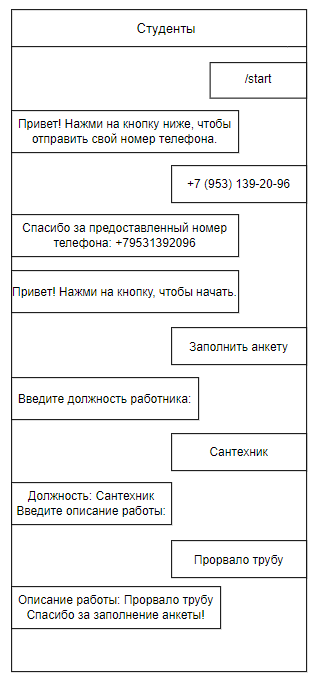


Рисунок 7 - телеграмм бот «Студенты»

# Разработка алгоритма решения задачи

Алгоритм: Создание новой заявки.

Входные данные:

* ФИО студента;
* Текст с описание поломки;
* Корпус и комната в которой проживает студент.

Выходные данные:

* Сообщение «Спасибо за создание заявки! Ваша заявка будет рассмотрена в ближайшее время.».

Шаги алгоритма:

* Получение заявки от студента;
* Вывод сообщения «Спасибо за создание заявки! Ваша заявка будет рассмотрена в ближайшее время.»;
* Занесение заявки в базу данных;
* Пересылка заявки работнику.

Алгоритм: Редактирование заявки.

Входные данные:

* Отредактированный текст заявки:

Выходные данные:

* Сообщение «Ваша заявка изменена».

Шаги алгоритма:

* Получение измененной заявки;
* Вывод сообщения «Ваша заявка изменена»;
* Обновление заявки в базе данных;
* Пересылка обновленной заявки работнику.

Алгоритм: Принятие заявки работником.

Входные данные:

* Заявка студента:

Выходные данные:

* Сообщение «Заявка принята».

Шаги алгоритма:

* Получение статуса заявки от работника;
* Вывод сообщения «Заявка принята»;
* Обновление статуса заявки и привязка ФИО работника в базе данных;

Алгоритм: Отклонение заявки работником.

Входные данные:

* Заявка студента:

Выходные данные:

* Сообщение «Заявка отклонена по причине «причина отклонения заявки»».

Шаги алгоритма:

* Получение статуса заявки от работника;
* Вывод сообщения «Заявка отклонена по причине «причина отклонения заявки»»;
* Пересылка этой заявки другому работнику;

Алгоритм: Завершение работы по заявке.

Входные данные:

* Изменение статуса заявки на «Выполнено»:

Выходные данные:

* Сообщение «Заявка выполнена».

Шаги алгоритма:

* Получение статуса заявки от работника;
* Вывод сообщения «Заявка выполнена»;
* Обновление статуса заявки в базе данных;
* Пересылка сообщения «Заявка выполнена» студенту;

Так же для понимания как реагирует система был составлен пользовательский сценарий, который указан в таблице A.

Таблица A – Пользовательские сценарий

|  |  |  |
| --- | --- | --- |
| № функции | Пользователь | Система |
| 1 | Создание заявки | 1. Предлагает ввести причину  2. После ввода причины дублирует причину и пишет «Спасибо за создание заявки! Ваша заявка будет рассмотрена в ближайшее время.»  3.Перенаправляет данную заявку сотрудникам |
| 2 | Принятие заявки | 1. Отображает сотруднику ФИО студента, который создал заявку, его корпус, комнату, и причину заявки  2. При нажатии на кнопку «Принять» пишет «Заявка принята»  3. Пишет студенту что заявка принята сотрудником |
| 3 | Отклонение заявки | 1. Отображает сотруднику ФИО студента, который создал заявку, его корпус, комнату, и причину заявки  2. При нажатии на кнопку «отклонить заявку» сотрудник указывает причину отклонения заявки  3. Пишет студенту что заявка отклонена сотрудником по причине и пишет причину отклонения |
| 4 | Завершение заявки | 1. При нажатии на кнопку «Завершить заявку» сотрудник завершает заявку, которую он принял.  2. Пишет студенту чья заявка выполнена о том что заявка была выполнена. |

# Определения языка

Для реализации проекта " Информационная система для распределения заявок по ремонту в общежитии ", будут использоваться следующие языки программирования и технологии:

* Python – мультипарадигмальный высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.
* PostgreSQL – свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ, включая различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.
* Библиотеки и фреймворки. В проекте будут использоваться специализированные библиотеки и фреймворки, которые обеспечат создание функционала системы для распределения заявок по ремонту в общажитии. В том числе: PyQt5 для реализации админ панели. Telebot для реализации телеграмм бота для сотрудников и для студентов.

# Структура программы и требования к техническим средствам

Функциональные требования:

* Авторизация и регистрация пользователей;
* Создание заявок;
* Принятие заявок;
* Отклонение заявок;
* Завершение заявок.

Нефункциональные требования:

* Надежность: система должна быть стабильной и доступной большую часть времени