|  |
| --- |
| **Министерство образования Республики Беларусь**  **Учреждение образования**  **«Брестский государственный технический университет»**  **Кафедра ИИТ**  **ПОЯСНИТЕЛЬНАЯ ЗАПИСКА** К КУРСОВОМУ ПРОЕКТУ ПО ДИСЦИПЛИНЕ«Конструирование программного обеспечения» **Тема: «Разработка программы расчета стипендии»**  **КП.ПО-9.1-40 01 01**  **Листов: 68**  **Выполнил:**  Студент 2-го курса,  ФЭИС,  Группы ПО-9  Харитонович З. С.  **Нормоконтроль:**  Сааков А.В.  **Проверил:**  Сааков А.В.  **Брест 2023** |

# ВВЕДЕНИЕ

В наше время информационные технологии внедряются во все сферы жизни. Повседневные задачи, которые раньше выполняли люди, автоматизируются с помощью программных средств.

Целью данного курсового проекта является создание автоматизированной системы, которая упрощает управление и работу с данными для конечного пользователя, с использованием современных технологий разработки: языка программирования высокого уровня, графического интерфейса, базы данных.

1. **ПОСТАНОВКА ЗАДАЧИ**

Разработка программы расчета стипендии.

Для расчета стипендии необходимо обработать информацию следующего вида: номер группы, Ф.И.О. студента, форма обучения (платная/бюджетная), зачеты по пяти предметам (зачет/незачет), оценки по 4 предметам, признак участия в общественной работе: 1 – активное участие, 0 – неучастие.

Индивидуальное задание: рассчитать стипендию для студентов-бюджетников. При этом: отличники (отметки 9, 10) и общественники получают 50% надбавку, а просто отличники – 25%. Студенты со средним баллом, равным или ниже 5, стипендию не получают. Базовый размер стипендии вводится с клавиатуры.

Для разработки программы будут использоваться следующая технология: объектно-ориентированное программирование (ООП).

Средствами разработки программы будут: язык программирования Java, FXML – язык разметки интерфейса.

Среда разработки – JetBrains IntelliJ IDEA.

Тип используемой СУБД – MySQL.

1. **АНАЛИЗ ЗАДАЧИ**

## 2.1 РАЗРАБОТКА МОДУЛЬНОЙ СТРУКТУРЫ ПРОГРАММЫ

Первым этапом работы программы является авторизация. Для этого необходимо создать таблицу, содержащую сведения об учётных записях пользователей следующего вида: login, password, role. В качестве password хранится хеш-значение, а не исходное. Role равняется 1 для администратора и 0 для обычного пользователя. Изначально необходимо вручную записать в таблицу учётную запись администратора.

Вторым этапом работы программы является работа с таблицей данных, которая становится возможной после прохождения авторизации.

Модуль администратора включает следующие подмодули со следующими функциональными возможностями:

1. Управление учётными записями пользователей: просмотреть все учётные записи; добавить учётную запись; отредактировать учётную запись; удалить учётную запись.
2. Работа с данными: просмотреть все данные; добавить запись; удалить запись; редактировать запись; выполнить поиск данных; выполнить сортировку по различным полям; произвести расчет стипендии для студентов-бюджетников.

Модуль пользователя включает подмодуль работы с данными со следующими функциональными возможностями: просмотреть все данные; выполнить поиск данных; выполнить сортировку по различным полям; произвести расчет стипендии для студентов-бюджетников.

## ВЫБОР СПОСОБОВ ОПИСАНИЯ И ОБЪЕДИНЕНИЯ ВХОДНЫХ ДАННЫХ

Класс **User** – класс представления пользователя –содержит в себе поля:

* String **login –** логин пользователя;
* String **password** – пароль пользователя;
* boolean **role –** роль пользователя (true – администратор, false – обычный пользователь).

Класс **Student** – класс представления студента – содержит поля:

* int **id** – индентификатор;
* String **NSP** – фамилия, имя, отчество студента;
* String **group** – название группы студента;
* boolean **eduForm** – форма обучения студента (true – бюджетная, false – платная);
* boolean[] **credits** – зачеты студента (true – зачтено, false – не зачтено);
* int[] **exams** – оценки студента по экзаменам;
* boolean **socWork** – признак участия студента в общественной работе;
* double **scholarship** – размер стипендии студента.

Класс **Database** – класс, содержащий методы для работы с базой данных.

Класс **MainApplication** – класс приложения**.**

Класс **AuthenticationController** – модуль авторизации.

Класс **MenuController** – меню выбора подмодуля.

Класс **AccountManagerController** – подмодуль управления учётными записями модуля администратора.

Класс **ScholarshipCalculationController** – подмодуль работы с данными модуля пользователя.

Класс **ScholarshipAdminController** – подмодуль работы с данными модуля администратора (наследует **ScholarshipCalculationController).**

## РАЗРАБОТКА ПЕРЕЧНЯ ПОЛЬЗОВАТЕЛЬСКИХ ФУНКЦИЙ ПРОГРАММЫ

Класс **Database** содержит методы:

* ResultSet **sqlRequest**(String query) – запрос в базу данных (рис. 1);
* void **sqlUpdate**(String query) – внесение изменений в базу данных (рис. 2).

Класс **MainApplication** содержит методы:

* void **changeScene**(String file, String title, int v, int v1) – смена сцены (рис. 3);
* void **start**(Stage stage) – инициализация окна приложения (рис. 4).

Класс **AuthenticationController** содержит методы:

* void **onLoginButtonClick**() – авторизация (рис. 5);
* User **auth**(String login, String password) – проверка авторизации (рис. 6);
* String **stringToHash**(String str) – перевод строки в хеш (рис. 7).

Класс **MenuController** содержит методы:

* void **onAccountManagerButtonClick**() – переход к подмодулю управления учётными записями (рис. 8);
* void **onScholarshipCalculationButtonClick**() – переход к подмодулю работы с данными модуля пользователя (рис. 9);
* void **onExitButtonClick**() – выход из учётной записи, возврат к модулю авторизации (рис. 10).

Класс **AccountManagerController** содержит методы:

* void **onBackButtonClick**() – возврат к меню (рис. 11);
* void **updateTable**() – обновление содержимого таблицы пользователей (рис. 12);
* void **onAddButtonClick**() – добавление записи пользователя (рис. 13);
* void **onDeleteButtonClicked**() – удаление записи пользователя (рис. 14);
* void **onEditButtonClicked**() – редактирование записи пользователя (рис. 15).

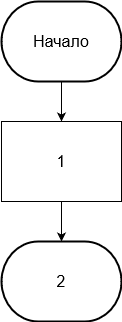
Класс **ScholarshipCalculationController** содержит методы:

* void **onBackButtonClick**() – возврат к меню (рис. 16);
* void **updateTable**() – обновление содержимого таблицы студентов (рис. 17);
* void **onSearchButtonClicked**() – поиск данных (рис. 18);
* void **onCalculateButtonClick**() – расчёт стипендии (рис. 19);
* void **onAdminButtonClicked**() – переход к подмодулю работы с данными модуля администратора (рис. 20).

Класс **ScholarshipAdminController** (наследует класс **ScholarshipCalculationController)** содержит методы:

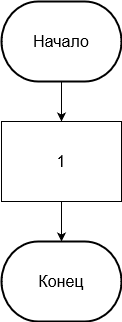
* void **onBackButtonClicked**() – возврат к подмодулю работы с данными модуля пользователя (рис. 21);
* void **onAddButtonClicked**() – добавление записи студента (рис. 22);
* void **onDeleteButtonClicked**() – удаление записи студента (рис. 23);
* void **onEditButtonClicked**() – редактирование записи студента (рис. 24).

# РАЗРАБОТКА АЛГОРИТМОВ РАБОТЫ ПОЛЬЗОВАТЕЛЬСКИХ ФУНКЦИЙ ПРОГРАММЫ



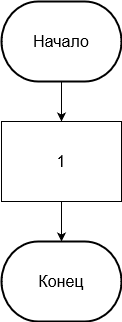
*Рисунок 1 – функция запроса в базу данных. (ResultSet sqlRequest(String query)*)

1. Подключение драйвера базы данных, установление соединения с базой данных, отправка SQL запроса, получение ответа от БД (набор данных).
2. Возврат набора данных.



*Рисунок 2 – функция внесения изменений в базу данных. (void sqlUpdate(String query))*

1. Подключение драйвера базы данных, установление соединения с базой данных, отправка SQL запроса.



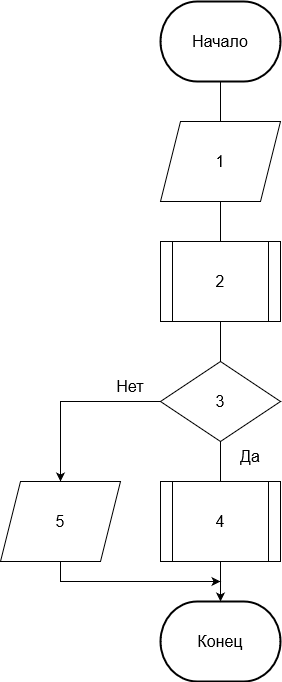
*Рисунок 3 – функция смены сцены. (void changeScene(String file, String title,*

*int v, int v1))*

1. Изменение текущей сцены на **file**, изменение имени окна на **title**, установка размеров окна **v** \* **v1**.

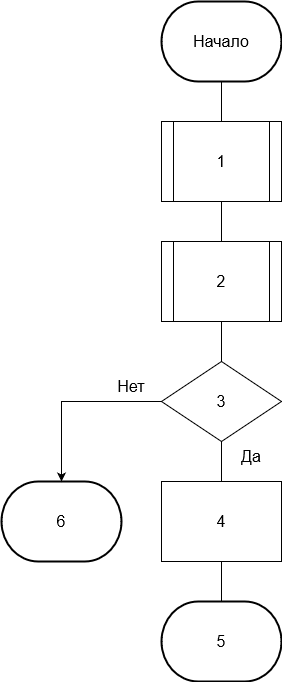


*Рисунок 4 – функция инициализации окна приложения. (void start(Stage stage))*



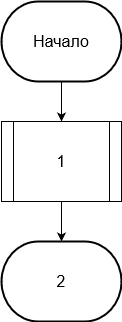
*Рисунок 5 – функция авторизации. (void onLoginButtonClick())*

1. Извлечение логина и пароля из формы.
2. Проверка авторизации (auth(login, password)).
3. Авторизация удалась (user != null)?
4. changeScene("menu-view.fxml", "Меню", 320, 240).
5. Вывод ошибки.



*Рисунок 6 – функция проверки авторизации. (User auth(String login, String password))*

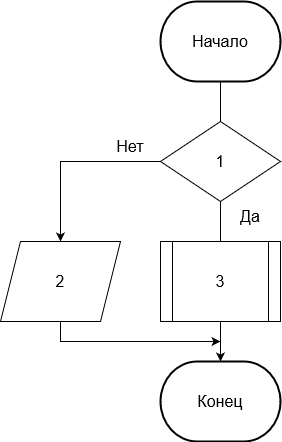
1. Хеширование пароля (stringToHash(password)).
2. Запрос записи пользователя login из базы данных (sqlRequest(query)).
3. Пароли совпадают?
4. Создание объекта user класса User на основе значений, полученных из БД.
5. Возврат объекта user.
6. Возврат null.



*Рисунок 7 – функция перевода строки в хеш.*

*(String stringToHash(String str))*

1. Хеширование строки по стандарту SHA-256, форматирование хеша в строку.
2. Возврат хеша.



*Рисунок 8 – функция перехода к подмодулю управления учётными записями. (void onAccountManagerButtonClick())*

1. Текущий пользователь является администратором?
2. Вывод ошибки.
3. changeScene("account-manager-view.fxml", "Управление учётными записями", 1280, 720).



*Рисунок 9 – функция переход к подмодулю работы с данными модуля пользователя. (void onScholarshipCalculationButtonClick())*

1. changeScene("scholarship-calculation-view.fxml", "Расчёт стипендии", 1280, 720).



*Рисунок 10 – функция выхода из учётной записи. (void onExitButtonClick())*

1. changeScene("authentication-view.fxml", "Авторизация", 320, 240).



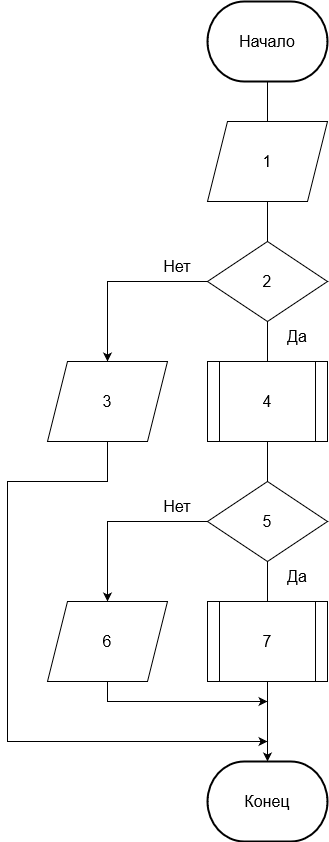
*Рисунок 11 – функция возврата к меню. (void onBackButtonClick())*

1. changeScene("menu-view.fxml", "Меню", 320, 240).



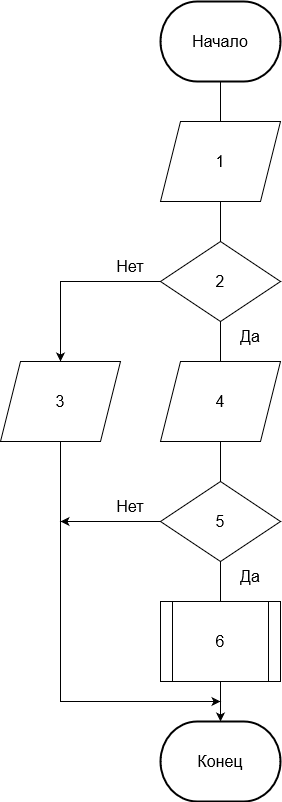
*Рисунок 12 – функция обновления содержимого таблицы пользователей. (void updateTable())*

1. Запрос записей всех пользователей из базы данных (sqlRequest(query)).
2. Перебор записей пользователей.
3. Создание объекта user класса User на основе значений, полученных из БД; добавление объекта user в массив данных data.
4. Вывод содержимого data на экран.



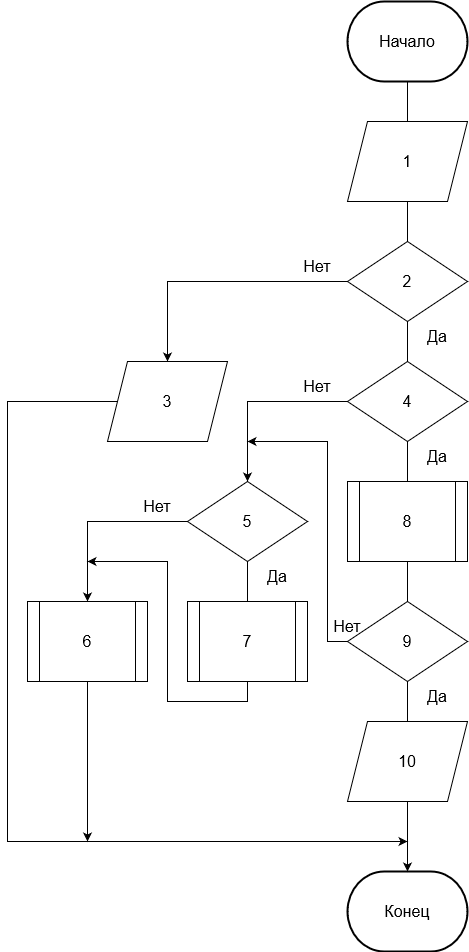
*Рисунок 13 – функция добавления записи пользователя. (void onAddButtonClick())*

1. Извлечение значений из полей формы.
2. Все поля заполнены?
3. Вывод ошибки (заполните все поля).
4. Запрос записи пользователя login из базы данных (sqlRequest(query)).
5. Запись не существует?
6. Вывод ошибки (такой пользователь уже существует).
7. Добавление записи в базу данных (sqlUpdate(query)); updateTable().



*Рисунок 14 – функция удаления записи пользователя. (void onDeleteButtonClicked())*

1. Извлечение выделенной в таблице записи.
2. Запись не пуста?
3. Вывод ошибки (выберите запись).
4. Запрос подтверждения операции.
5. Операция подтверждена?
6. Удаление записи из базы данных (sqlUpdate()); updateTable().



*Рисунок 15 – функция редактирования записи пользователя. (void onEditButtonClicked())*

1. Извлечений значений из полей формы.
2. Поля не пусты и запись в таблице выделена?
3. Вывод ошибки.
4. Логин был изменён?
5. Пароль был изменён?
6. Изменение записи в базе данных (sqlUpdate()).
7. Хеширование нового пароля.
8. Запрос записи пользователя newLogin из базы данных (sqlRequest(query)).
9. Запись не существует?
10. Вывод ошибки.



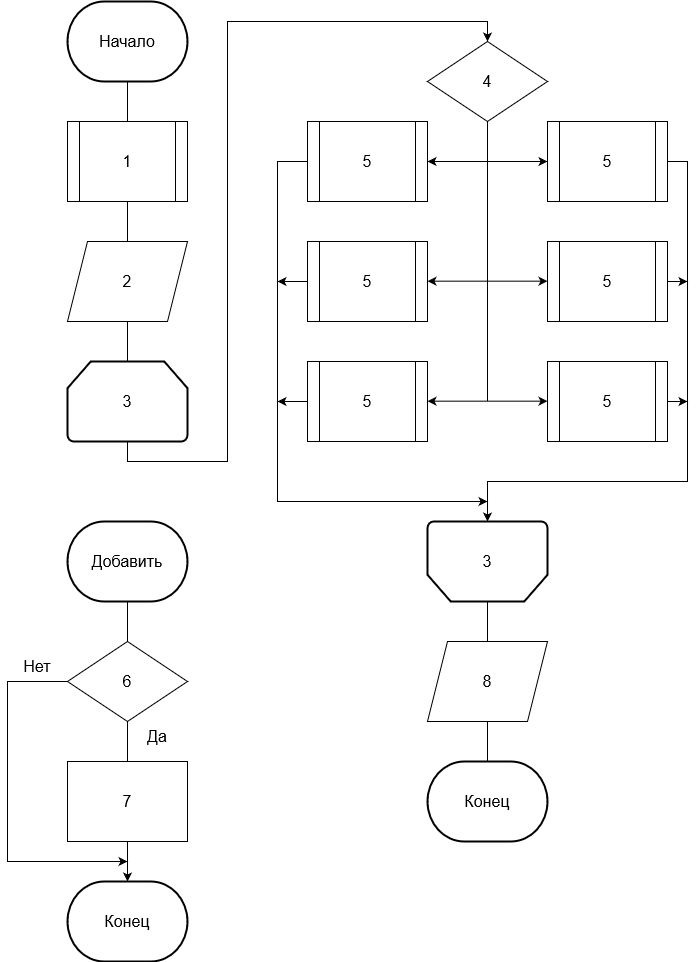
*Рисунок 16 – функция возврата к меню. (void onBackButtonClick())*

1. changeScene("menu-view.fxml", "Меню", 320, 240).



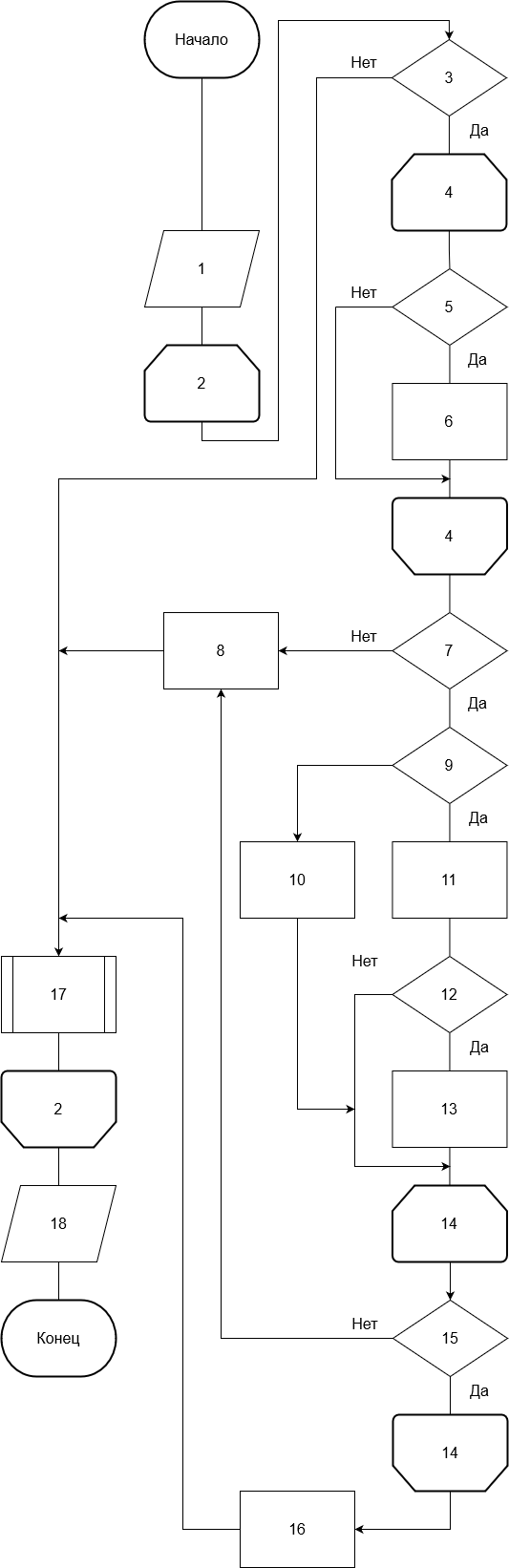
*Рисунок 17 – функция обновления содержимого таблицы студентов. (void updateTable())*

1. Запрос записей всех студентов из базы данных (sqlRequest(query)).
2. Перебор записей студентов.
3. Создание объекта student класса Student на основе значений, полученных из БД; добавление объекта student в массив данных data.
4. Вывод содержимого data на экран.



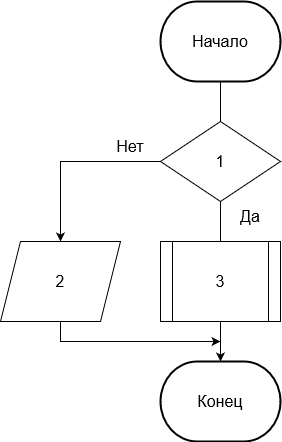
*Рисунок 18 – функция поиска данных. (void onSearchButtonClicked())*

1. updateTable().
2. Извлечение поискового запроса из формы.
3. Перебор всех студентов в таблице.
4. Поле, по которому осуществляется поиск.
5. Вызов «Добавить» для соответствующего поля.
6. Содержится ли поисковой запрос в значении соответствующего поля.
7. Добавить текущего студента в массив данных newData.
8. Вывод содержимого newData на экран.



*Рисунок 19 – функция расчёта стипендии. (void onCalculateButtonClick())*

1. Извлечение значения базового размера стипендии из формы.
2. Перебор всех студентов в таблице (data).
3. Текущий студент является бюджетником?
4. Перебор оценок по экзаменам.
5. Оценка ниже 9?
6. Студент не является отличником (isExc = false).
7. Средний балл не ниже 5?
8. Стипендия не выплачивается (scholarship = 0);
9. Студент является отличником?
10. Коэффициент k = 1.0
11. Коэффициент k = 1.25
12. Студент является общественником?
13. Коэффициент k = 1.5
14. Перебор зачётов.
15. Зачтено?
16. Установить стипендию в размере базовая \* k.
17. Изменение записи студента в базе данных.
18. Вывод содержимого обновлённого data на экран.



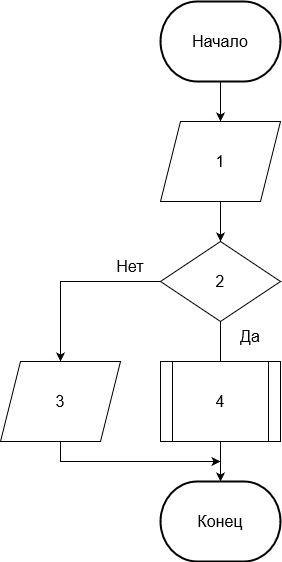
*Рисунок 20 – функция перехода к подмодулю работы с данными модуля администратора. (void onAdminButtonClicked())*

1. Текущий пользователь является администратором?
2. Вывод ошибки.
3. changeScene("scholarship-admin-view.fxml", "Расчёт стипендии", 1280, 720).



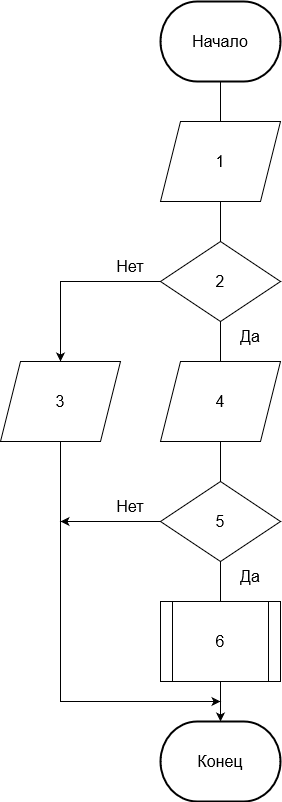
*Рисунок 21 – функция возврата к подмодулю работы с данными модуля пользователя. (void onBackButtonClicked())*

1. changeScene("scholarship-calculation-view.fxml", "Расчёт стипендии", 1280, 720).



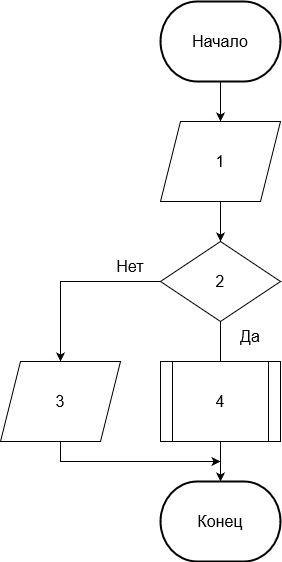
*Рисунок 22 – функция добавление записи студента. (void onAddButtonClicked())*

1. Извлечение значений из полей формы.
2. Поля не пусты?
3. Вывод ошибки.
4. Добавление записи студента в базу данных (sqlUpdate(query)).



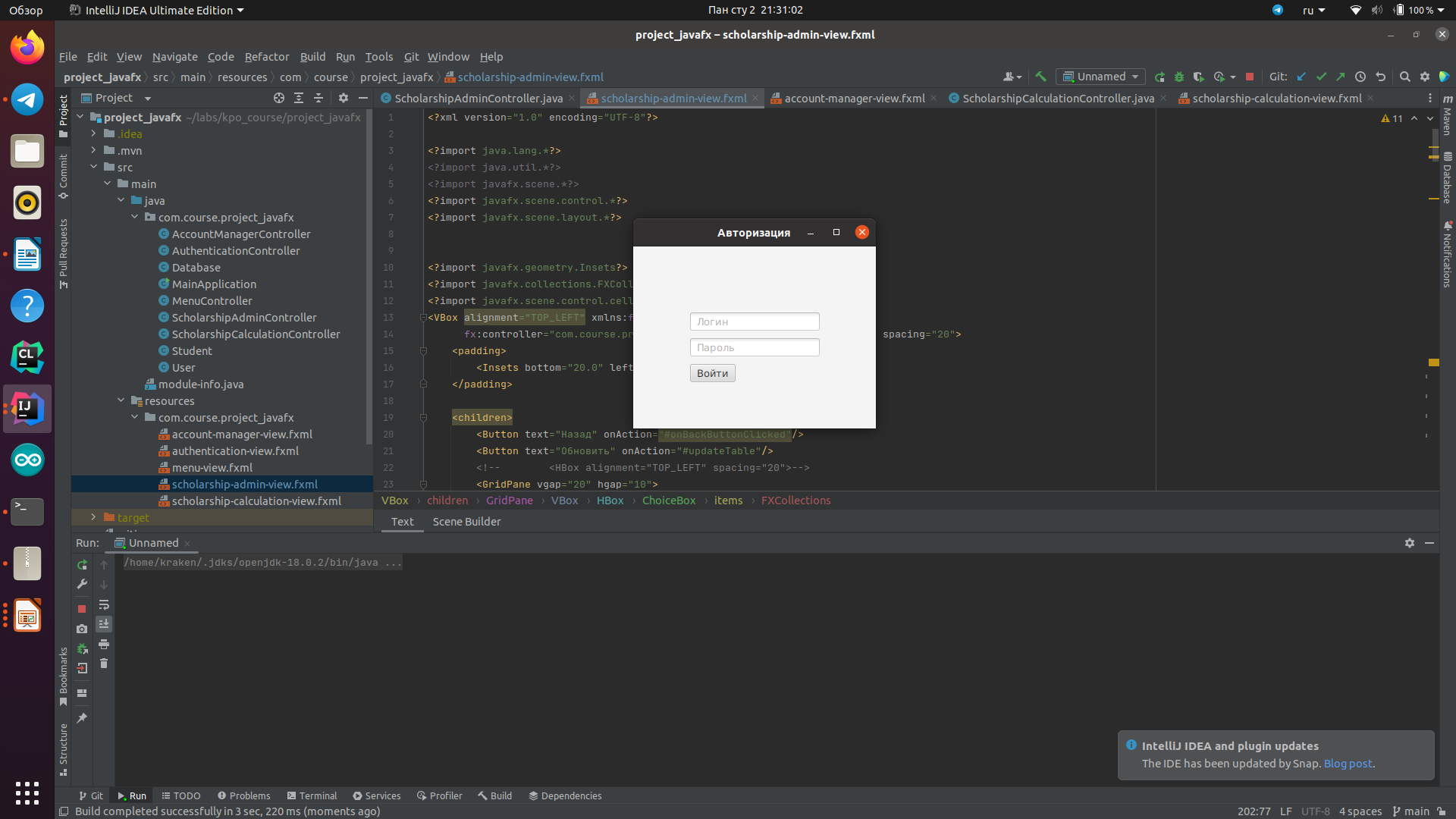
*Рисунок 23 – функция удаления записи студента. (void onDeleteButtonClicked())*

1. Извлечение выделенной в таблице записи.
2. Запись не пуста?
3. Вывод ошибки (выберите запись).
4. Запрос подтверждения операции.
5. Операция подтверждена?
6. Удаление записи студента из базы данных (sqlUpdate()); updateTable().



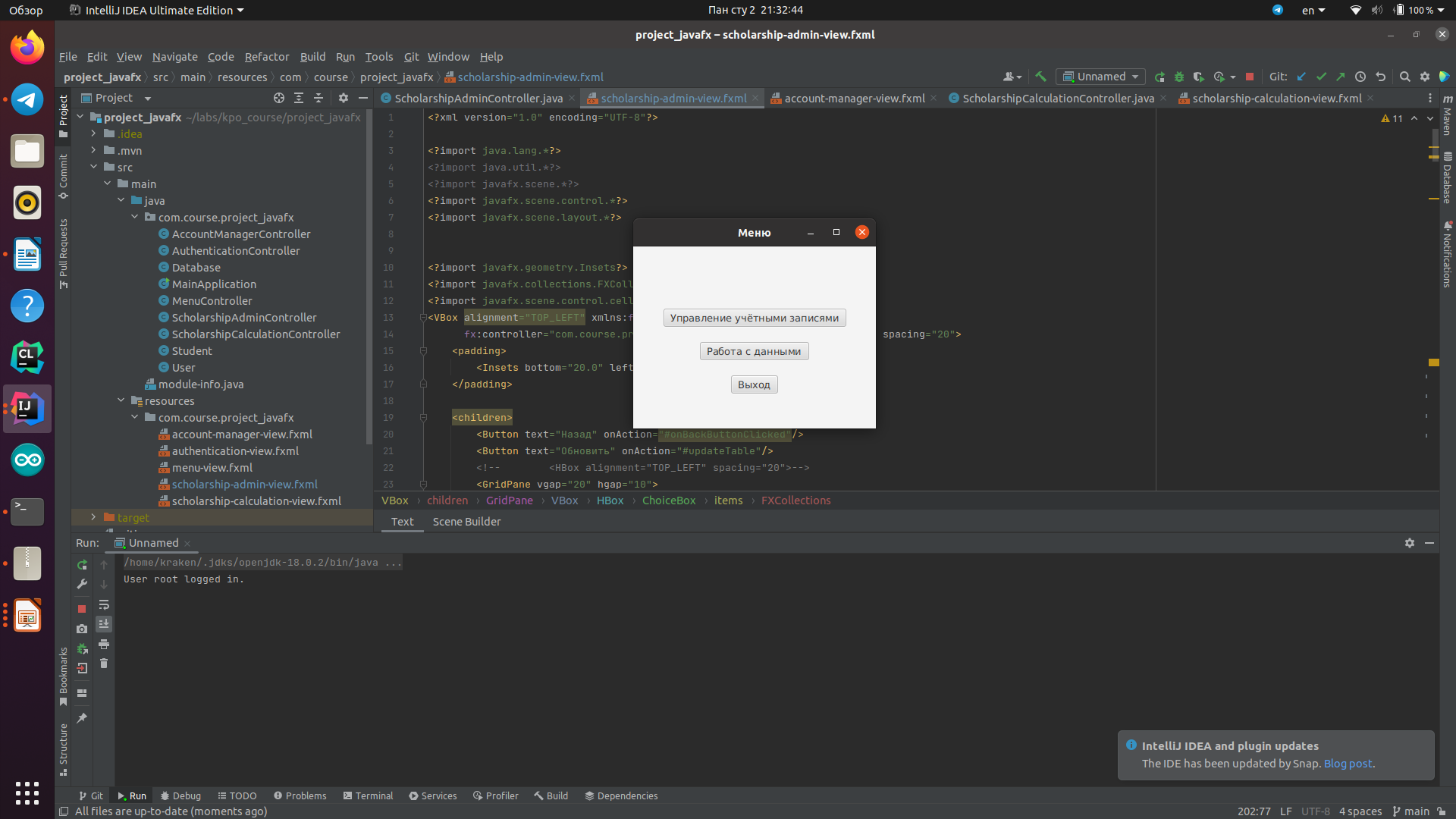
*Рисунок 24 – функция редактирования записи студента. (void onEditButtonClicked())*

1. Извлечений значений из полей формы.
2. Поля не пусты и запись в таблице выделена?
3. Вывод ошибки.
4. Изменение записи студента в базе данных (sqlUpdate()); updateTable().
5. **РАЗРАБОТКА И ОПИСАНИЕ РАБОТЫ ПРОГРАММЫ**



*Рисунок 25 – Модуль авторизации.*

При запуске программы пользователя встречает окно авторизации (рис. 23). В окне авторизации пользователь логин и пароль. Пароль хешируется. Отправляется запрос в таблицу пользователей. Если в таблице нет записи с соответствующим логином или пароли не совпадают, то будет выведена ошибка о неверно введённых данных.



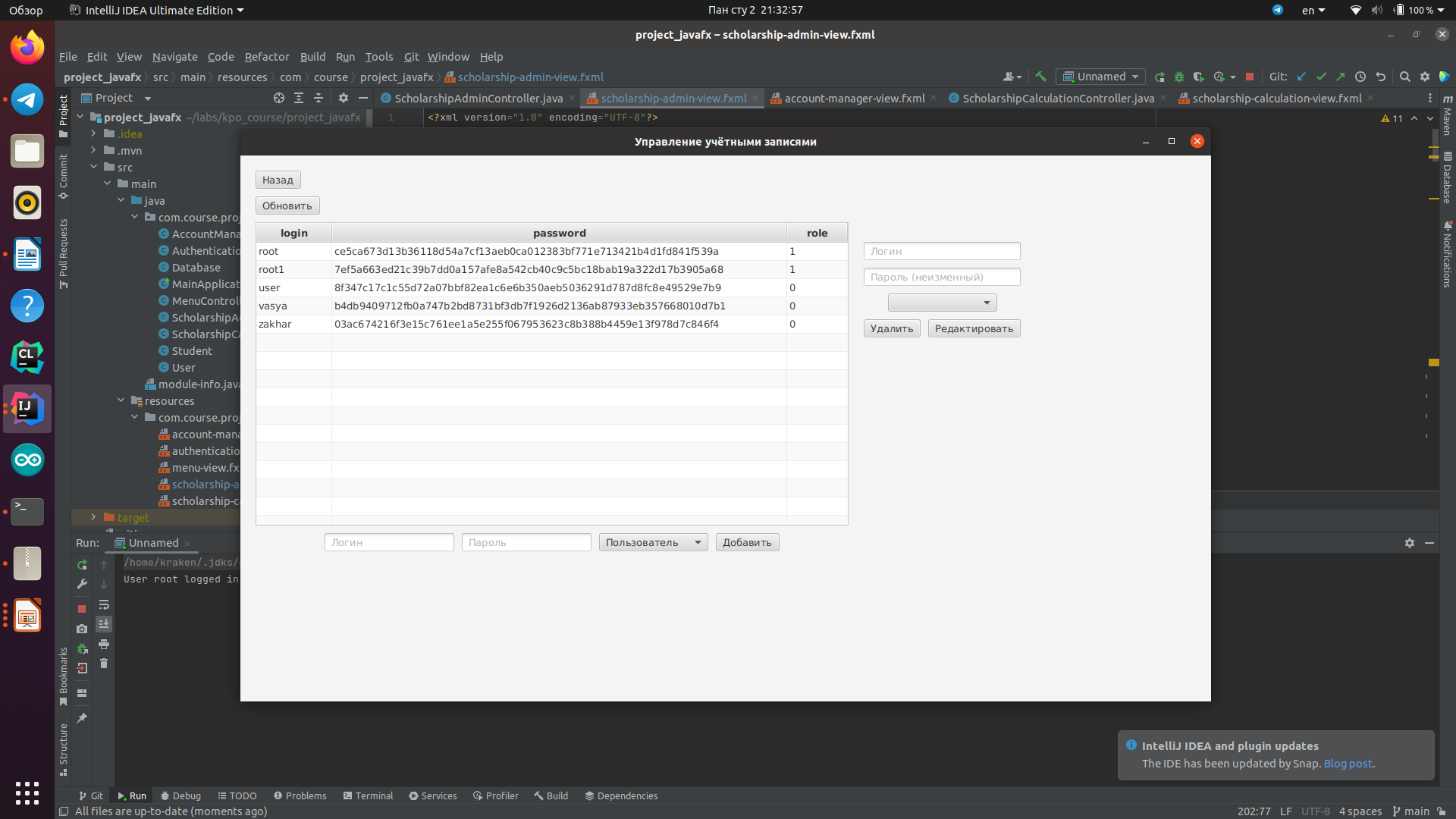
*Рисунок 26 – меню.*

После авторизации мы переходим к меню выбора подмодулей.

Кнопка «Управление учётными записями» - переход к подмодулю управления учётными записями (рис. 27).

Кнопка «Работа с данными» - переход к подмодулю работы с данными модуля пользователя (рис. 31).

Кнопка «Выход» - выход из учётной записи, возврат к модулю авторизации.



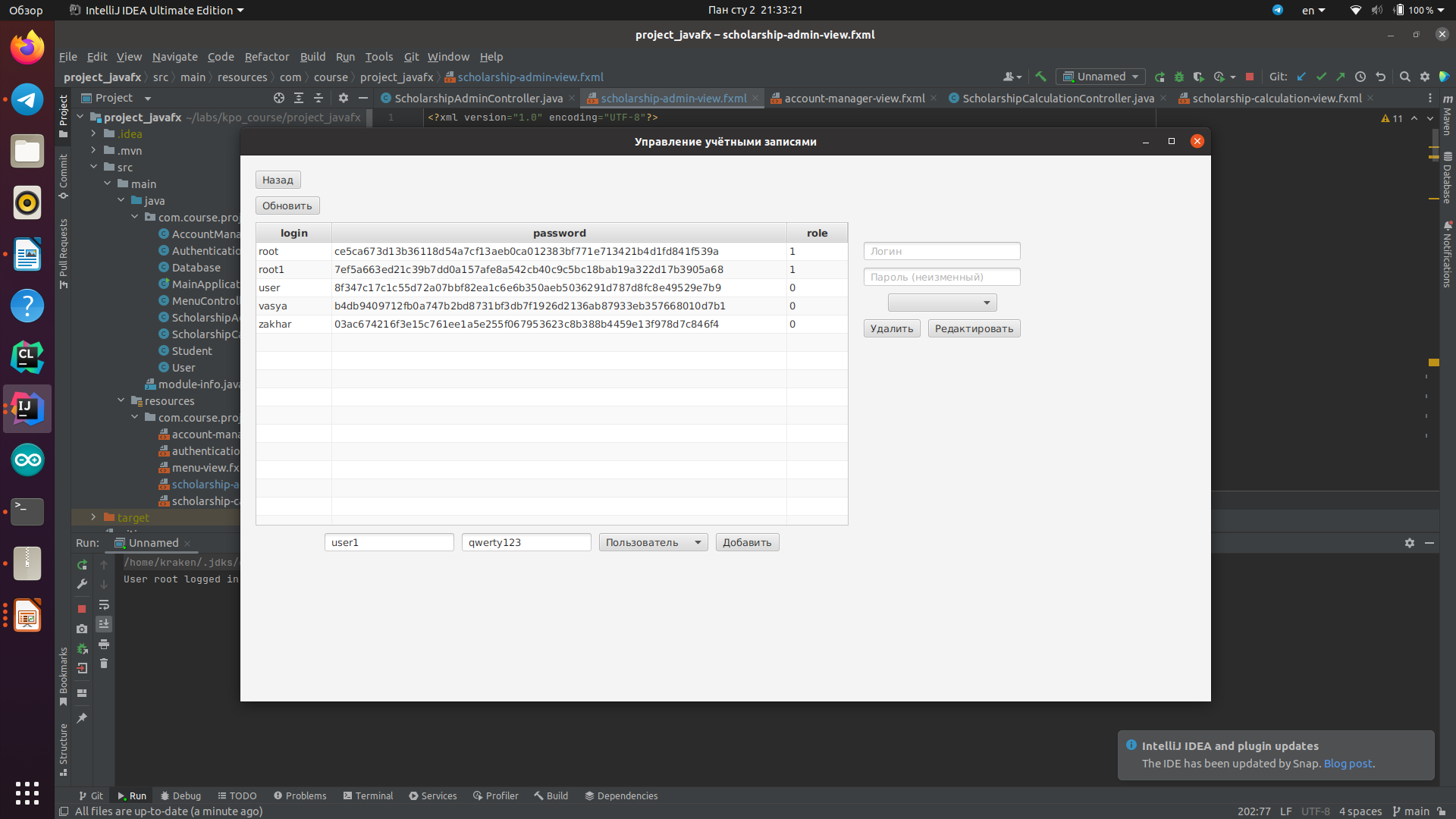
*Рисунок 27 - подмодуль управления учётными записями пользователей*

Кнопка «Назад» - возврат к меню.

Кнопка «Обновить» - обновление содержимого таблицы пользователей.

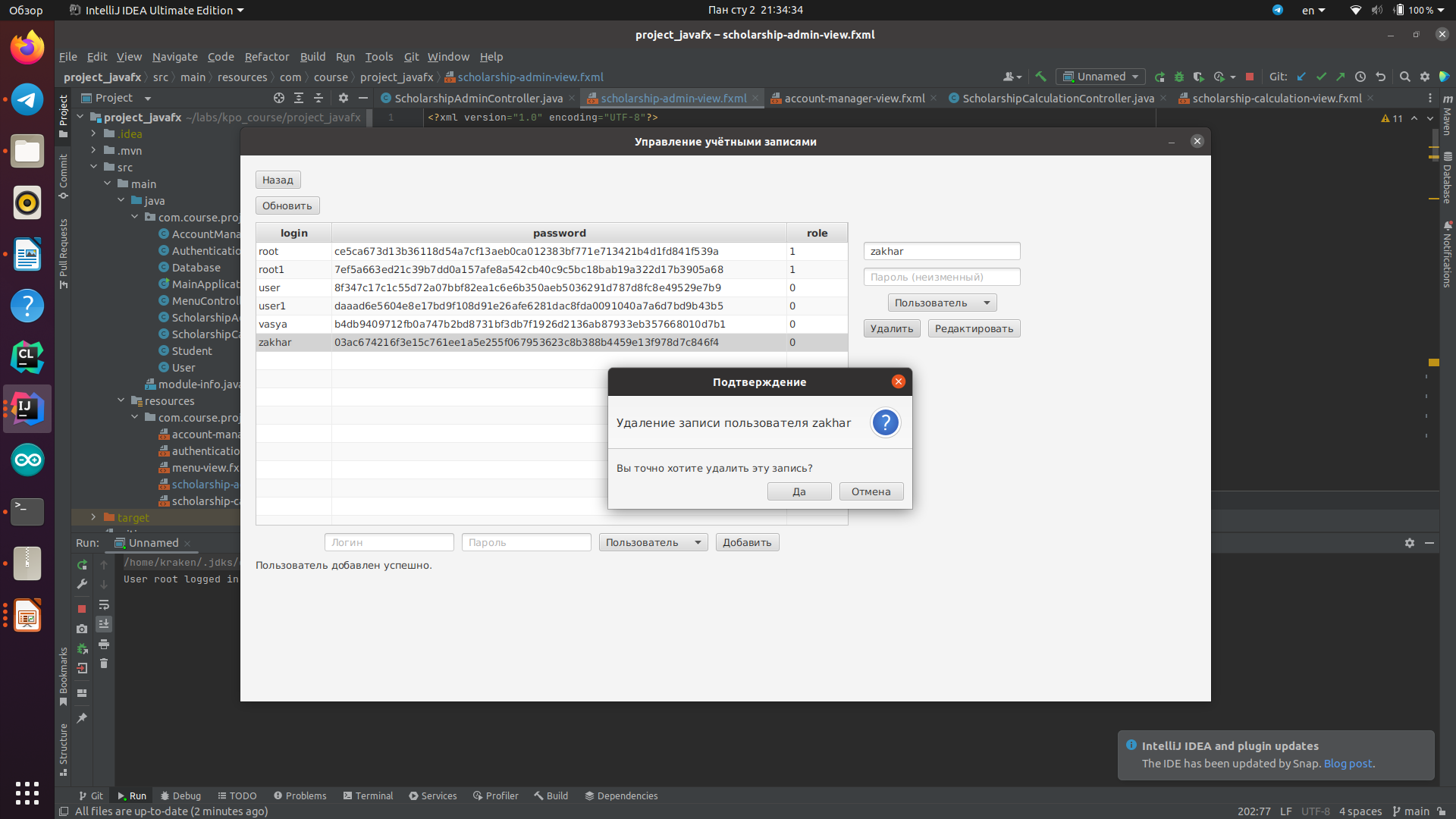
Окно также содержит форму добавления записи (под таблицей) (рис. 28) и форму редактирования и удаления записи (справа от таблицы); таблицу пользователей.

При нажатии на строку в таблице, она выделяется, а в форму редактирования и удаления записи вносятся значения полей выбранной записи (рис. 29).



*Рисунок 28 – форма добавления записи.*

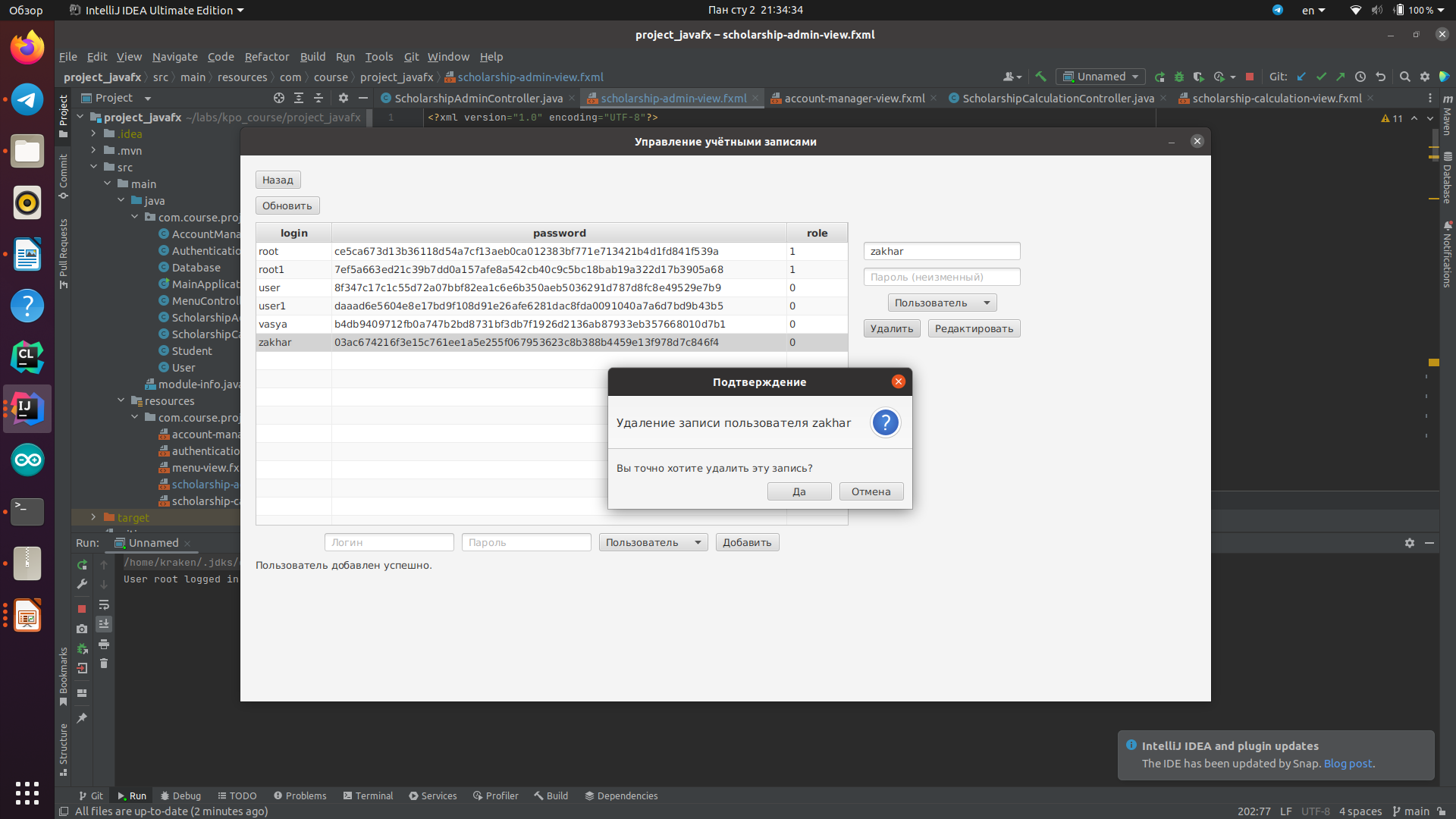
Администратор заполняет поля логина, пароля, выбирает роль нового пользователя (пользователь или администратор) и нажимает кнопку добавить. Если не все поля заполнены, выводится соответствующая ошибка. Если запись с таким логином уже существует, выводится соответствующая ошибка. Запись добавляется в таблицу базы данных. Таблица пользователей автоматически обновляется. Выводится сообщение об успешном выполнении операции.



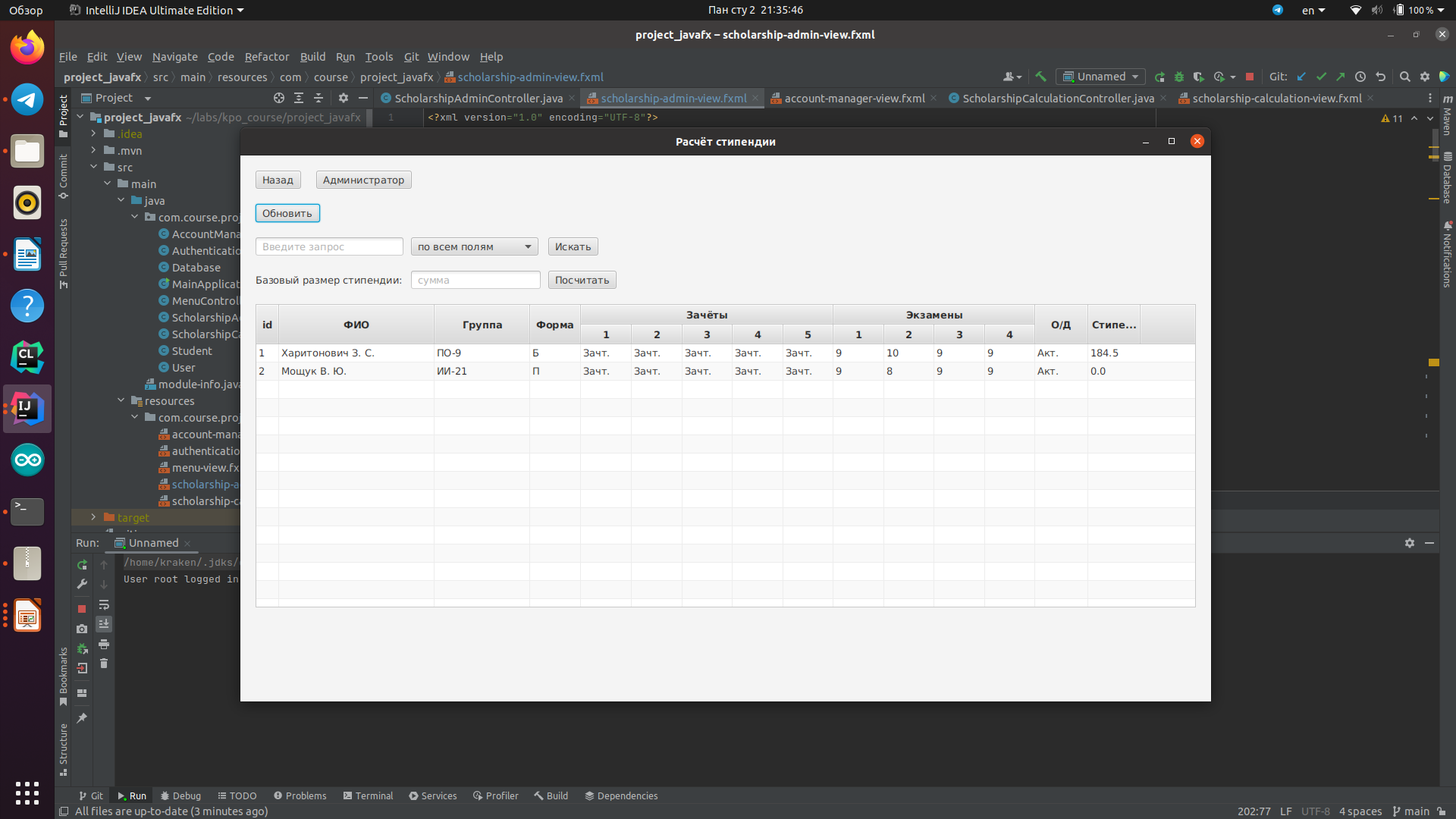
*Рисунок 29 – форма редактирования и удаления записи.*

Администратор может изменить логин, пароль, роль выбранного пользователя. Если поле пароля остаётся пустым, пароль пользователя не изменяется. Если логин изменён, но пользователь с новым логином уже существует, будет выведена соответствующая ошибка. Изменения вносятся в таблицу базы данных. Таблица пользователей автоматически обновляется. выводится сообщение об успешном выполнении операции.

Администратор может удалить выбранную запись пользователя. Если строка в таблице не выбрана, выводится соответствующая ошибка. Появляется диалоговое окно подтверждения операции (рис. 30). После того, как администратор подтверждает удаление, запись удаляется из таблицы базы данных. Таблица пользователей автоматически обновляется. Выводится сообщение об успешном выполнении операции.



*Рисунок 30 – окно подтверждения операции.*



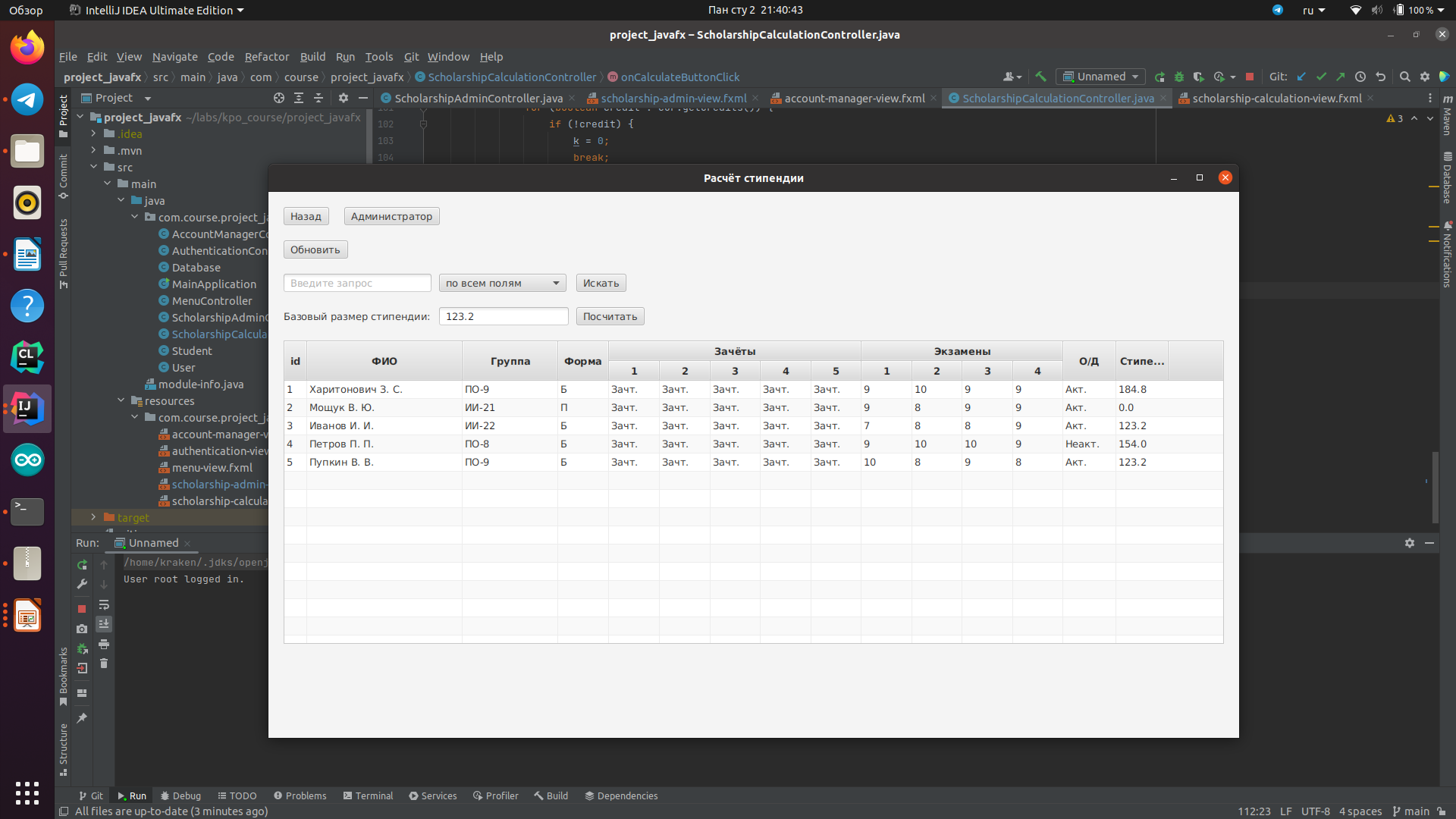
*Рисунок 31 – подмодуль работы с данными модуля пользователя.*

Кнопка «Назад» - возврат к меню.

Кнопка «Администратор» - переход к подмодулю работы с данными модуля администратора (рис. 34).

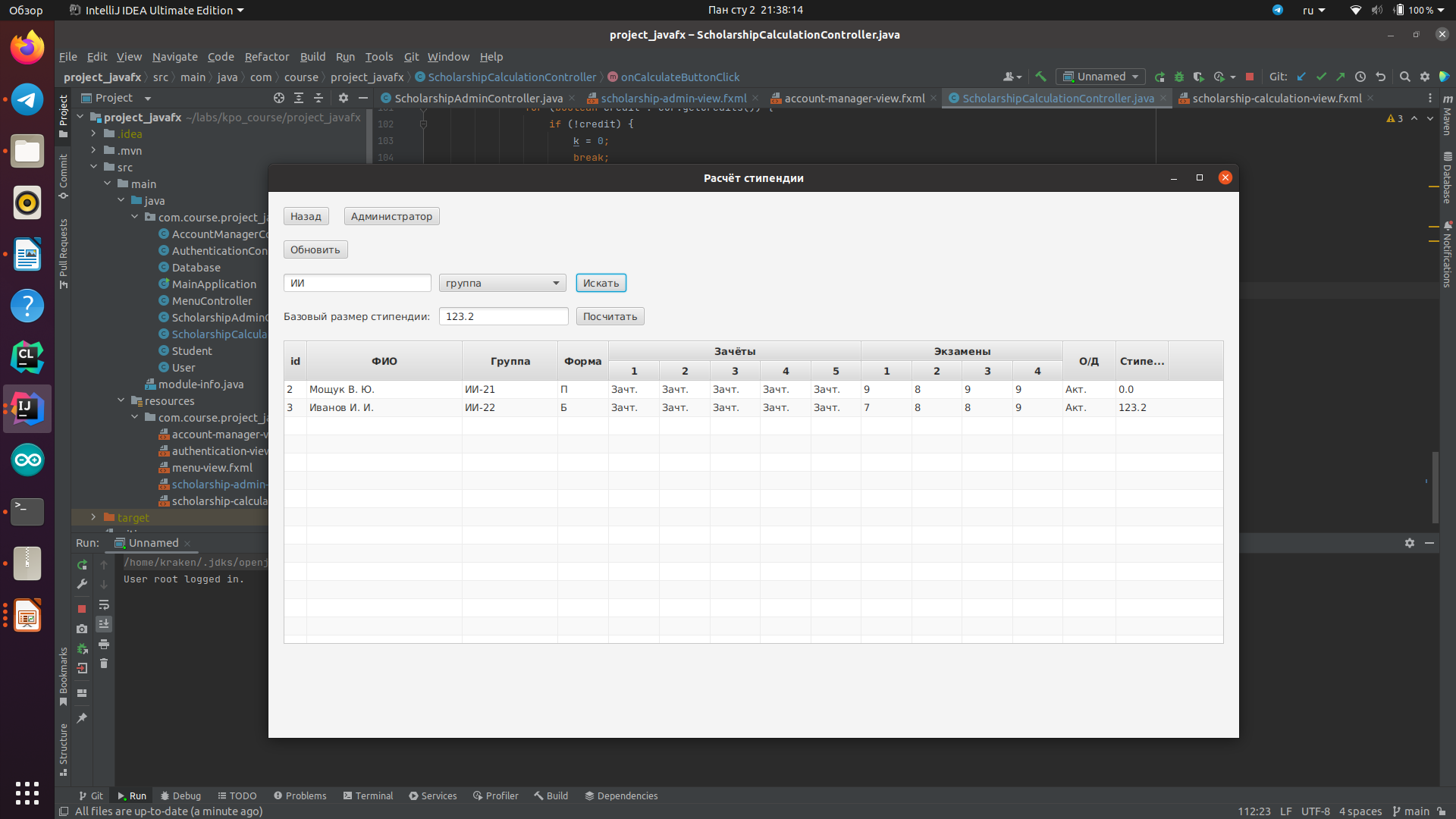
Кнопка «Обновить» - обновление содержимого таблицы студентов.

Окно также содержит форму расчёта стипендии (рис. 32) и форму поиска по данным (рис. 33) с соответствующими функциональными возможностями; таблицу студентов.



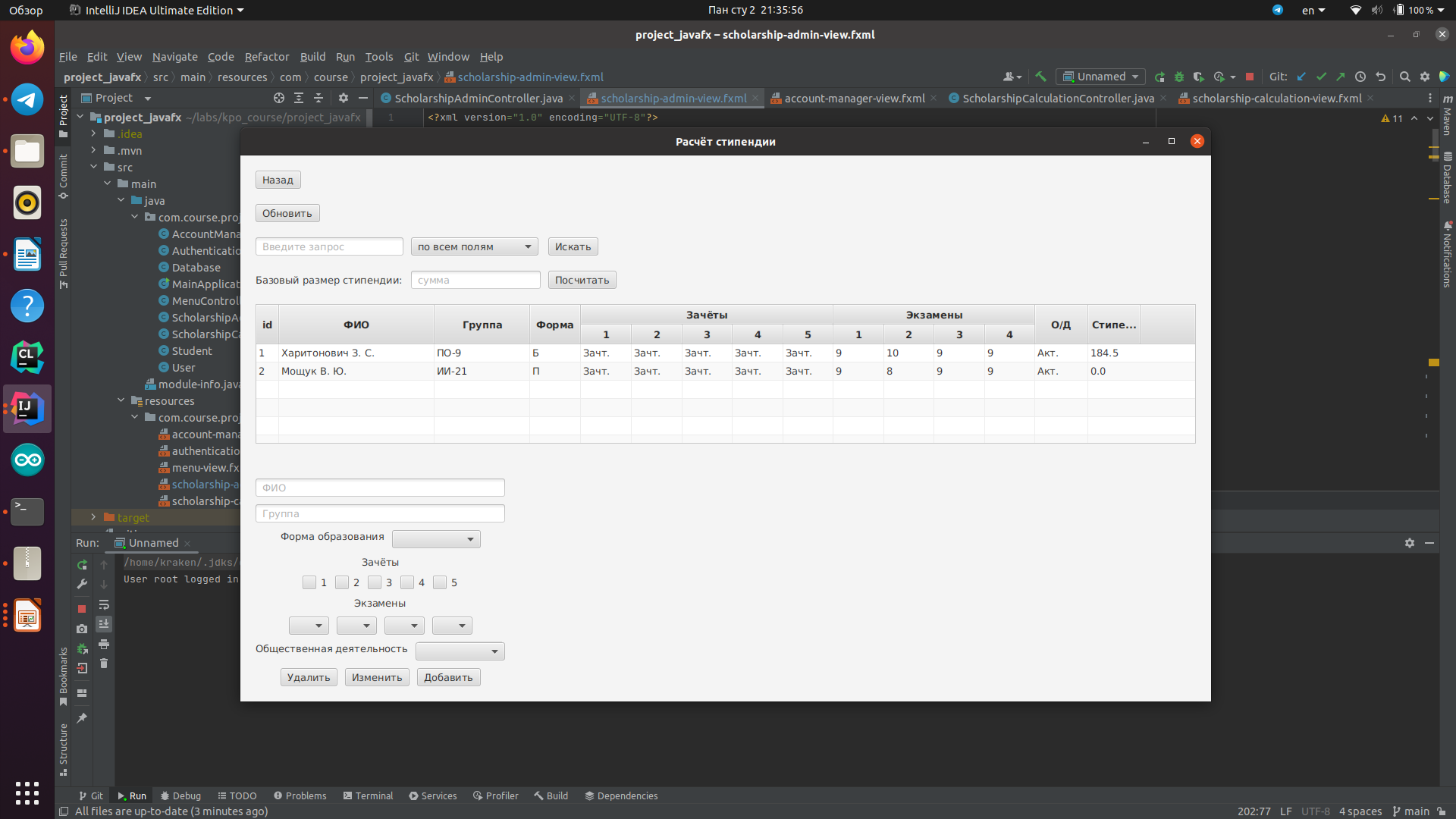
*Рисунок 32 – форма расчёта стипендии.*

Пользователь вводит размер базовой стипендии и нажимает кнопку «Посчитать». Происходит перебор всех записей, для каждого студента определяется коэффициент повышения стипендии и рассчитывается сумма. Итоговая сумма добавляется к записи студента в таблице базы данных. Таблица студентов автоматически обновляется.



*Рисунок 33 – форма поиска по данным.*

Пользователь вводит поисковой запрос, выбирает поле, по которому должен осуществляться поиск, и нажимает кнопку «Искать». Происходит перебор записей студентов. Если запись соответствует условию поиска, она выводится в таблицу.



*Рисунок 34 – подмодуль работы с данными модуля администратора.*

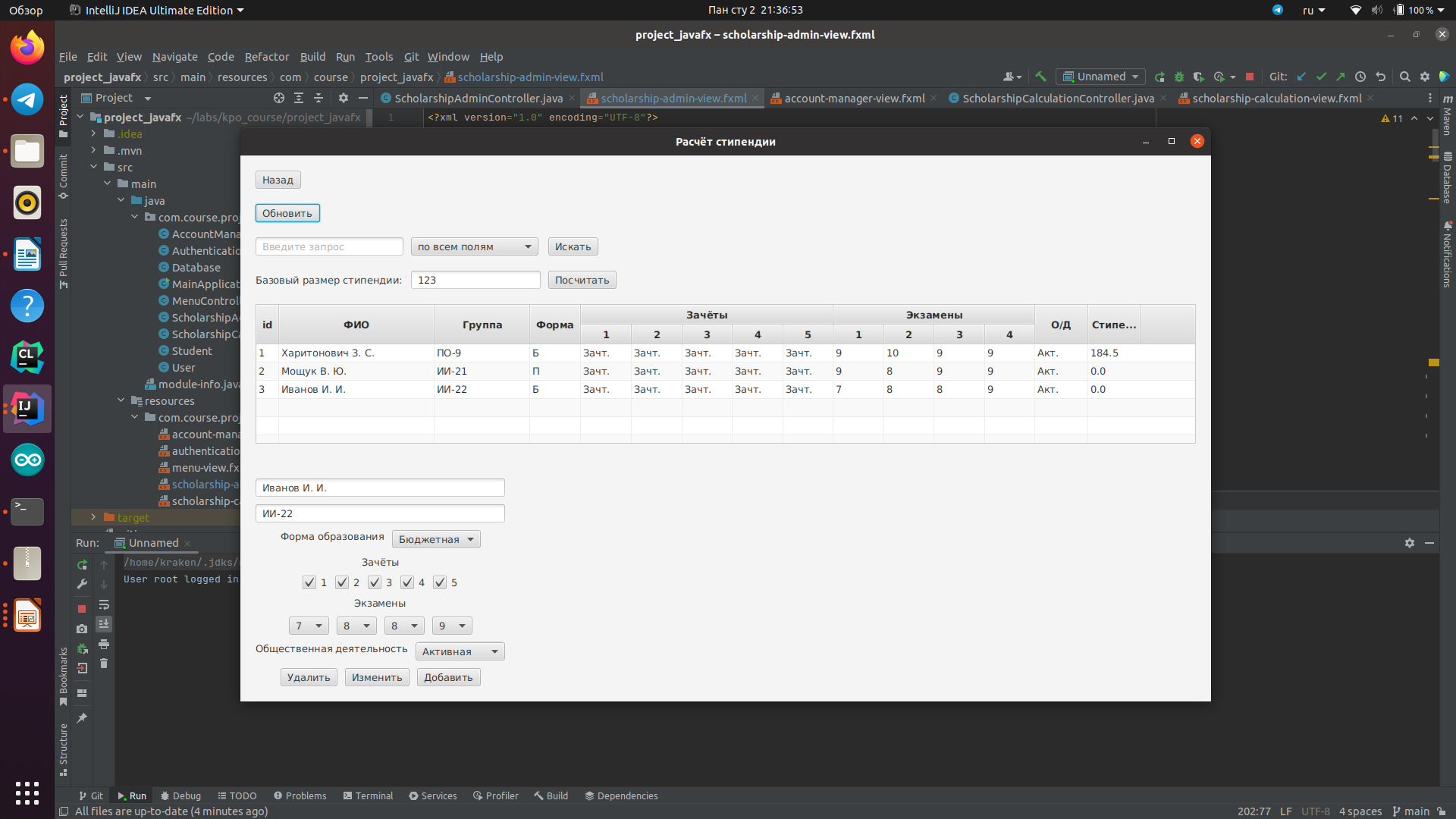
Кнопка «Назад» - возврат к меню.

Кнопка «Администратор» - переход к подмодулю работы с данными модуля администратора (рис. 34).

Кнопка «Обновить» - обновление содержимого таблицы студентов.

Окно также содержит форму расчёта стипендии (рис. 32) и форму поиска по данным (рис. 33) с соответствующими функциональными возможностями; таблицу студентов; форму добавления, редактирования и удаления.

При нажатии на строку в таблице, она выделяется, а в форму добавления, редактирования и удаления записи вносятся значения полей выбранной записи (рис. 34).



*Рисунок 34 – форма добавления, редактирования и удаления записи.*

Администратор заполняет поля и нажимает кнопку добавить. Если не все поля заполнены, выводится соответствующая ошибка. Запись добавляется в таблицу базы данных. Таблица студентов автоматически обновляется. Выводится сообщение об успешном выполнении операции.

Администратор может изменить значения полей выбранного студента. Если не все поля заполнены, выводится соответствующая ошибка. Изменения вносятся в таблицу базы данных. Таблица пользователей автоматически обновляется. выводится сообщение об успешном выполнении операции.

Администратор может удалить выбранную запись студента. Если строка в таблице не выбрана, выводится соответствующая ошибка. Появляется диалоговое окно подтверждения операции, аналогичное рис. 30. После того, как администратор подтверждает удаление, запись удаляется из таблицы базы данных. Таблица студентов автоматически обновляется. Выводится сообщение об успешном выполнении операции.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсовой работы было создано приложение для управления и работы с данными посредством графического интерфейса

Был получен опыт создания приложений с использованием современных технологий разработки. Получены навыки работы с графическим пользовательским интерфейсом, базой данных. Улучшены навыки программирования на языке программирования Java.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. JavaFX. Документация [Электронный ресурс]. URL: https://docs.oracle.com/javase/8/javase-clienttechnologies.htm