Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский ядерный университет «МИФИ»

ОТЧЕТ
по лабораторной работе №4 по дисциплине: «Теория и технология программирования»

Выполнила: студентка группы Б23-902 Деребас Л. И.

Проверил: Смирнов Д. С.

Москва 2025 г

Оглавление

Постановка задачи

Юные маглы! Марк Оливандер, новый владелец магазина волшебных палочек «Олливандеры» что в Косом переулке просит Вашей помощи!

Для ведения семейного бизнеса ему необходимо создать учетную систему, которая позволила бы заменить удивительную память его предшественника, Гаррика Олливандера, которая, к сожалению не передалась ему по наследству.

Вышеуказанная учетная система должна использоваться для сбора и хранения следующих данных:

1. Готовых палочек, выставленных на продажу или купленных
2. Покупателей палочек
3. Информации о поставках компонент для палочек

Система должна позволять:

a. Отслеживать состояние склада магазина
b. Заносить данные о новых изготовленных палочках
c. Заносить данные о волшебниках купивших какую-либо палочку
d. Кроме того, система должна позволять полностью очищать все данные, и давать возможность начать работу с чистого листа
e. Учетная система не должна забывать информацию при выключении, иначе как пользователь нажмет кнопку полной очистки!

Знайте же, юные маглы, палочка состоит из двух компонент – сердцевины и древесины, из которого изготавливается корпус палочки. Поставки компонент магазин старается заказывать не реже раза в неделю. Обычно в одну поставку входит 10-15 уникальных позиций, но летом, при подготовке к школьному сезону количество заказов увеличивается, и приходится массово заказывать ходовые сорта древесины.

При выполнении данной работы маглы (студенты) могут пользоваться любыми СУБД на их усмотрение. Для выявления функциональных требований к программе рекомендуется построить расширенную диаграмму прецедентов и деятельности. Отчет о выполнении работы должен, помимо прочих диаграмм, ОБЯЗАТЕЛЬНО содержать ER-диаграммы, на 3-х уровнях – концептуальном, логическом и физическом.

Ссылка на GitHub

https://github.com/Shirouky/lab4-Java

Диаграммы

Рис 1. DFD-диаграмма 0 уровня

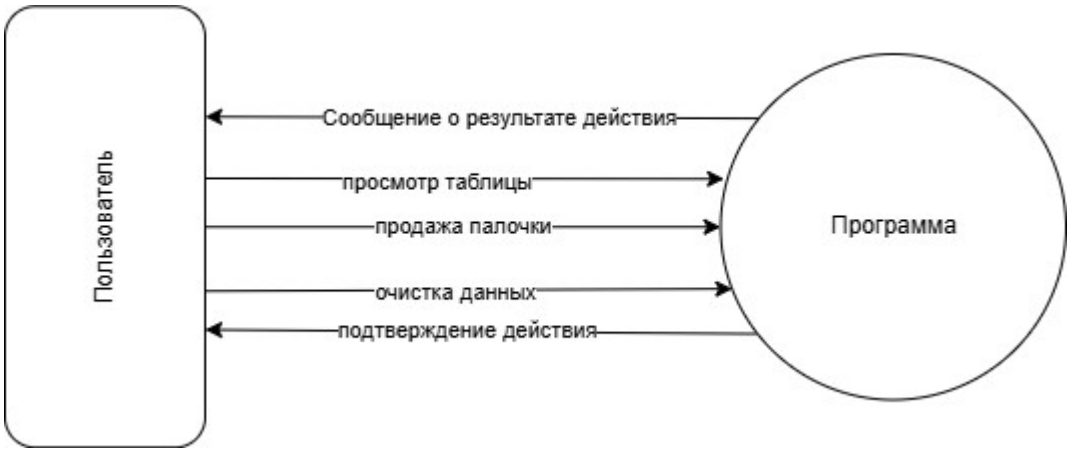
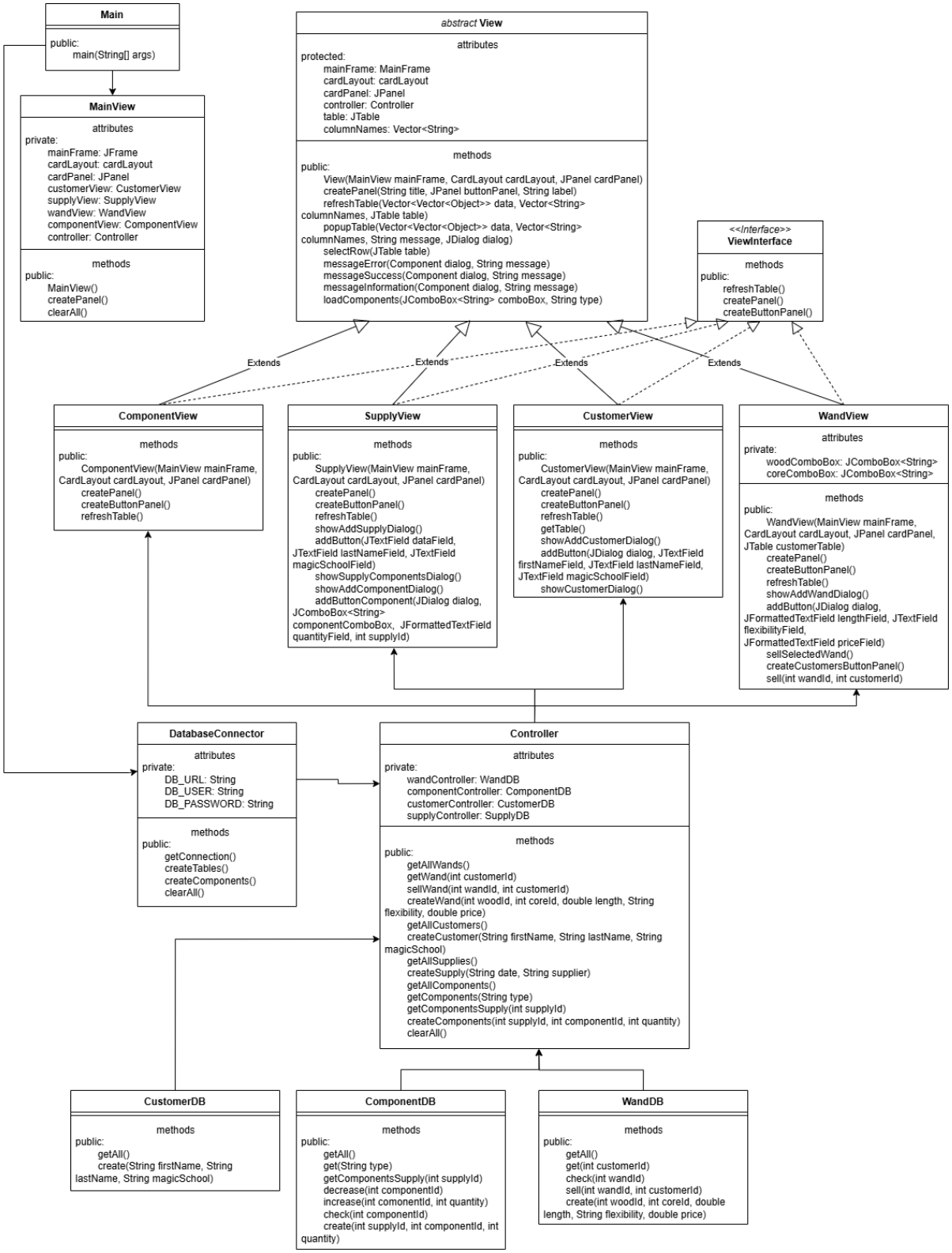


Рис 2. UML-диаграмма



ER-диаграммы

Рис 3. Концептуальный уровень

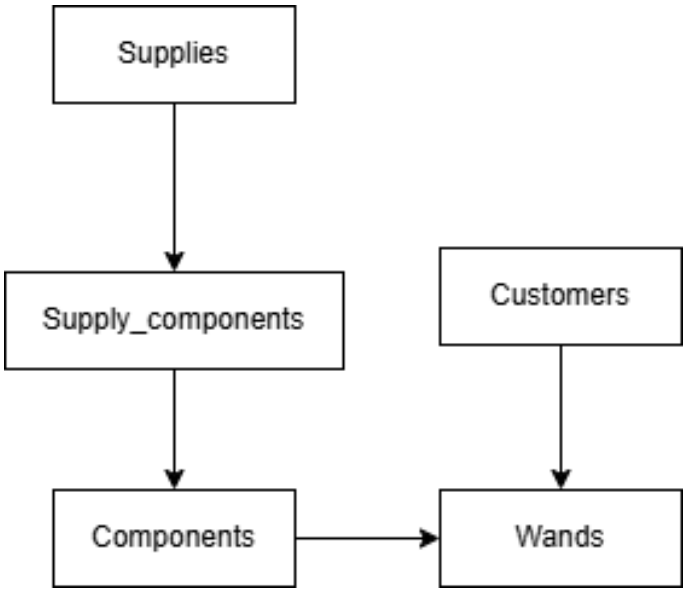


Рис 4. Логический уровень

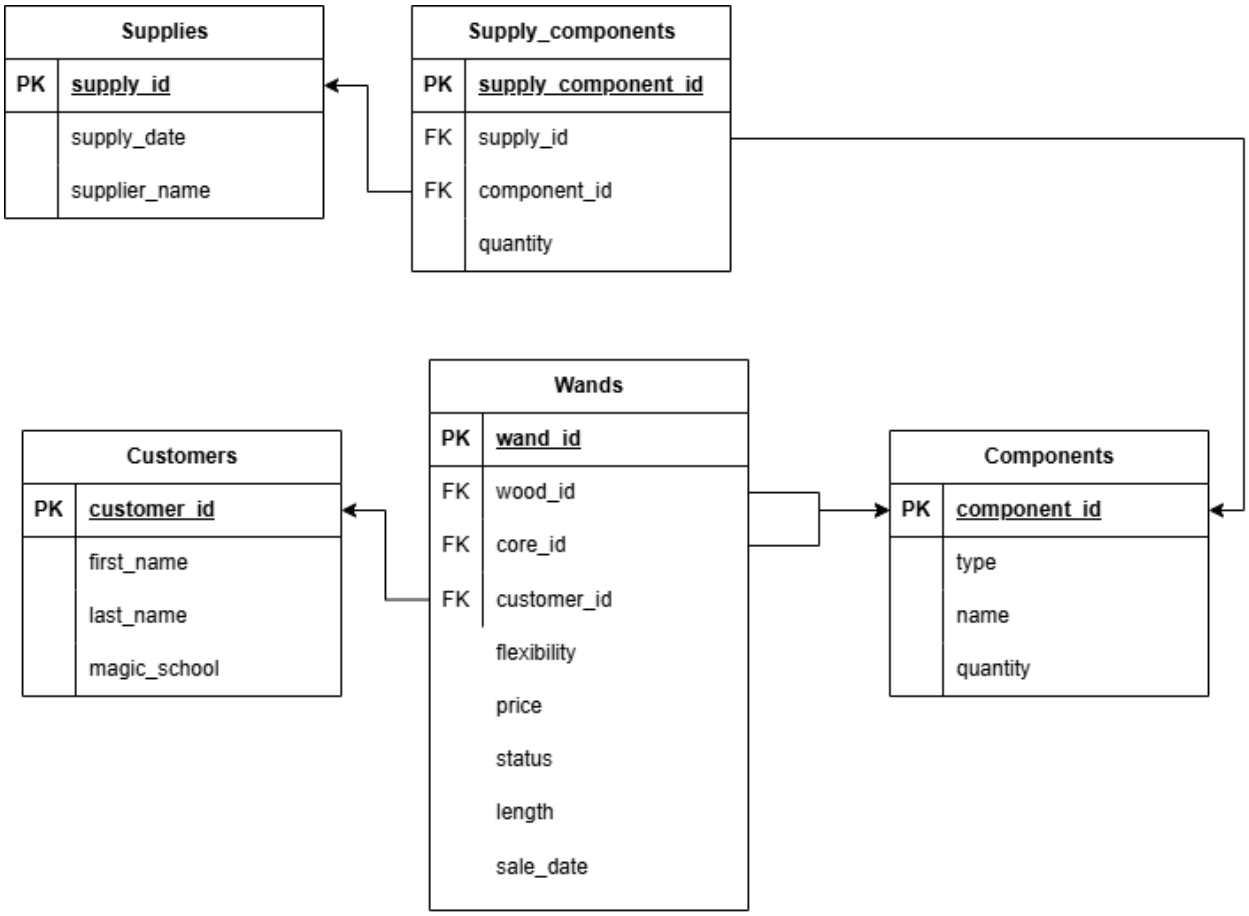
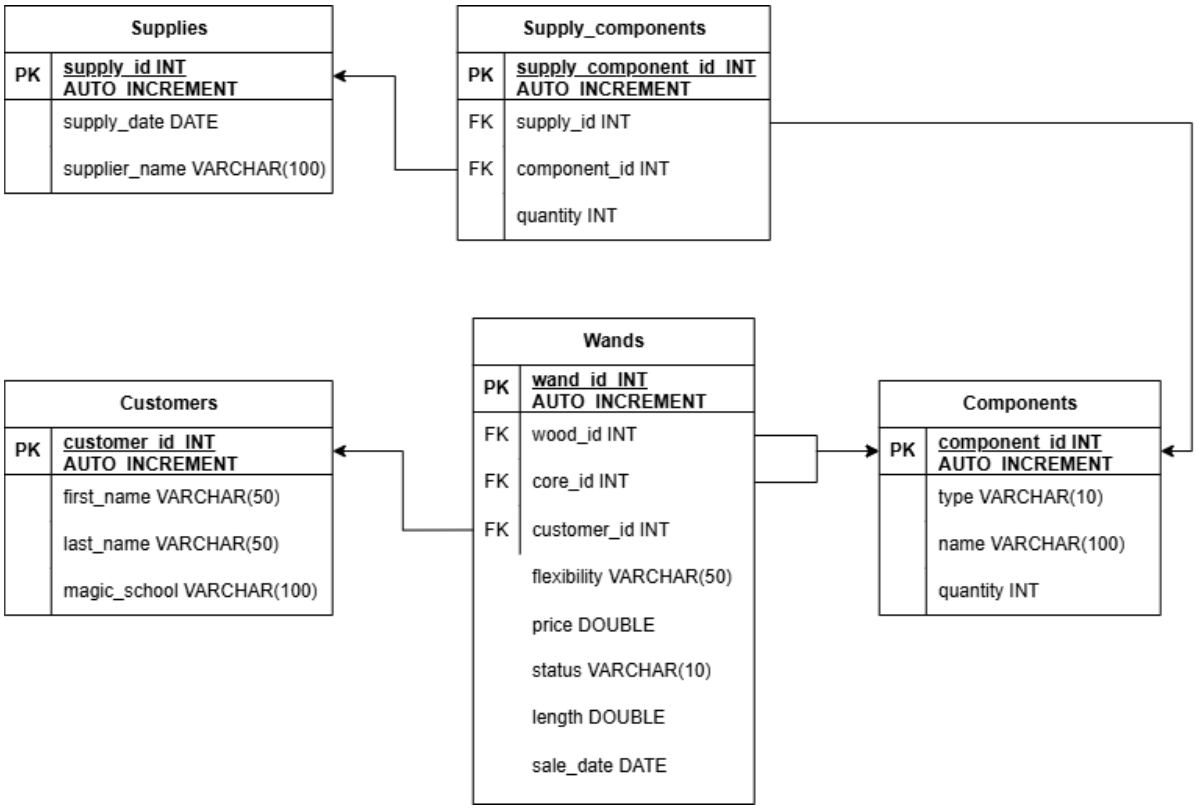


Рис 5. Физический уровень



Интерфейс

Рис 6. Главное окно

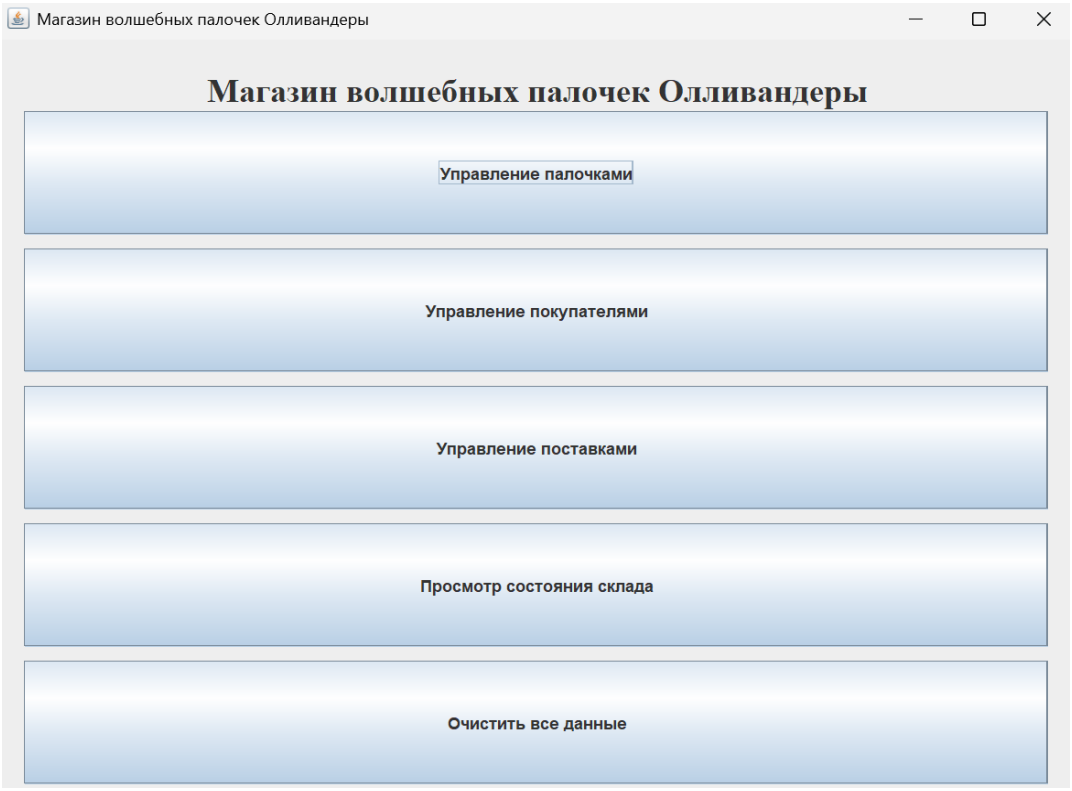


Рис 7. Создание новой палочки и таблица палочек

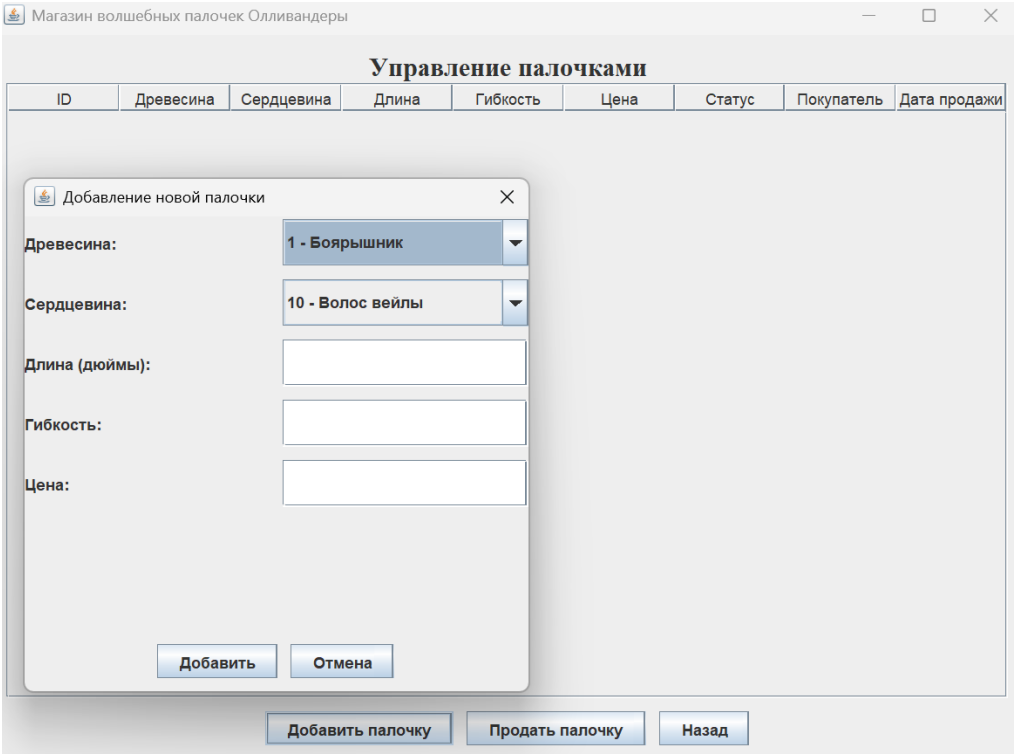


Рис 8. Создание нового покупателя и таблица покупателей

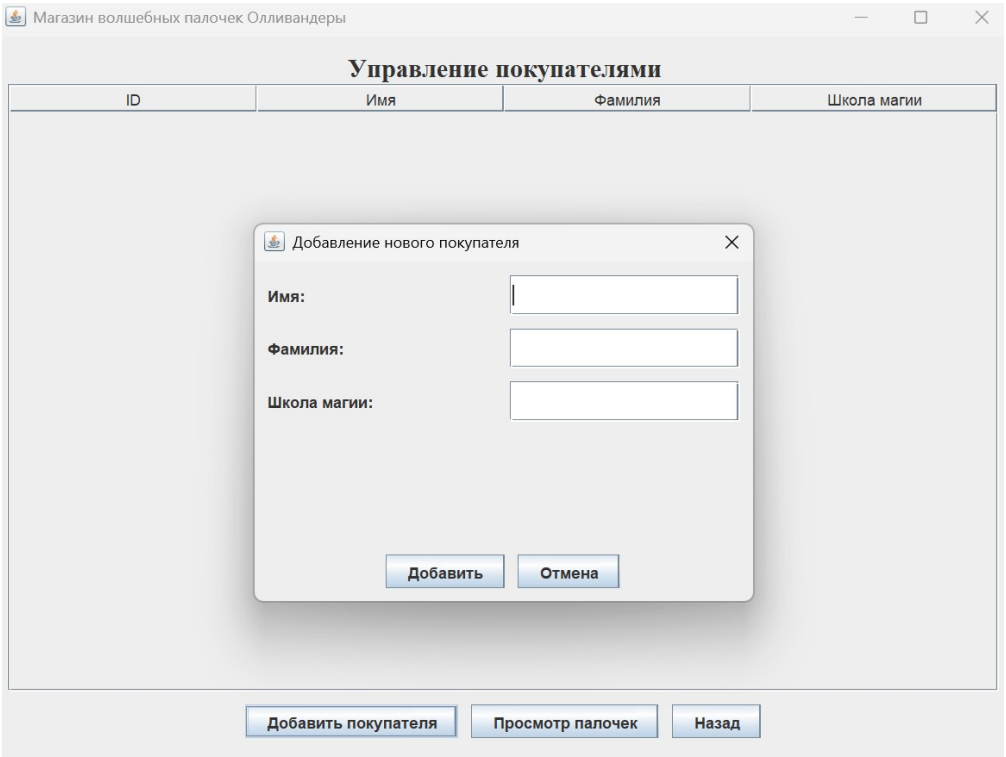


Рис 9. Создание новой поставки и таблица поставок

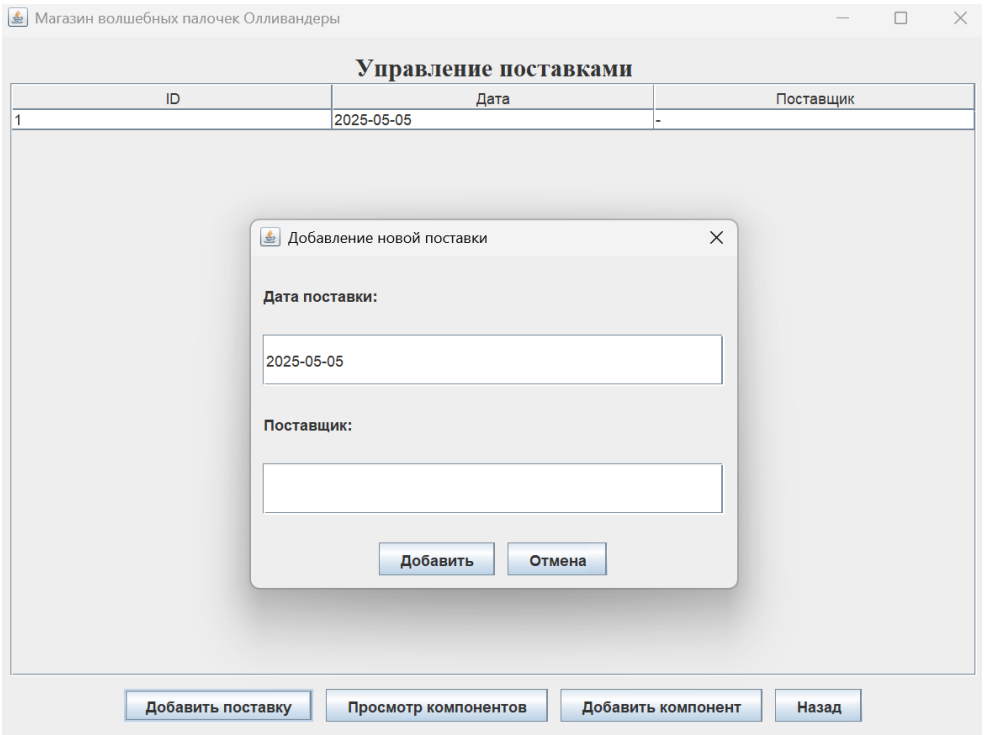
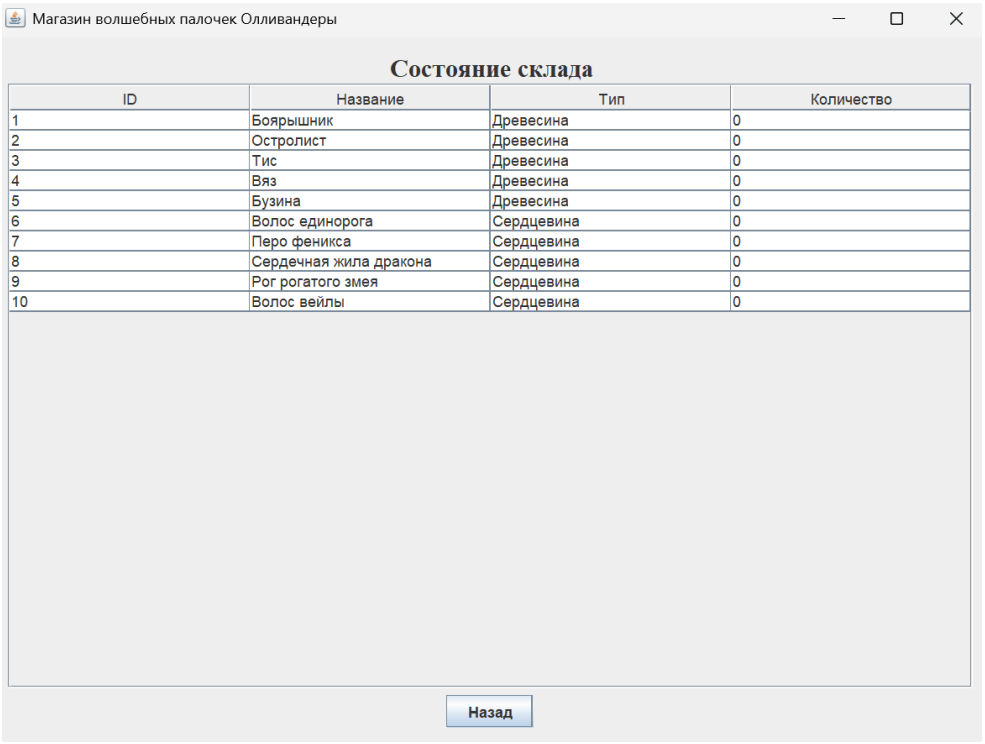


Рис 10. Создание склада



Важные части кода

Рис 11. Структура кода

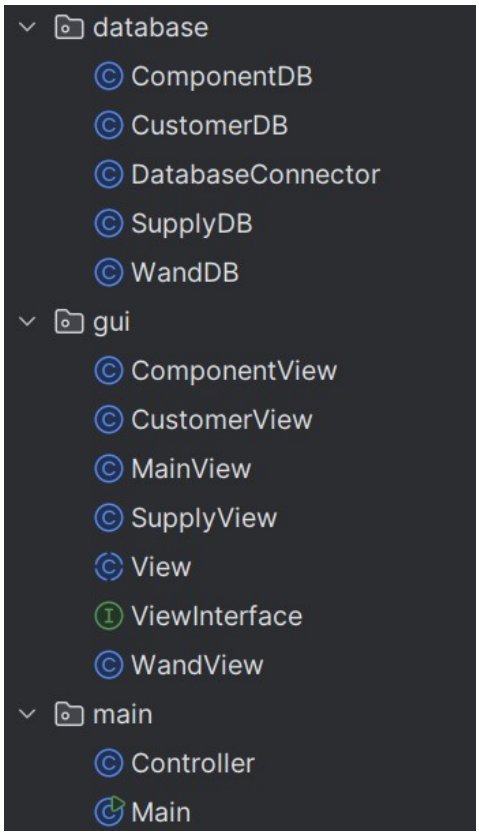


Рис 12. Пример реализации классов View

```
public class ComponentView extends View implements ViewInterface{  2 usages  Shirouky
    public ComponentView(MainView mainFrame, CardLayout cardLayout, JPanel cardPanel) {  4 usages  Shirouky
        super(mainFrame, cardLayout, cardPanel);

        columnNames.add("ID");
        columnNames.add("Название");
        columnNames.add("Тип");
        columnNames.add("Количество");
    }

    @Override  4 usages  Shirouky
    public void createPanel() {
        createPanel( title: "Состояние склада", createButtonPanel(),  label: "Inventory");
        refreshTable();
    }

    @Override  4 usages  Shirouky
    public JPanel createButtonPanel() {
        JButton backButton = new JButton( text: "Назад");
        backButton.addActionListener(e -> cardLayout.show(cardPanel,  name: "MainMenu"));

        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
        buttonPanel.add(backButton);
        return buttonPanel;
    }
```

Рис 13. Пример реализации классов DB

```
public class SupplyDB {  7 usages  Shirouky
    public Vector<Vector<Object>> getAll() throws SQLException {  1 usage  Shirouky
        Vector<Vector<Object>> data = new Vector<>();
        String sql = "SELECT supply_id, supply_date, supplier_name FROM supplies ORDER BY supply_date DESC";

        Statement statement = DatabaseConnector.getConnection().createStatement();
        ResultSet result = statement.executeQuery(sql);

        while (result.next()) {
            Vector<Object> row = new Vector<>();
            row.add(result.getInt( columnLabel: "supply_id"));
            row.add(result.getDate( columnLabel: "supply_date"));
            row.add(result.getString( columnLabel: "supplier_name") != null ? result.getString( columnLabel: "supplier_name") : "-");
            data.add(row);
        }

        result.close();
        statement.close();
        return data;
    }
```