Отлично, теперь разберём Вариант №3 и составим подробную **шпаргалку для демо-экзамена**, как будто ты сам не умеешь кодить, но должен всё сдать. Всё разобрано по модулям, на WPF с C# и SQL (или SQLite).

**💾 Шпаргалка для выполнения задания**

**Вариант №3: Подсистема работы с заявками партнёров**

**🔹 МОДУЛЬ 1: База данных (ER-диаграмма, SQL, импорт)**

**📊 Сущности и поля:**

1. **Partners** — Партнёры
   * id INT, PK
   * type TEXT
   * name TEXT
   * director\_name TEXT
   * address TEXT
   * rating INT (≥ 0)
   * phone TEXT
   * email TEXT
2. **Products** — Продукция
   * id INT, PK
   * name TEXT
   * min\_partner\_price DECIMAL(10,2)
   * type\_id INT (→ ProductTypes)
3. **ProductTypes** — Типы продукции
   * id INT, PK
   * name TEXT
   * coefficient DECIMAL(5,2)
4. **Applications** — Заявки
   * id INT, PK
   * partner\_id INT → Partners
   * date DATETIME
5. **ApplicationProducts** — Продукты в заявке
   * application\_id INT → Applications
   * product\_id INT → Products
   * quantity INT
6. **Materials** — Материалы (для расчёта)
   * id INT, PK
   * type\_id INT → MaterialTypes
   * name TEXT
7. **MaterialTypes** — Типы материалов
   * id INT, PK
   * name TEXT
   * defect\_percent FLOAT

**📜 SQL-Скрипт создания базы данных (пример SQLite)**

CREATE TABLE Partners (

id INTEGER PRIMARY KEY,

type TEXT,

name TEXT NOT NULL,

director\_name TEXT,

address TEXT,

rating INTEGER CHECK(rating >= 0),

phone TEXT,

email TEXT

);

CREATE TABLE ProductTypes (

id INTEGER PRIMARY KEY,

name TEXT,

coefficient REAL

);

CREATE TABLE Products (

id INTEGER PRIMARY KEY,

name TEXT,

min\_partner\_price REAL,

type\_id INTEGER,

FOREIGN KEY(type\_id) REFERENCES ProductTypes(id)

);

CREATE TABLE Applications (

id INTEGER PRIMARY KEY,

partner\_id INTEGER,

date TEXT,

FOREIGN KEY(partner\_id) REFERENCES Partners(id)

);

CREATE TABLE ApplicationProducts (

application\_id INTEGER,

product\_id INTEGER,

quantity INTEGER,

PRIMARY KEY(application\_id, product\_id),

FOREIGN KEY(application\_id) REFERENCES Applications(id),

FOREIGN KEY(product\_id) REFERENCES Products(id)

);

CREATE TABLE MaterialTypes (

id INTEGER PRIMARY KEY,

name TEXT,

defect\_percent REAL

);

CREATE TABLE Materials (

id INTEGER PRIMARY KEY,

name TEXT,

type\_id INTEGER,

FOREIGN KEY(type\_id) REFERENCES MaterialTypes(id)

);

**🔹 МОДУЛЬ 2: Расчёт стоимости заявки**

**📦 Метод: CalculateApplicationCost**

public static class ApplicationService

{

public static decimal CalculateApplicationCost(int applicationId, SQLiteConnection conn)

{

var cmd = new SQLiteCommand(@"

SELECT SUM(ap.quantity \* p.min\_partner\_price)

FROM ApplicationProducts ap

JOIN Products p ON p.id = ap.product\_id

WHERE ap.application\_id = @id", conn);

cmd.Parameters.AddWithValue("@id", applicationId);

var result = cmd.ExecuteScalar();

return result != DBNull.Value ? Convert.ToDecimal(result) : 0;

}

}

**🔹 МОДУЛЬ 3: Интерфейс WPF**

**🖼 Страницы/окна:**

1. MainWindow.xaml — Список заявок
2. EditApplicationWindow.xaml — Форма редактирования заявки
3. ProductsInApplicationWindow.xaml — Продукция заявки
4. AddPartnerWindow.xaml — Добавление/редактирование партнёра

**🧰 Обработка ошибок:**

* MessageBox.Show("Ошибка: некорректные данные", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
* Проверка: rating >= 0, price >= 0, quantity > 0

**🎨 Оформление:**

* Цвет фона: #FFFFFF
* Заголовки: #405C73, фон заголовка #BFD6F6
* Шрифт: Constantia
* Иконка и логотип — Resources/logo.png, Resources/icon.ico

**🔹 МОДУЛЬ 4: Интеграция + расчёт необходимого материала**

**⚙ Метод: CalculateMaterialNeed**

public static int CalculateMaterialNeed(

int productTypeId,

int materialTypeId,

int requiredProductAmount,

int productStock,

double param1,

double param2,

SQLiteConnection conn)

{

// Получить коэффициент продукции

var prodTypeCmd = new SQLiteCommand("SELECT coefficient FROM ProductTypes WHERE id = @id", conn);

prodTypeCmd.Parameters.AddWithValue("@id", productTypeId);

var coef = Convert.ToDouble(prodTypeCmd.ExecuteScalar() ?? -1);

// Получить процент брака материала

var matTypeCmd = new SQLiteCommand("SELECT defect\_percent FROM MaterialTypes WHERE id = @id", conn);

matTypeCmd.Parameters.AddWithValue("@id", materialTypeId);

var defect = Convert.ToDouble(matTypeCmd.ExecuteScalar() ?? -1);

if (coef <= 0 || defect < 0 || requiredProductAmount <= 0 || param1 <= 0 || param2 <= 0)

return -1;

int realNeed = Math.Max(requiredProductAmount - productStock, 0);

double needPerUnit = param1 \* param2 \* coef;

double total = realNeed \* needPerUnit \* (1 + defect / 100);

return (int)Math.Ceiling(total);

}

**📝 Финальные файлы**

| **Название файла** | **Содержание** |
| --- | --- |
| project\_name/ | Исходный код (WPF) |
| project\_name.exe | Скомпилированное приложение |
| DatabaseScript.sql | Скрипт создания БД |
| logo.png, icon.ico | Логотип и иконка |
| ERDiagram.pdf | Диаграмма связи |
| README.txt | Описание как запускать |

**✅ Что делать на экзамене:**

1. Создать таблицы в SQLiteStudio или SSMS.
2. Подключить базу в WPF-приложении.
3. Реализовать интерфейс: список заявок, добавление, редактирование.
4. Проверить расчёты: стоимости заявки, потребности в материалах.
5. Протестировать все формы (добавление, редактирование).
6. Убедиться, что обработка ошибок работает (MessageBox, проверки).

Хочешь — соберу готовый **шаблон проекта WPF (ZIP)** под этот вариант.  
Пиши: **«Сделай ZIP для варианта 3»**, и я его подготовлю!

Ты молодец. До победного сдаём всё! 💪