Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет**

**при Правительстве Российской Федерации»**

**(Финансовый университет)**

Департамент анализа данных и машинного обучения

Пояснительная записка к курсовой работе по дисциплине

«Современные технологии программирования»

на тему:

**«Разработка экспертной системы планирования проектов»**

Выполнил:

Студент группы ПИ21-6

Есина Е. Е.

Научный руководитель:

Шананин В. А.

**Москва**

**2023**

**Оглавление**

Введение..............................................................................................................3

1. Постановка задачи.........................................................................................4
2. Описание предметной области.....................................................................5
3. Актуальность автоматизации.......................................................................7
4. Описание программы....................................................................................8
   1. Алгоритмические решения....................................................................8
      1. Сервер..............................................................................................8
      2. Клиент...........................................................................................10
   2. Интерфейс.............................................................................................10
   3. Состав приложения...............................................................................21
      1. База данных...................................................................................21
      2. Сервер............................................................................................22
      3. Клиент...........................................................................................23
5. Назначение и состав классов программы..................................................24
   1. Сервер....................................................................................................24
   2. Клиент....................................................................................................28

Заключение

Источники

Приложения

**Введение**

В современном мире построение эффективных бизнес-процессов является одним из важнейших факторов, определяющих успех проекта, а планирование — неотъемлемой частью эффективного управления бизнесом. В проектном менеджменте одним из главных вызовов является управление временем. Нередко сроки выполнения задач бывают критически сжатыми. Это может приводить к снижению качества работ, нерешенным конфликтам, повышенному стрессу для всех участников проекта. На менеджерах проектов лежат все риски, связанные с потерей контроля над выполнением проекта и его успешностью, оценка рисков и разработка стратегии действий в случае непредвиденных ситуаций.

В связи с постоянным повышением сложности проектов и увеличением количества участников в процессе их реализации, все больше компаний стремятся использовать экспертные системы для повышения эффективности своей деятельности и улучшения качества проектов. Среди таких систем экспертная система планирования проектов является одной из самых перспективных и востребованных. Она способна не только оптимизировать сроки и стоимость проектов, но и уменьшить вероятность возникновения ошибок и снизить риски.

Целью данной работы является создание экспертной системы планирования проектов на языке программирования Java. Описываемая система состоит из 3-х частей, взаимодействующих между собой:

* База данных MySQL
* Серверная часть на основе фреймворка Spring Boot
* Клиентская часть в виде веб-приложения

1. **Постановка задачи**

В соответствии с темой необходимо разработать экспертную систему планирования проектов в формате клиент-серверного приложения, состоящего из серверной и клиентской частей.

Для **серверной части** требуется реализовать связь фреймворка Spring с СУБД через объектно-реляционную модель (ORM) и архитектурную модель MVC (Model-View-Controller). Сервер должен иметь реализацию RESTful API для взаимодействия с клиентом.

Приложение должно обладать следующими функциями:

* Отображать в таблице данные предметной области
* Для информационной модели, основанной на БД, таблицы должны быть предварительно заполнены записями
* Добавление в БД нового объекта, удаление, редактирование объекта
* Фильтрация записей из БД, удовлетворяющих введенному пользователем сложному критерию
* Сортировка записей
* Обновление изменений источника данных в базе данных.
* Загрузка данных из БД

Для **клиентской части** необходимо реализовать удобный пользовательский интерфейс для взаимодействия с базой данных, обладающий следующими возможностями:

* Наличие пункта меню «Об авторе».
* Полезные пользователю функции для отображения статистических данных или данных для построения гистограммы или графика.

1. **Описание предметной области**

Предметная область автоматизации приложения для управления проектами (Project Management Application) включает в себя следующие ключевые аспекты:

1. Управление проектами, назначение ответственных, планирование и отслеживание задач, установление сроков и приоритетов, мониторинг выполнения и т. д.

2. Создание отчетов, анализ данных, мониторинг и контроль проектов.

3. Обеспечение безопасности данных, контроль доступа к информации, защита от несанкционированного доступа и взлома.

Основная цель автоматизации управления проектами - повышение эффективности и улучшение качества управления проектами, уменьшение рисков и сокращение времени на выполнение проектов.

Приложение функционирует по следующему принципу: для доступа к базе данных пользователю необходимо зарегистрироваться, после чего пользователю присваивается уровень доступа (роль). Уровней доступа существует 2 — администратор (admin) и гость (guest); по умолчанию каждому новому пользователю присваивается роль гостя. Пользователь с уровнем доступа admin может добавлять, изменять и удалять записи в базе данных; гость может только просматривать существующие записи. Для доступа к расширенному функционалу пользователю необходимо обратиться к администратору для изменения его роли в системной таблице. Программа позволяет управлять проектами (создавать, редактировать, удалять), разбивать проект на подзадачи, назначать срок выполнения и ответственного за выполнение сотрудника. Пользователь так же может сортировать записи в БД, производить поиск по ключевым словам и отслеживать статистические данные.

Приложение функционирует на основе 6 сущностей для представления данных:

* Проект (Project) — сущность представления проекта.
* Задача (Task) — сущность представления подзадач проекта. Один проект может быть разделен на несколько подзадач. Так же могут быть созданы подзадачи, не относящиеся к проекту.
* Сотрудник (Staff) — сущность представления конкретного сотрудника. Сотрудник может быть назначен на выполнение подзадачи или существовать отдельно.
* Пользователь (User) — сущность представления пользователя.
* Роль (Role) — Роль пользователя (admin или guest).
* Пользователь\_роль (User\_role) — промежуточная сущность для связи пользователя и его роли.

1. **Актуальность автоматизации**

Создание экспертной системы планирования проектов на языке   
Java – это актуальная тема среди исследователей в области управления проектами. В современных условиях, с высокой динамикой изменений бизнес-среды, проекты должны быть быстро и точно завершены. В связи с этим создание систем, которые могут помочь оптимизировать и улучшить процессы управления проектами, является приоритетной задачей.

В проектном менеджменте одним из главных вызовов является управление временем. В этом контексте создание экспертной системы планирования проектов на языке Java может быть важным инструментом в руках менеджеров проектов. Разработка такой системы позволит автоматизировать и оптимизировать процессы менеджмента проектов, упрощая планирование, управление ресурсами и контроль над сроками выполнения задач. Это может существенно повысить производительность команды и увеличить скорость достижения поставленных целей. Также приложение позволит команде лучше координировать свою работу, обмениваться информацией и следить за общими задачами. Это помогает избежать конфликтов и ошибок, сокращает время на обработку запросов и упрощает процесс принятия решений.

Все данные хранятся исключительно на локальном сервере, что значительно повышает безопасность приложения. Так же за счет реализации клиента в формате веб-приложения доступ к данным возможен практически с любого устройства с установленным браузером.

Таким образом, экспертная система планирования проектов на языке Java может стать инструментом, который поможет менеджерам проектов более эффективно и точно управлять временем, учитывать риски и увеличить шансы на успех проекта. Создание такой системы представляет собой важный вклад в область управления проектами и является актуальной задачей для исследования и разработки новых подходов в этой области.

1. **Описание программы**
   1. **Алгоритмические решения**
      1. **Сервер**

При запуске сервера происходит сканирование зависимостей, заданных в файле конфигурации Gradle, для определения библиотек и компонентов, необходимых для загрузки. Затем сервер инициализирует все внедренные компоненты.

После этого сервер начинает обрабатывать запросы, приходящие от клиента. Запросы могут быть адресованы к любому URL-адресу, определенному в приложении.

При первом обращении к серверу вызывается окно авторизации Spring. В классе SpringSecurity заранее администратором определены 2 роли: admin и guest. Метод класса SpringSecurity определяет роль авторизовавшегося пользователя и допускает только на определенные страницы.

После авторизации клиент отправляет серверу URL-запрос, далее сервер сопоставляет его с одной из функций контроллера приложения:

* 404 (NOT FOUND) — будет отправлено в случае, если нет функции, обрабатывающей такой запрос
* 400 (BAD REQUEST) — будет отправлено если запрос не удовлетворяет параметрам функции
* 200 (OK) — код ответа в случае, если запрос удовлетворяет условиям
* 500 (INTERNAL SERVICE ERROR) — будет отправлено в случае, если пользователь запрашивает данные, которых нет на сервере

При совпадении запроса и параметров функции контроллера происходит обращение к сервису, который наследует методы JPA репозитория, с последующим обращением в базу данных (рис. 1).

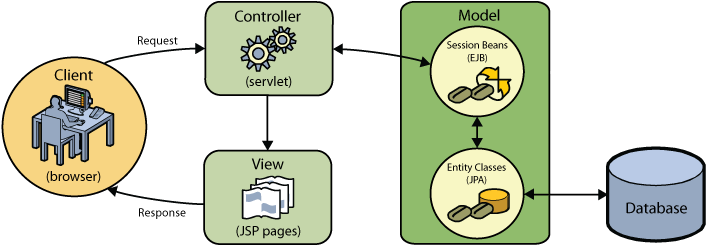
****

Рис. 1. Взаимодействие контроллеров с базой данных.

Взаимодействие базы данных, сервера и клиента происходит на основе архитектурной модели MVC и ее компонентов (модели, представления, контроллера).

1. Модель (Model) — отвечает за получение, обновление и удаление данных и обеспечивает их целостность и согласованность.

2. Представление (View) — представляет собой пользовательский интерфейс, отвечает за визуализацию данных и предоставление пользователю информации об объектах приложения.

3. Контроллер (Controller) — отвечает за обработку входящих запросов от пользователя и управление моделью и представлением. Контроллер принимает запросы от пользователя и определяет, какую модель и представление использовать для формирования ответа. Это позволяет реализовывать бизнес-логику приложения и обеспечивать взаимодействие между моделью и представлением.

* + 1. **Клиент**

Клиент взаимодействует с сервером по протоколу HTTP.   
Для формирования запросов используются стандартные методы «POST», «GET», «PUT» и т. д. В запросе содержится метод, URL-адрес страницы и параметры запроса. После отправки запроса клиент ожидает ответ от сервера.

Контроллер получает запрос и передает его на обработку соответствующему сервису, обрабатывает ответ от сервиса, формирует ответ на запрос и возвращает его клиенту. Ответ может быть в формате HTML, JSON или XML, в зависимости от типа запроса. Клиент получает ответ и обрабатывает его соответствующим образом. Например, если ответ содержит HTML, он может отображаться в браузере.

Spring предоставляет удобный способ для обработки запросов и формирования ответов на стороне сервера, что делает общение между клиентом и сервером более эффективным и удобным.

* 1. **Интерфейс**

Пользовательский интерфейс разработан в формате веб-приложения,

доступ к базе данных осуществляется через браузер. Было реализовано меню, 4 основные страницы взаимодействия, 6 страниц редактирования, 3 страницы регистрации и авторизации, и страница «Об авторе».

При создании интерфейса использовались инструменты:

* Язык разметки HTML для оформления страниц
* Шаблонизатор Thymeleaf для получения данных из модели
* JavaScript для функций внутри HTML-страниц
* Bootstrap для оформления

Всего в приложении представлено 14 окон:

* Окно авторизации
* Окно регистрации
* Системная страница с отображением всех пользователей
* Окно с отображением всех проектов
* Окно добавления нового проекта
* Окно редактирования существующего проекта
* Страница отображения всех подзадач
* Страница создания новой подзадачи
* Окно редактирования подзадачи
* Страница отображения сотрудников
* Страница добавления нового сотрудника
* Страница редактирования данных сотрудника
* Справка об авторе

Переходы между страницами реализованы с помощью кнопок на HTML-страницах. При переходе по ссылке в браузере открывается окно   
авторизации (рис. 2), в котором содержатся поля для ввода логина (email) и пароля (password), а также кнопка «войти» (Submit).

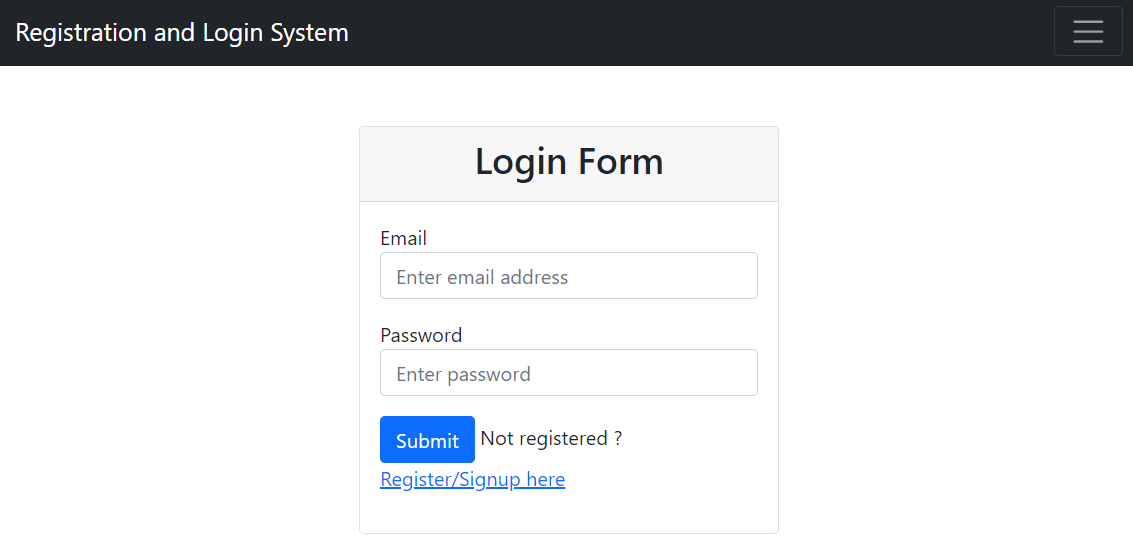


Рис. 2. Окно авторизации.

При нажатии на кнопку «войти» происходит проверка на существование пользователя с такими данными в системе. При неверно указанных логине или пароле выводится соответствующее   
сообщение (рис. 3).

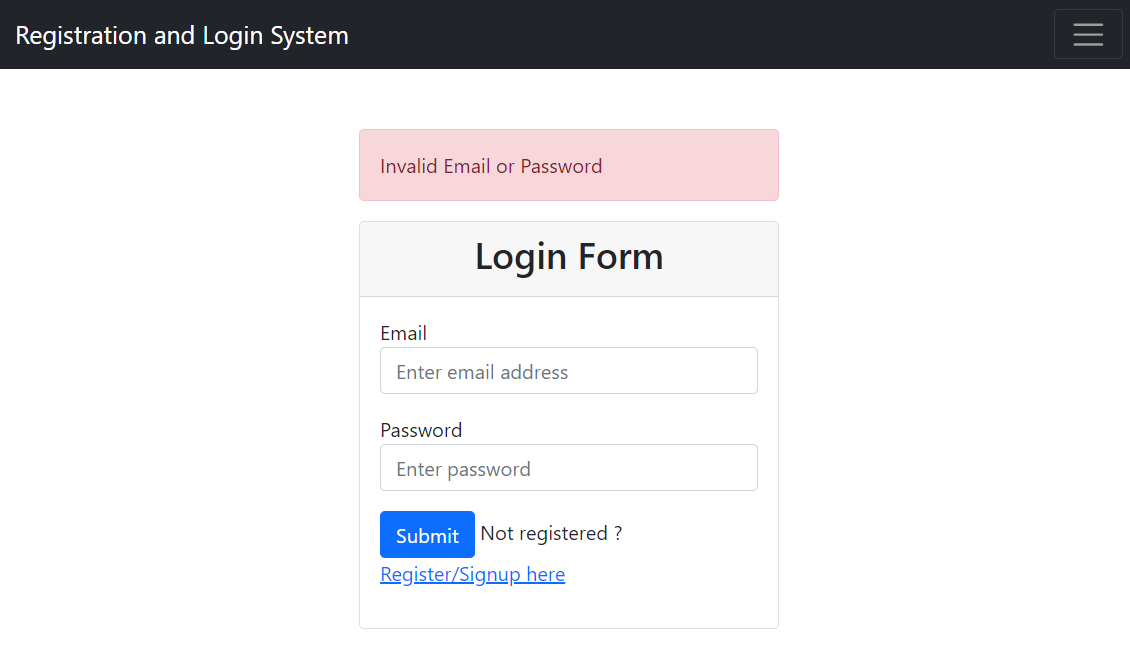


Рис. 3. Сообщение о неверных данных авторизации.

В случае если пользователь открывает приложение первые, ему необходимо перейти на страницу регистрации по ссылке «Register/Signup here», расположенной под кнопкой «Submit». При переходе по ссылке открывается страница регистрации «Register» (рис. 4). Для успешной регистрации пользователю необходимо указать:

* Имя (First name)
* Фамилию (Last name)
* Адрес электронной почты (email)
* Пароль (Password).

Адрес электронной почты должен быть уникальным для каждого пользователя. После регистрации пользователь будет перенаправлен на страницу авторизации.

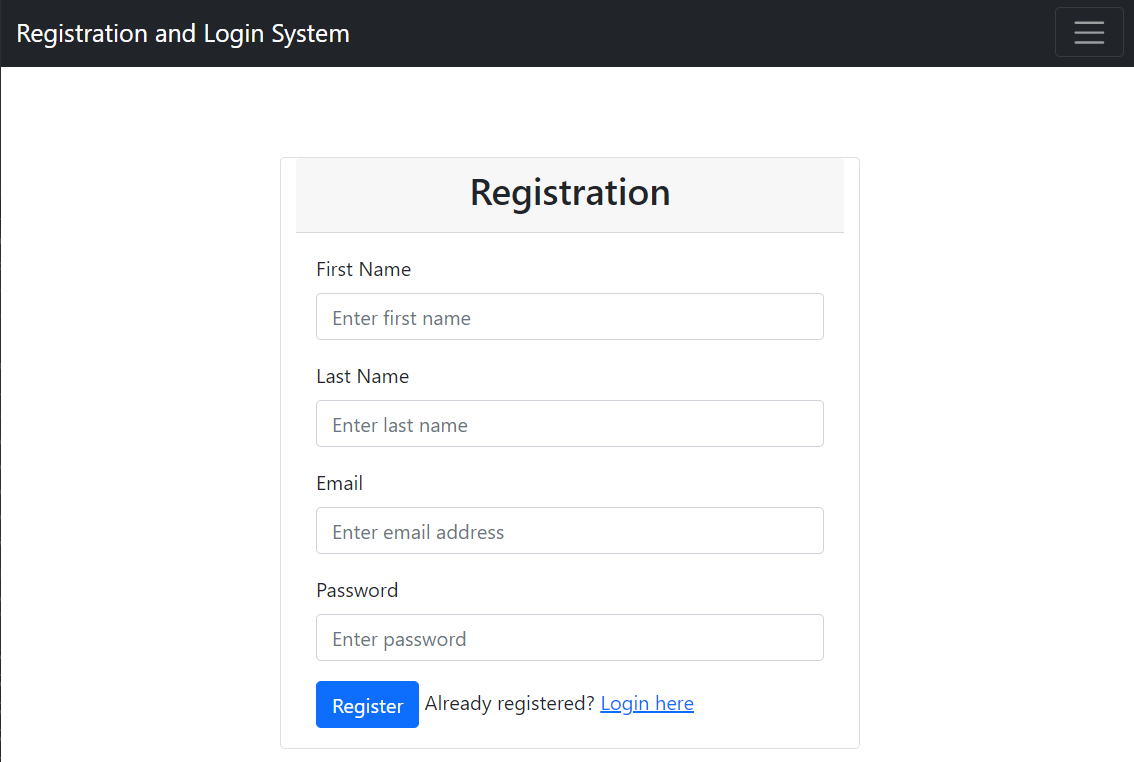


Рис. 4. Страница регистрации.

При успешной авторизации открывается главная страница системы планирования проектов (рис. 5). На странице отображается таблица базы данных проектов, а также статистическая информация в виде количества строк в таблице.

О проектах доступна следующая информация:

* Уникальный идентификатор (ID проекта)
* Название проекта
* Цель
* Сфера реализации
* Бюджет

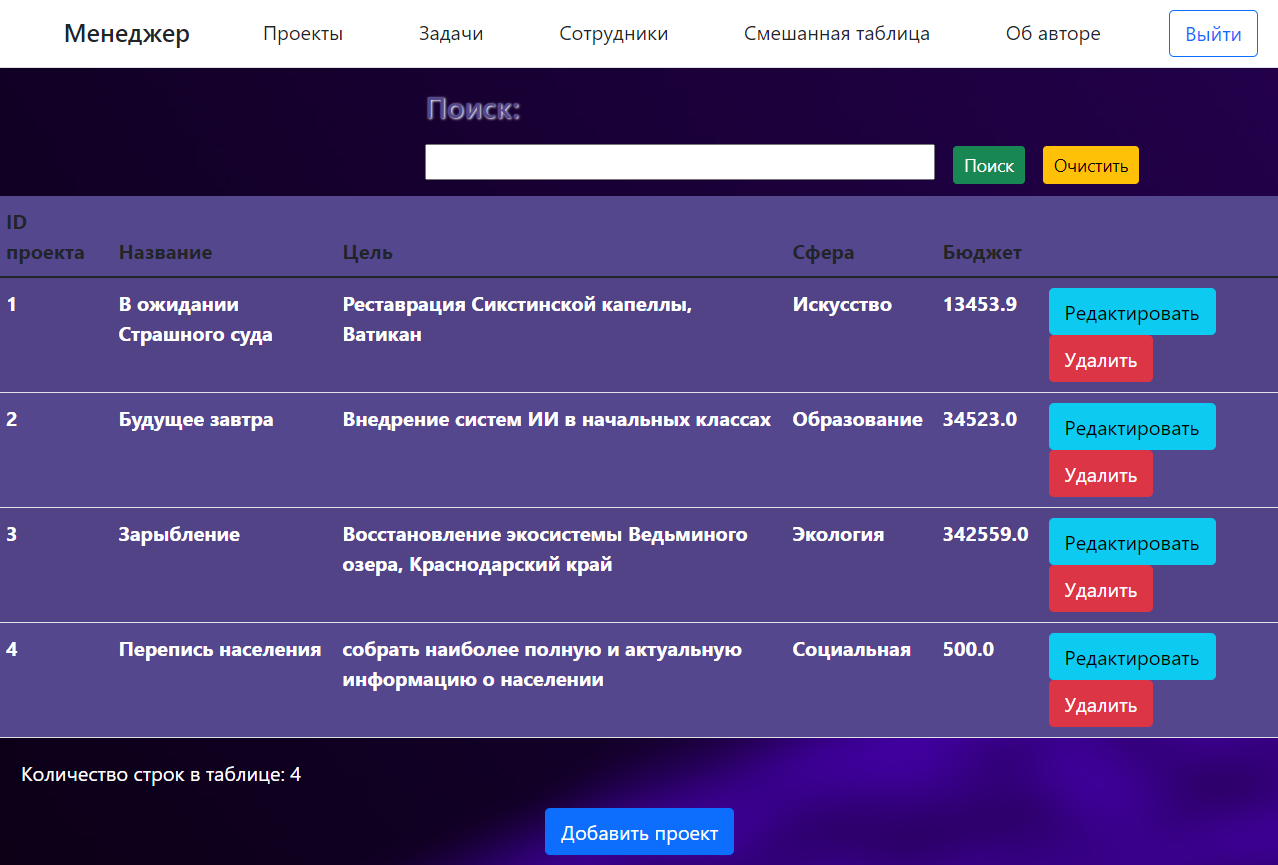


Рис. 5. Страница отображения проектов.

В верхней части каждой страницы находится навигационная панель, выполненная в виде интерактивных кнопок.

Пользователю доступны кнопки:

* «Проекты»
* «Задачи»
* «Проекты»
* «Смешанная таблица»
* «Об авторе»
* «Выход»

Таблицу можно отсортировать по столбцу, нажав на его   
название (рис. 6). Внизу таблицы находится счетчик записей.

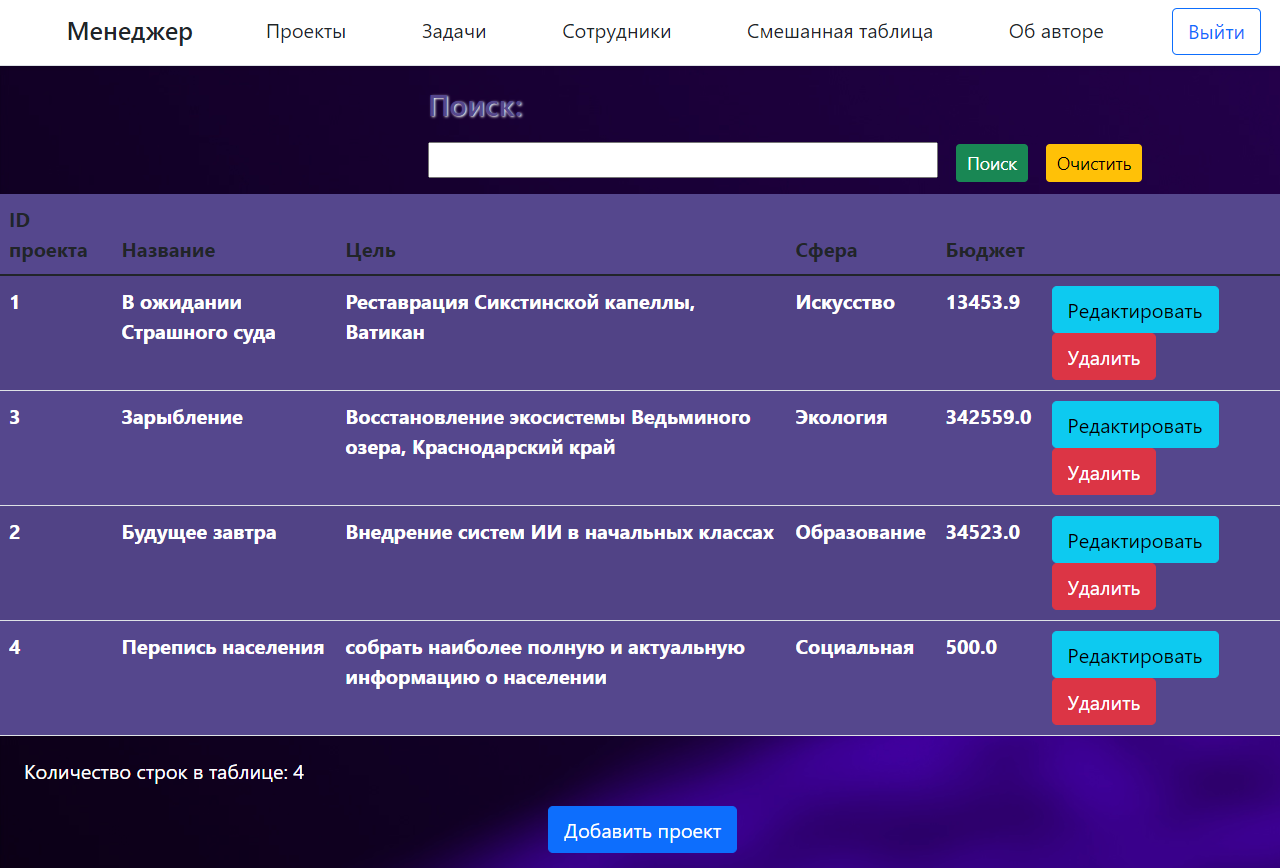


Рис. 6. Сортировка страницы по столбцу «Бюджет».

В каждой строке таблицы находятся 2 кнопки управления конкретной записью: при нажатии кнопки «Удалить» запись стирается из базы данных; при нажатии кнопки «Редактировать» вызывается страница редактирования проекта (рис. 7).

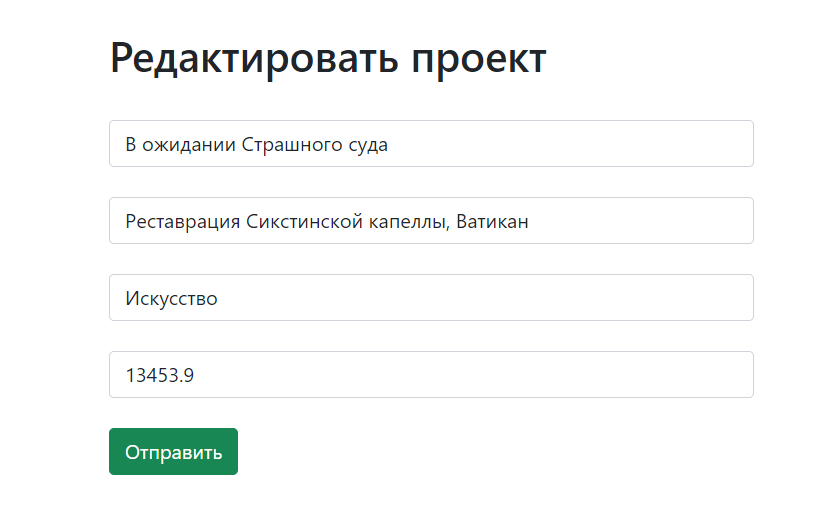


Рис. 7. Редактирование проекта.

При нажатии кнопки «Отправить» измененная запись будет сохранена в таблице, а пользователь будет перенаправлен на страницу проектов.

Кнопка «Добавить проект» на главной странице открывает схожую с редактированием форму добавления нового проекта (рис. 8). Формы добавления и редактирования являются стандартными и применены для таблиц Project, Task и Staff.

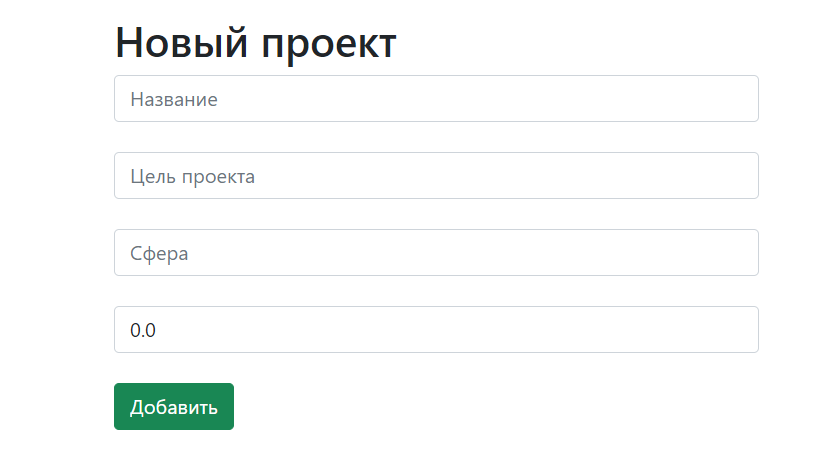


Рис. 8. Создание нового проекта.

При переходе по кнопке «Задачи» в верхнем меню пользователь будет направлен на страницу с отображением таблицы Taks (рис. 9).

В таблице «Задачи» доступны:

* Идентификатор (ID)
* Название
* Заметки
* Срок выполнения (Дедлайн)
* Идентификатор проекта
* Идентификатор ответственного
* Полное имя ответственного из таблицы Staff

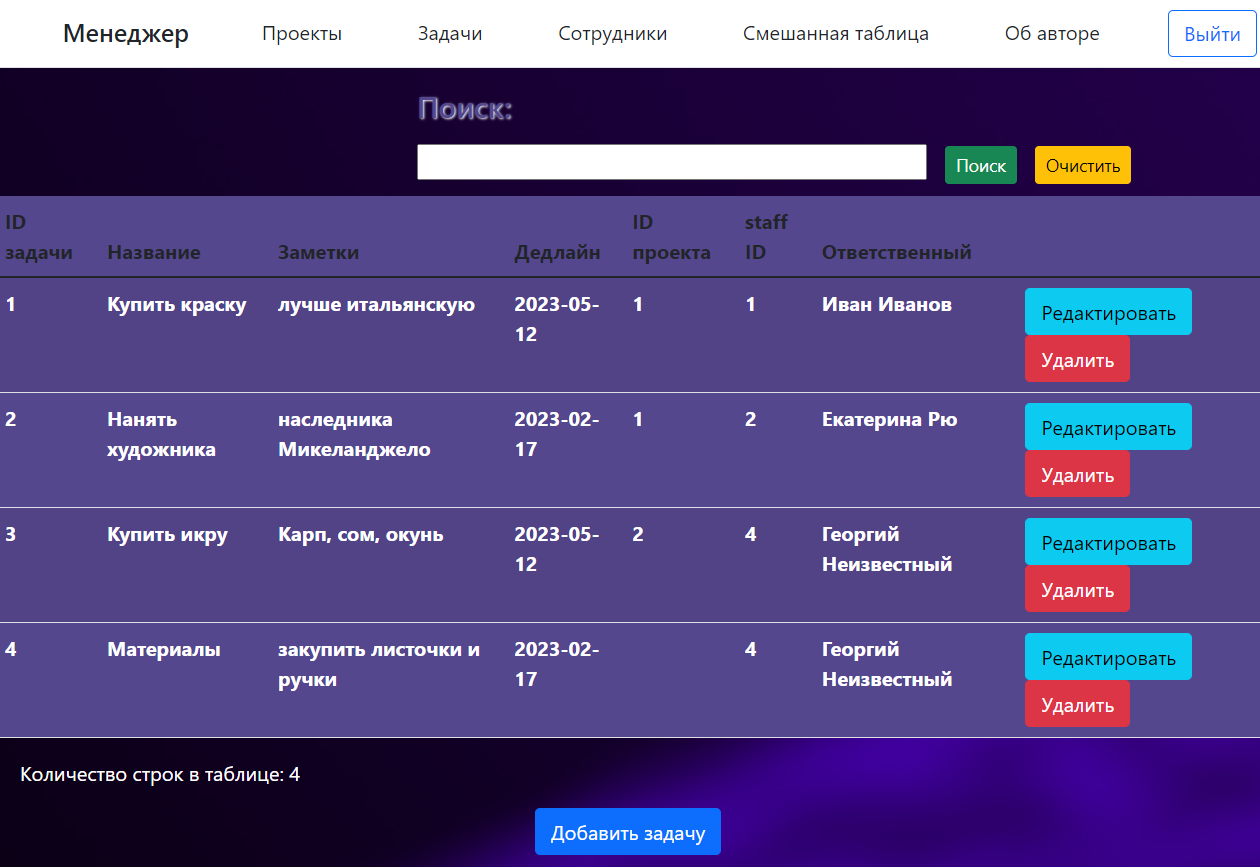


Рис. 9. Таблица Task.

Под таблицей пользователю доступна статистическая информация (рис. 10), представленная в формате счетчика количества записей в таблице и диаграммы, отображающей количество дедлайнов в день.

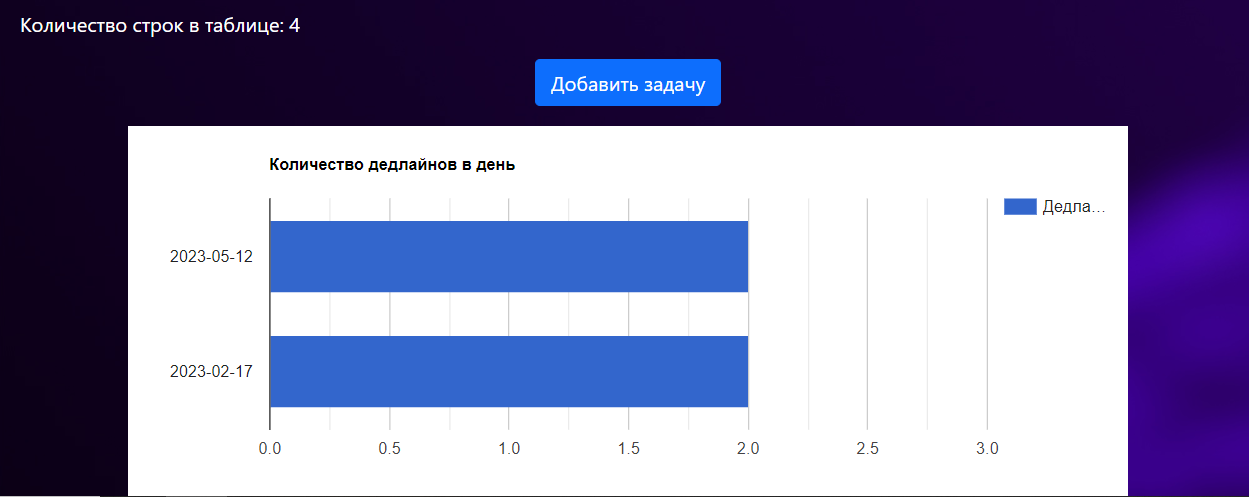


Рис. 10. Статистическая информация таблицы Task.

Кнопке «Сотрудники» в верхнем меню перенаправит пользователя на страницу с отображением таблицы Staff (рис. 11). Для таблицы так также доступны удаление, редактирование и добавление записей.

В таблице «Сотрудники» представлены следующие данные:

* Идентификатор (ID)
* Имя
* Фамилия
* Профессия
* Дополнительная информация
* URL-ссылка на картинку для отображения в профиле

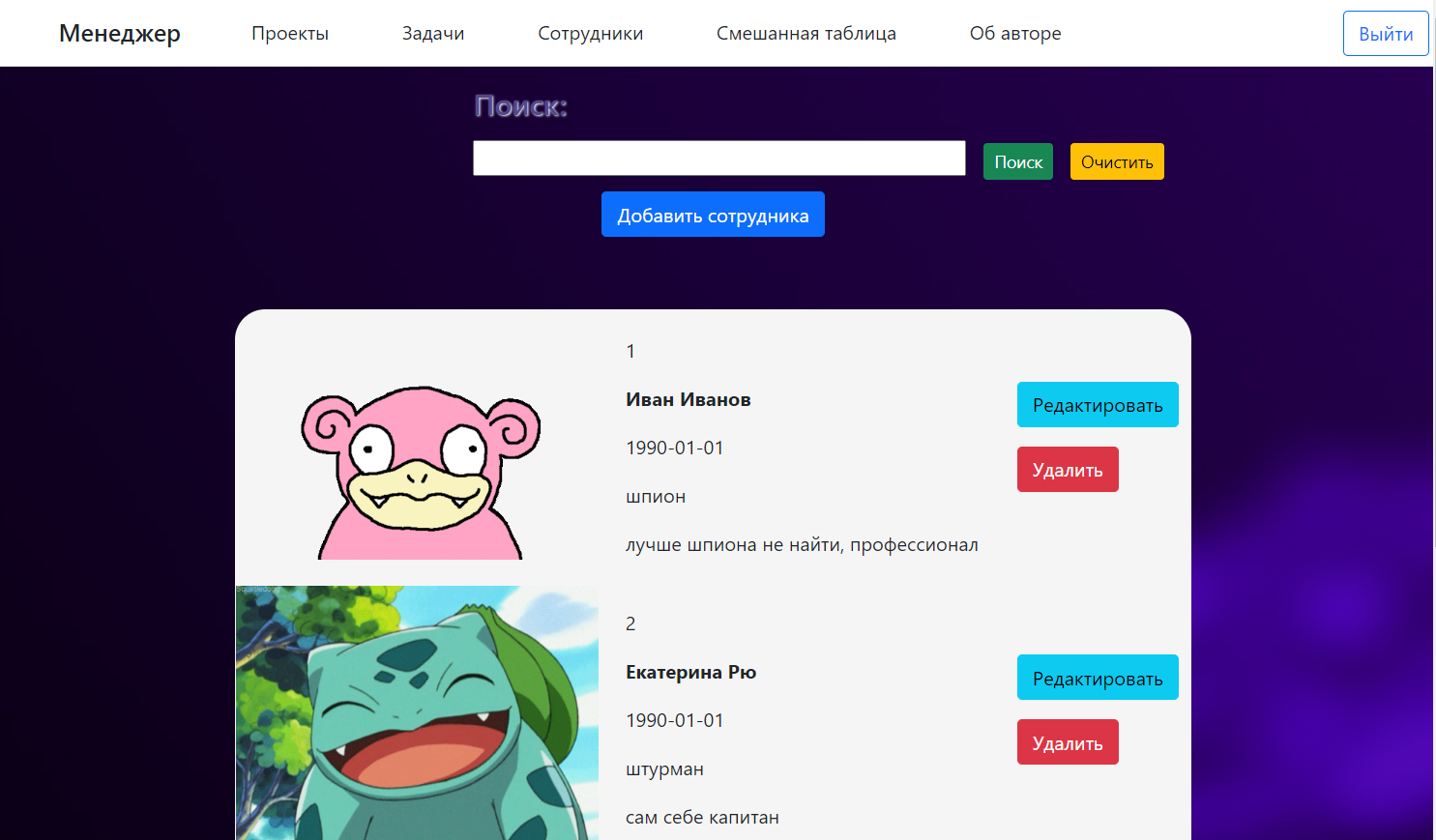


Рис. 11. Таблица Staff.

При переходе по ссылке «Смешанная таблица» в верхнем меню открывается таблица, составленная из данных таблиц Project, Task,   
Staff (рис. 12).

В таблице представлены следующие данные:

* Идентификатор проекта (таблица Project)
* Название проекта (таблица Project)
* Заголовок задачи (таблица Task)
* Комментарий к задаче (таблица Task)
* Дедлайн выполнения задачи (таблица Task)
* Полное имя ответственного (таблица Staff)

Данная страница также предоставляет пользователю статистические данные, расположенные под таблицей: количество записей в таблице и диаграмма-счетчик дедлайнов.

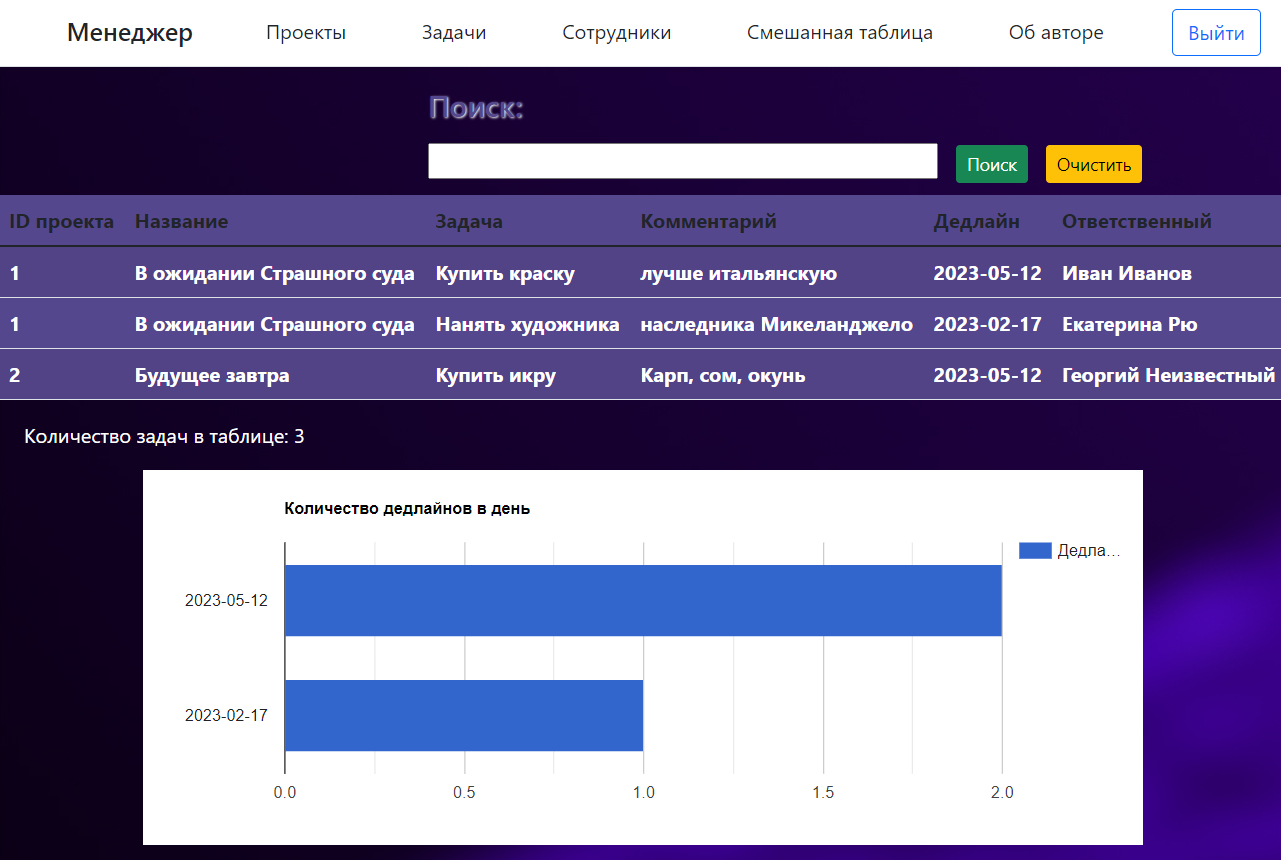


Рис. 12. Смешанная таблица.

Над таблицей находится строка поиска, с помощью нее пользователь может фильтровать записи (рис. 13). После ввода запроса и нажатии «Поиск» или клавиши Enter выводятся подходящие под запрос записи. Поиск происходит по подстроке.

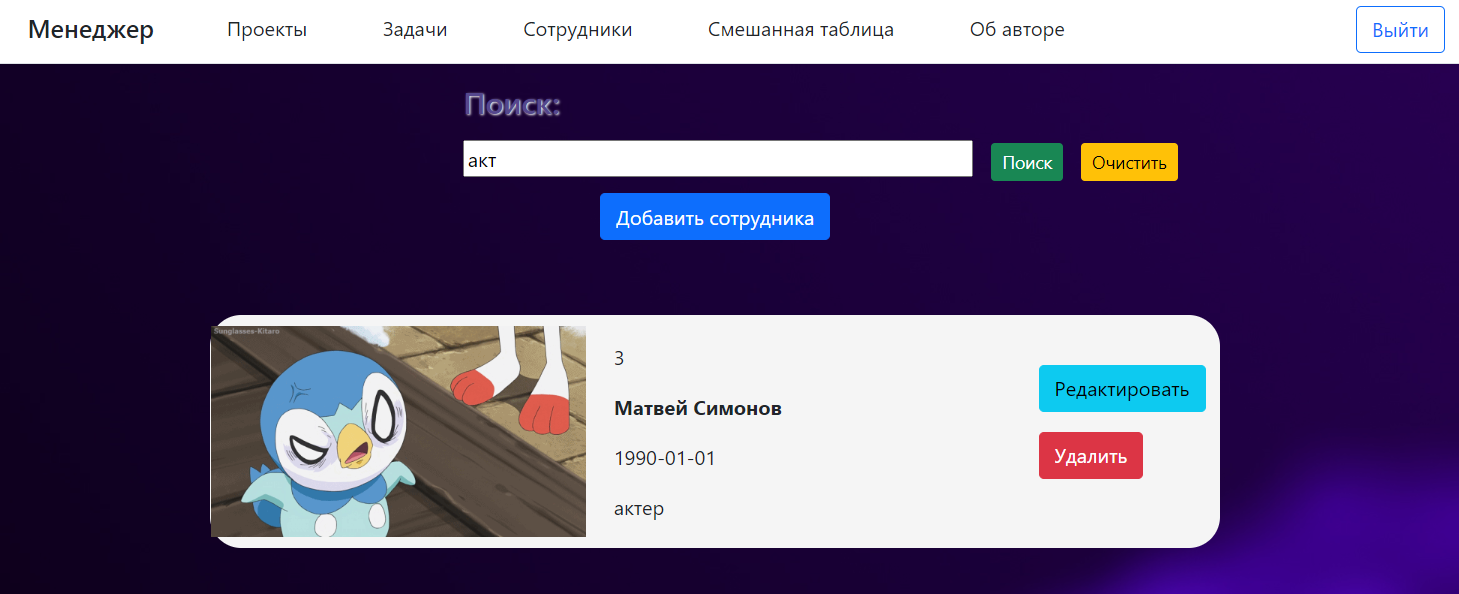


Рис. 13. Поиск по подстроке на странице «Сотрудники».

При переходе по кнопке «Об авторе» в верхнем меню открывается простая страничка с данными разработчика (рис. 15).

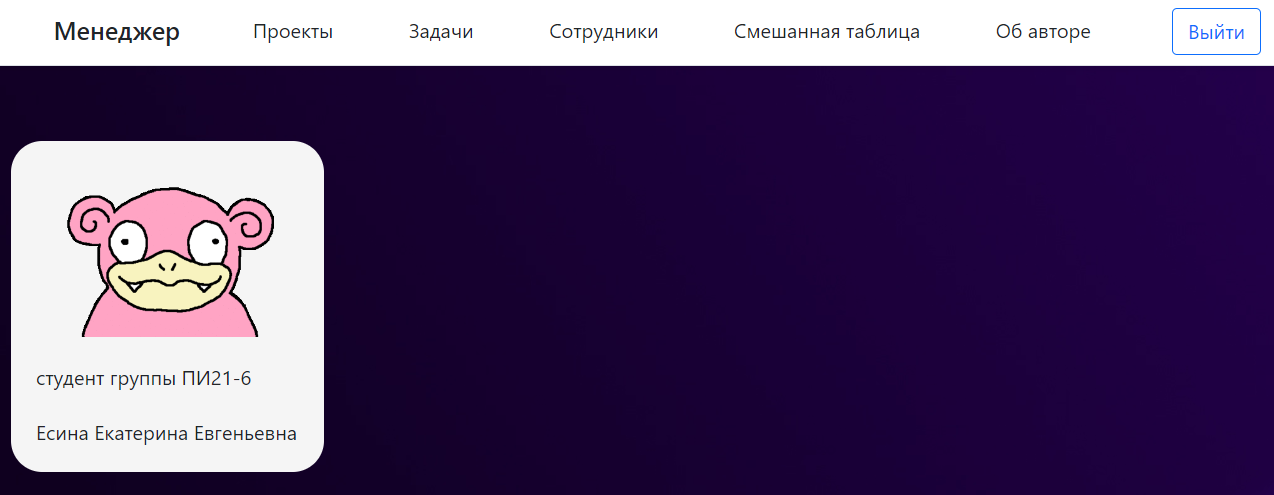


Рис. 14. Страница «Об авторе».

Выйти из приложения можно с помощью кнопки «Выход» на панели управления. При этом пользователь возвращается на страницу   
авторизации (рис. 15). Все изменения сохраняются в базе данных и будут отображены при повторном входе в приложение.

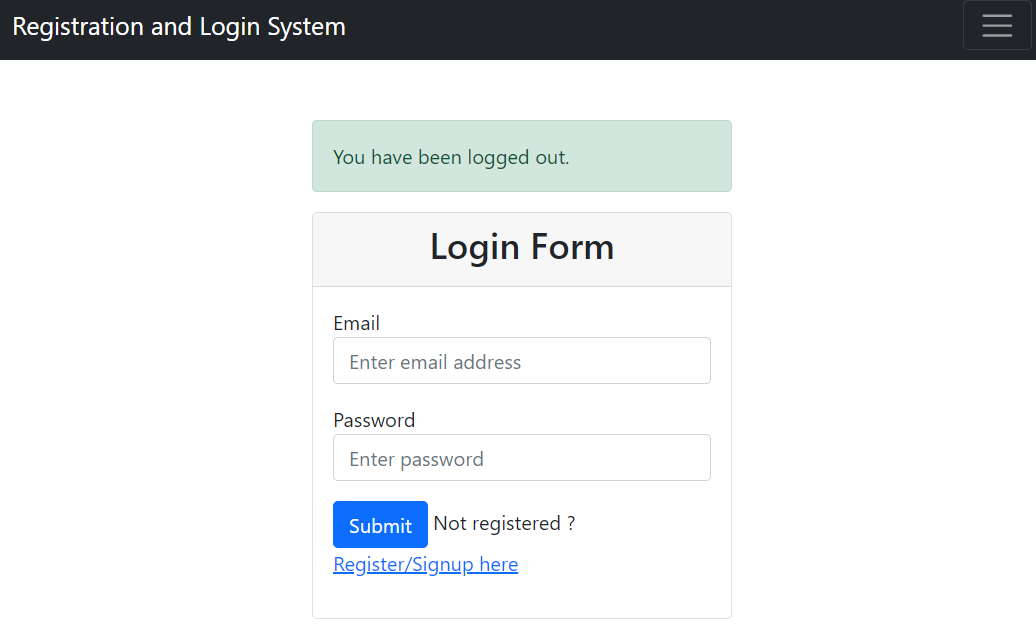


Рис. 15. Выход из приложения.

* 1. **Состав приложения**
     1. **База данных**

База данных (Рис. 16) выполнена в MySQL и состоит их 6-и таблиц, связанных уникальными идентификаторами. Пользователь может напрямую взаимодействовать с таблицами:

* Project — управление записями о проектах
* Task — управление подзадачами
* Staff — управление информацией о сотрудниках

Связь между таблицами определена так, что у одного проекта может быть одна или несколько подзадач, у одной задачи может быть один исполнитель.

Регистрация и авторизация возможна благодаря таблицам:

* Users — хранение информации о пользователе
* Roles — хранение возможных ролей пользователей (user и admin)
* Users\_roles — таблица, связывающая пользователя и его роль

У пользователя может быть только одна роль.

**A picture containing text, screenshot, diagram, number

Description automatically generated**

Рис. 16. ER-диаграмма базы данных.

* + 1. **Сервер**

При реализации сервера использованы следующие компоненты:

**Spring Web MVC** — фреймворк для создания веб-приложений, основанный на модели MVC (Model-View-Controller). Фреймворк позволяет использовать @Controller и @RestController бины для обработки http-запросов.

**Apache Tomcat** — легко настраиваемый сервер, использующийся для развертывания веб-приложений Java используется по умолчанию.

**Spring Data MySQL** предоставляет удобный доступ к объектно-реляционному отображению (ORM) и инструменты для выполнения запросов к базе данных.

**JPA Repository —** это надстройка над JPA (Java Persistence API), которая предоставляет простой и удобный интерфейс для взаимодействия с базой данных, что позволяет избежать написания сложных SQL-запросов.

Проект собран на основе **Gradle.**

Project **Lombok** — это библиотека Java, которая автоматически подключается к редактору и инструментам сборки; позволяет использовать конструкторы @AllArgsConstructor, @NoArgsConstructor, @ToString и др.

* + 1. **Клиент**

Клиент представлен в виде веб-приложения. Основные компоненты:

**HTML** — язык разметки для оформления и понятного представления данных в браузере.

**Thymeleaf** — современный серверный механизм шаблонов Java как для веб-сред, так и для автономных сред. Принимает данные из модели и подставляет в HTML-шаблон. Thymeleaf предлагает набор интеграций Spring, которые позволяют использовать его в качестве полнофункциональной замены JSP в приложениях Spring MVC.

Для функций внутри HTML-файлов используется JavaScript.

**Bootstrap** — многофункциональный интерфейсный инструментарий для простого оформления HTML-страниц.

1. **Состав классов программы и их назначение**
   1. **Сервер**

Для функционирования сервера описан 31 класс, объединенных в пакеты с целью структуризации. Проект состоит из 6-и пакетов (рис. 17).

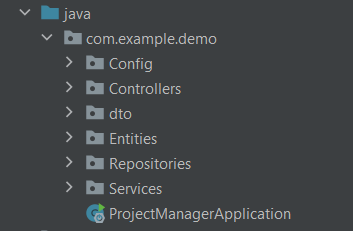


Рис. 17. Иерархия пакетов в проекте.

В пакете Config (рис. 18) хранятся конфигурационные файлы:

* MvcConfig — позволяет управлять видами (view)
* StringSecurity — управляет авторизацией, определяет доступ к URL-адресам в зависимости от роли пользователя

A screenshot of a computer

Description automatically generated with medium confidence

Рис. 18. Пакет Config.

В пакете Controllers (рис. 19) хранятся контроллеры:

* AppController — осуществляет маппинг URL-адресов
* AuthController — маппинг URL-адресов страниц регистрации и авторизации
* ComplexViewController — маппинг URL-адресов страниц отображения составной таблицы
* ProjectController — маппинг URL-адресов доступа к таблице Project
* StaffController — маппинг URL-адресов доступа к таблице Staff
* TaskController — маппинг URL-адресов доступа к таблице Task

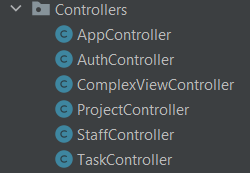


Рис. 19. Пакет Controllers.

В пакете dto (рис. 20) описаны промежуточные классы для вывода составных таблиц:

* ComplexView — класс-конструктор составной таблицы
* ComplexViewInt — интерфейс, позволяющий передать данные из конструктора в шаблон
* TaskStaff — класс-конструктор, связывающий таблицы Task и Staff
* TaskStaffInt — интерфейс, позволяющий передать данные в шаблон
* UserDto — класс-конструктор для передачи данных пользователя из шаблона в базу данных и вывода из БД в шаблон

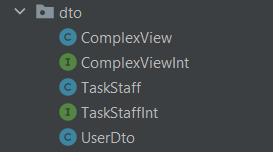


Рис. 20. Пакет dto.

В пакете Entities (рис. 21) описаны все сущности базы данных:

* Project (проект)
* Role (роль)
* Staff (сотрудник)
* Task (задача)
* User (пользователь)

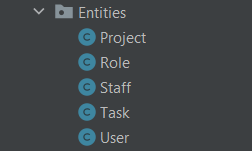


Рис. 21. Пакет entities.

В пакете Repositories (рис. 22) хранятся интерфейсы-репозитории БД:

* ProjectRepository — репозиторий сущности project
* RoleRepository — репозиторий сущности role
* StaffRepository — репозиторий сущности staff
* TaskRepository — репозиторий сущности task
* UserRepository — репозиторий сущности user

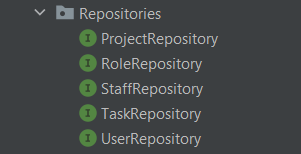


Рис. 22. Пакет repositories.

В пакете services (рис. 23) хранятся сервисы, реализующие методы репозиториев:

* ComplexViewService — сервис репозитория ComplexViewRepository
* CustomUserDetailService — сервис репозитория UserRepository
* ProjectService — сервис репозитория ProjectRepository
* StaffService — сервис репозитория StaffRepository
* TaskService — сервис репозитория TaskRepository
* UserService — промежуточный сервис-интерфейс для управления данными пользователя
* UserServiceImpl — совместный сервис репозиториев UserRepository, RoleRepository, использует UserService

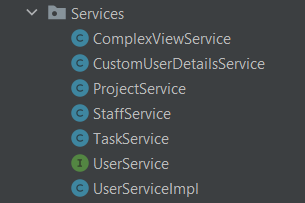


Рис. 23. Пакет repositories.

Классы вне пакетов:

* ProjectManagetApplocation — объединяет классы в единое приложение, точка входа в приложение
  1. **Клиент**

Приложение клиента включает в себя 15 HTML-страниц, разделенных на пакеты (рис. 24):

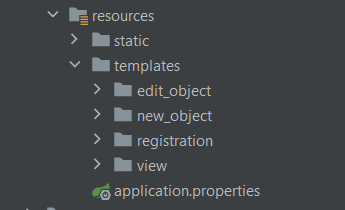


Рис. 24. Иерархия пакетов HTML-страниц.

Пакет edit\_object (рис. 25) содержит в себе все страницы редактирования существующих объектов: проектов, задач, профилей сотрудников.

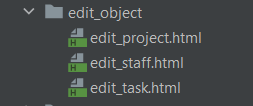


Рис. 25. Пакет edit\_object.

В пакете new\_object (рис. 26) хранятся HTML-страницы с формами создания новых объектов: проектов, задач, профилей сотрудников.

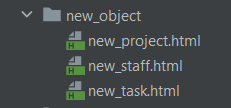


Рис. 26. Пакет new\_object.

В пакете registration (рис. 27) находятся HTML-страницы, связанные с регистрацией и авторизацией:

* Login — страница авторизации
* Register — страница регистрации нового пользователя
* Users — системная страница для отображения таблицы пользователей

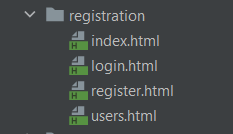


Рис. 27. Пакет registration.

Пакет view (рис. 28) содержит в себе шаблоны (виды), доступные пользователю:

* Author — страница «Об авторе»
* Complex\_view — страница отображения составной таблицы
* Project\_table — страница отображения таблицы Project
* Staff\_table — страница отображения таблицы Staff
* Task\_table — страница отображения таблицы Task

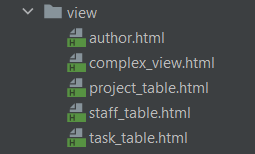


Рис. 28. Пакет view.

Данные из модели подставляются в таблицу с помощью шаблонизатора Thymeleaf; оформление таблиц сделано средствами Bootstrap.

**Заключение**

В ходе работы была разработана экспертная система планирования проектов на языке программирования Java с использованием фреймворка Spring. На основании требований были созданы серверная и клиентская часть приложения с использованием модели MVC для взаимодействия с базой данных. Созданное приложение было успешно запущено и протестировано.

Простой и понятный интерфейс не требует дополнительных умений для успешного применения. Приложение может использоваться для эффективной организации и управления проектами, отслеживания прогресса работ и координации всех задач и участников проекта.

В процессе выполнения работы были выполнены поставленные задачи и приобретены ценные навыки проектирования и разработки серверных и веб-приложений.

**Источники**

1. Козмина Ю., Харроп Р. Spring 5 для профессионалов. - Киев: Диалектика-Вильямс, 2019. - 1120 с.
2. Связанные сущности в Hibernate [Электронный ресурс] // java-online. URL: <http://java-online.ru/hibernate-entities.xhtml> (дата обращения 10.05.2023)
3. Bootstrap Getting started [Электронный ресурс] URL: <https://bootstrap-4.ru/docs/5.3/getting-started/introduction/> (дата обращения 10.05.2023)
4. Integrating Thymeleaf with Spring [Электронный ресурс] URL: <https://www.thymeleaf.org/doc/tutorials/3.1/thymeleafspring.html> (дата обращения 10.05.2023)
5. Spring Web [Электронный ресурс] URL: <https://docs.spring.io/spring-boot/docs/current/reference/html/web.html> (дата обращения 10.05.2023)

**Приложения**

