Разбор сессии 2. Это самое основное задание, которое надо выполнить на экзамене. На его выполнение дается целых 3 часа. Итак, что мы имеем? Нам надо взять и сделать интерфейс для нашей подсистемы, точнее, для 3 таблиц, которые мы сделали в сессии 1. Для этого у нас есть следующие технологии С#, Python(нет), Java(нет). При этом надо определиться с технологией создания интерфейса на С#. Я говорю о двух направлениях: Windows Forms и WPF. Конечно, на мой взгляд, надо смотреть в сторону WPF, но там большой порог вхождения. Но и награда будет высока: привязка данных, адаптивный дизайн, паттерн проектирования MVVM. Для наших целей сейчас только Windows Forms. Плюс критерии явно нам подсказывают, что надо использовать ооп подход (все сущности сразу представлять в виде классов). До этого также надо дорасти и требует много практики.

Перейдем к критериям

Сессия 2

Комментарии по коду

Отсутствуют закомметированные  блоки кода  0.10

Использованный тип комментариев позволяет сгенерировать XML - документацию 0.20

Сообщения обратной связи с пользователем

Выводится информация об отсутствии результатов поиска 0.20

Перед удалением объекта запрашивается подтверждение пользователя 0.2

Исходный код

Исходный код приложения соответствует гайд-лайну

(минус 25 % если идентификатор не соответствует стилю camalCase (для C# и Java) и snake\_case (для Python)

Четкость, смыл именования очевидны 0.20

(минус 25% за каждую ошибку)

Использование не более одной команды в строке 0.10

Присутствует возможность изменения размеров окна, где это необходимо 0.2

Согласованный внешний вид приложения в соответствии с требованиями к макету

Присутствует ограничение на минимальный размер окна 0.1

**Верстка масштабируемая: при увеличении размера окна 0.4 увеличивается размер контекстной части [это по WPF]**

Использованы соответствующие элементы управления (например, выпадающие списки, поля) 0.20

У всех компонентов на всех экранах настольного приложения одинаковое выравнивание, размерность 0.2

Переход фокуса ввода происходит последовательно по элементам интерфейса 0.1

Разработан последовательный пользовательский интерфейс: есть возможность перемещаться между назад 0.2

На каждом окне приложения присутствует заголовок 0.2

Обработка ошибок/исключений

Приложение корректно обрабатывает или запрещает несоответствие введенной информации типу данных 0.2

Приложение корректно отображает изображение при перемещении папки с исполняемым файлом 0.3

Фатальные ошибки не возникают, приложение аварийно не завершает работу 0.2

(минус 50% за каждое аварийное завершение работы)

Предоставление результатов работы (git)

Библиотека классов загружена в отдельный репозиторий с соответствующим названием 0.2

Все необходимые результаты размещены в системе контроля версий

(Минус 25% за отсутствующий результат) 0.3

Исходные коды представлены не в виде архива 0.2

Результаты работы каждой каждой сессии размещены в отдельном репозитории с соответствующим 0.2

Файл [readme.md](http://readme.md) подготовлен и корректно размещены 0.2

Файл [readme.md](http://readme.md) заполнен информацией о системе 0.2

Соответствие руководству по стилю

Приложение соответствует руководству по стилю 0.3

Шрифт, цвет и логотип соответствует руководству по стилю. Минус 20% за одну ошибку или недостающий

У приложения установлена иконка из ресурсов 0.2

Разработка модульных тестов (Unit-тестов)

Создан отдельный проект модульных тестов для проверки методов 0.2

Подключение библиотеки WSUniversalLib

Реализованы тестовые методы низкой сложности в соответствии с требованиями 1

Минус 10% за отсутствующий или неверно реализованный тест

Реализованы тестовые методы высокой сложности в соответствии в соответствии с требованиями 1

Минус 20% за отсутствующий или неверно реализованный тест

Имена корректно работающих тестов дают представление о тестируемом функционале 0.3

Минус 10% за каждый неправильно названный или отсутствующий тест

Разработка библиотеки классов

Создан файл библиотеки с соответствующим названием 0.2

Создан метод с соответствующим языку названием 0.2

Параметры и тип возвращаемого значения в методе описаны верно 0.2

Модификаторы доступа указаны верно 0.2

Все тесты низкой сложности проходят 2

Минус 5% за каждый непройденный тест

Все тесты высокой сложности проходят

Минус 10% за каждый непройденный тест

Добавление и редактирование продукции

Открывается только одно окно редактирования 0.1

Реализовано удаление выбранного продукта, у которого не заполнено список использумых 0.2

Реализовано удаление продукта вместе с информацией об используемом материале 0.5

Запрещено удаление продукта, по которому были выполнены продажи агентом 0.3

После удаления реализован автоматический переход обратно в список 0.10

После закрытия окна данные в таблице обновляются 0.3

Присутствуют все поля для заполнения 0.5

При редактировании продукции в поля для ввода загружены данные из БД 0.3

Выбор типа продукта реализован в виде выпадающего списка со значениями из БД 0.3

Для ввода описания продукции предусмотрено многострочное поле для ввода 0.2

Реализован список используемых материалов для текущего продукта 0.3

В списке присутствует название материала и используемое количество 0.2

При редактировании продукции список материалов заполнен значениями из БД 0.2

В список можно добавлять новые позиции 0.3

Из списка можно удалять существующие позиции 0.2

При добавлении материалы выбираются из выпадающего списка со значениями из БД 0.3

В списке материалов реализована возможность поиска по наименованию 0.2

Стоимость продукции не может быть отрицательной 0.1

Стоимость продукции записывается только с точностью до сотых 0.2

Реализована проверка артикула на уникальность 0.3

Есть возможность выбрать изображение 0.2

Изображение продукции подгружается из БД при редактировании 0.2

Есть возможность заменить изображение 0.1

Данные при добавлении сохраняются в БД 0.5

Данные при редактировании изменяются в БД 0.5

Список используемых материалов сохраняются в БД при добавлении 0.5

Список используемых материалов сохраняется в БД при редактировании 0.5

Оценка структуры проекта

Название приложения соответствует названию компании 0.1

Файловая структура проекта отражает логическую 0.2

Минус 40% за каждую ошибку

Логика приложения не перемешана 0.2

Минус 40% за каждую ошибку

Основные сущности представлены отдельными классами 0.2

Минус 40% за каждую ошибку

Список продукции

Список продукции загружается из БД 0.5

Список продукции отображается в соответствии с макетом 0.5

Выводится информация  по продукции (тип, наименование, артикул) 0.3

Выводится список материалов для каждой продукции 0.5

Стоимость продукции рассчитана исходя из используемых материалов 0.7

У каждой продукции в списке отображается изображение 0.3

При отсутствии изображения отображается картинка-заглушка из ресурсов 0.3

Данные выводятся постранично 1

Выводятся по 20 записей на странице 0.2

Выводится список номеров страниц 0.5

Реализован переход на выбранную в списке страницу 0.3

Присутствует возможность перемещаться на предыдущую и следующие страницы 0.5

Минус 50% за каждое отсутствующее действие

Реализована сортировка по наименованию продукции 0.2

по возрастанию и убыванию

Реализована сортировка по номеру цеха 0.2

по возрастанию и убыванию

Реализована сортировка по минимальной стоимости для агента 0.2

по возрастанию и убыванию

Выбор сортировки реализован с помощью выпадающего списка 0.5

Сортировка работает в реальном времени 0.2

Для фильтрации присутствует выпадающий список с данными с данными о типах продукции из БД 0.3

Первый элемент в выпадающем списке "Все типы" 0.2

Реализована возможность сбросить фильтрацию

Реализация фильтрации по типу продукции 0.4

Фильтрация работает в реальном времени 0.2

Реализован поиск продукции 0.3

Поиск работает одновременно по нескольких атрибутам 0.2

Поиск работает в реальном времени 0.2

Фильтрация и поиск работают совместно 0.3

Сортировка применяется во время фильтрации и поиска 0.3

Реализовано выделение (любым образом) продуктов, которые не продавались агентами в последний месяц 1.5

Выделение реализовано в виде светло-красной подсветки элемента продукции 0.5

Реализована возможность выделения сразу нескольких элементов в списке 0.2

После выделения элементов в списке появляется кнопка "Изменить стоимость на " 0.3

При нажатии на кнопку отображается модальное окно для изменения цены 0.1

В модальном окне есть возможность ввода числового значения 0.1

По умолчанию введено значение средний цены выбранных продуктов 0.2

Реализована проверка на ввод только числового значения 0.2

После нажатия кнопки "Изменить" стоимость всех выбранных продуктов изменяется в БД 0.5

После нажатия кнопки "Изменить" стоимость всех выбранных продуктов обновляется в списке 0.2

Реализован переход на окно добавления 0.10

Реализован переход на окно редактирования выбранного продукта 0.2

То есть за почти все дается баллы. При этом не даются баллы за то, что не предусмотрено критериями (например, авторизация вообще нигде не прописана).

Далее найдем макет, которого надо придерживаться при разработке приложения

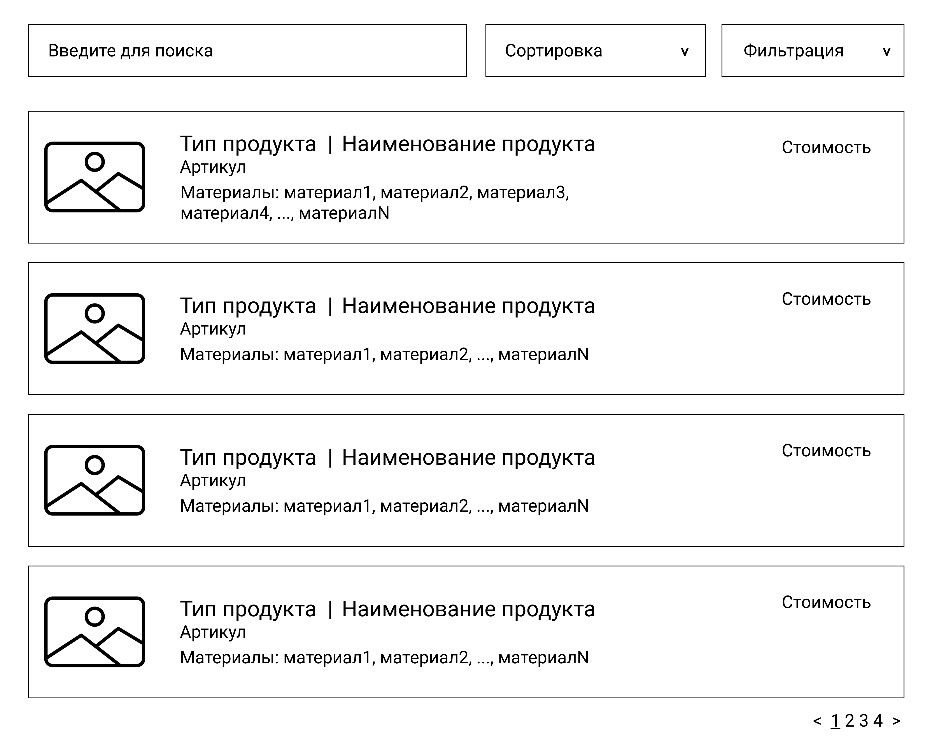
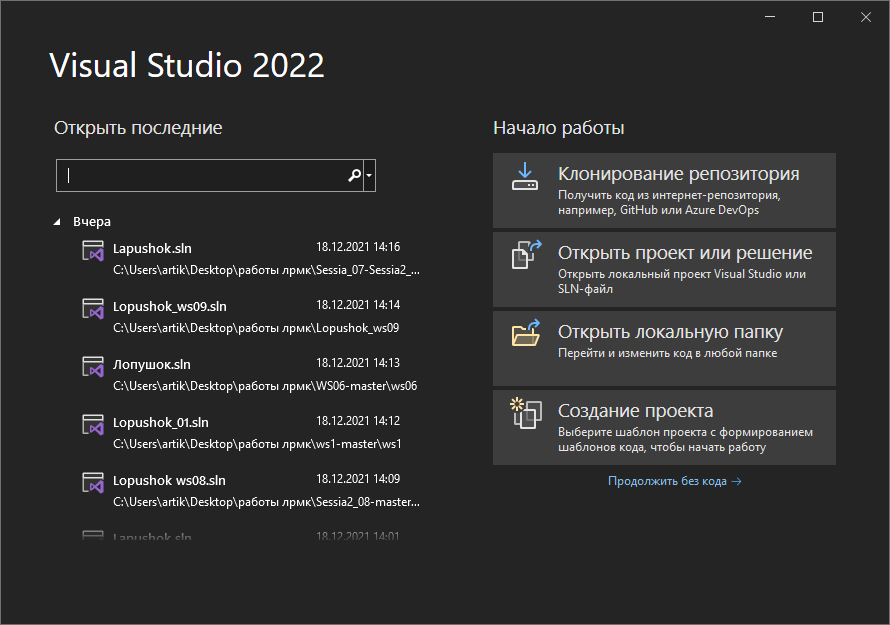


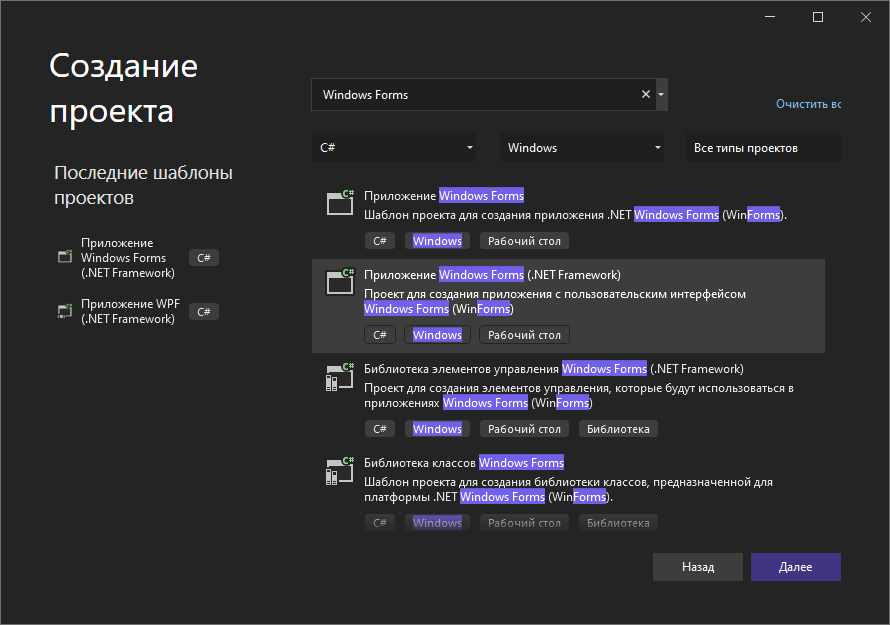
Рисунок 1. Макет приложения

Смотря на этот макет, можно понять, что это часть работы R в аббревиатуре CRUD, то есть R (read). Скорей всего это будет цикл, в котором будет выводить значения из таблицы продукты, а также связанные с ними материалы. То есть там предполагается двойной цикл. А чтобы наше приложение работало быстро предлагается сделать пагинацию (pagination), то есть постраничный вывод. Это, кстати, четко прописано в критериях.

Нам надо сделать: просмотре списка продукции, добавление удаление редактирование данных о продукции, управление списком материалов, необходимых для производства продукции.



Создаем проект Windows Forms



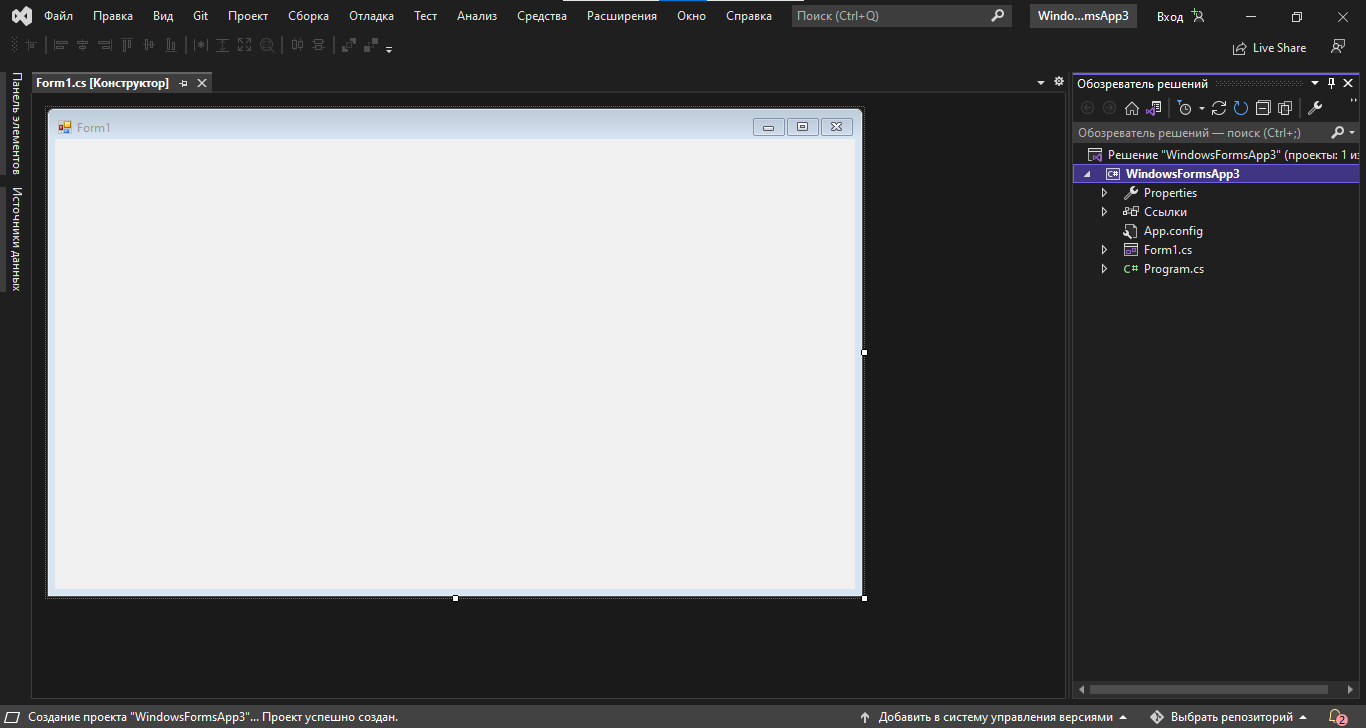


Рисунок 1 – Начальный экран приложения

Далее давайте определимся с горячими сочетаниями клавиш, которые очень часто придется использовать. Это панель инструментов Ctrl + Alt + X и обозреватель решений Ctrl + Alt + L.

Первым делом надо проверить подключение к MySQL серверу и базе данных. Для этого в конструкторе стандартной формы мы пропишем следующие настройки, но перед этим надо убедиться, что мы может работать с MySQL. На рисунке 1 показано добавление MySQL.Data в проект (управление пакетами NuGet).

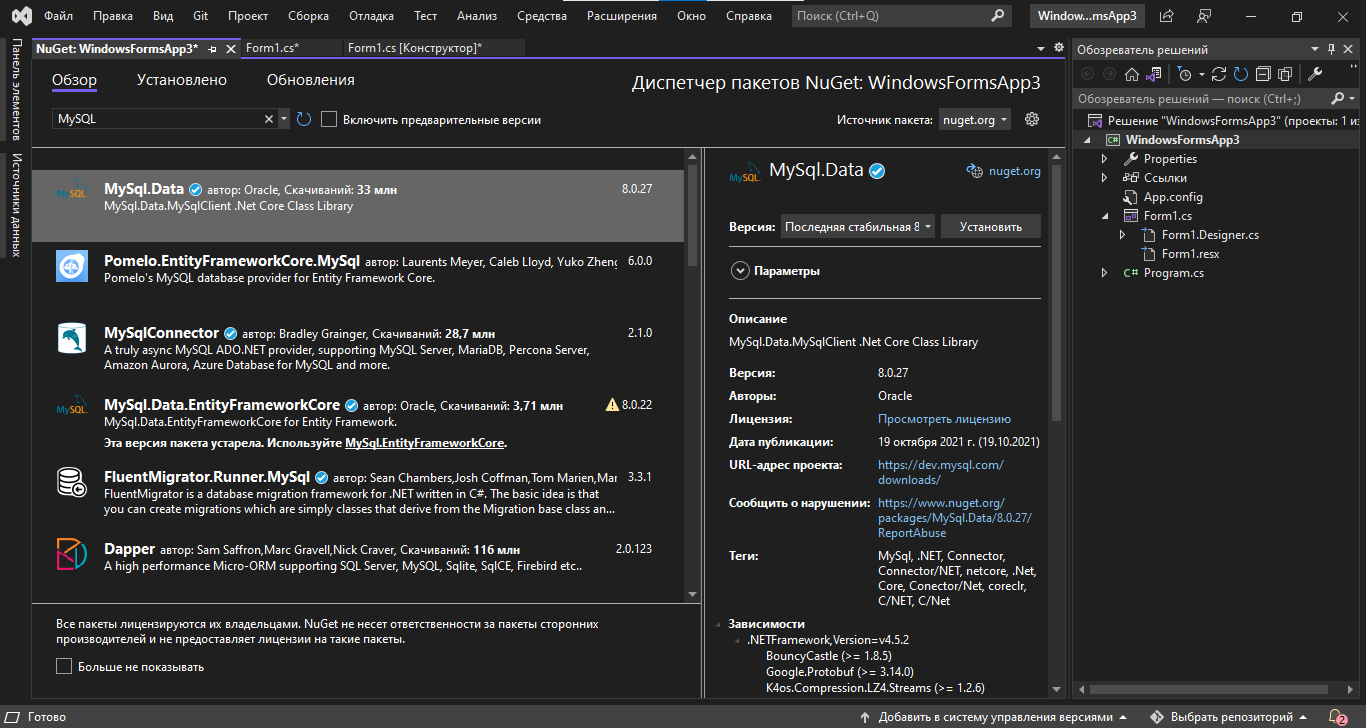


Рисунок 1. Управление пакетами NuGet. Mysql Data

Теперь подключаем

using MySql.Data;

using MySql.Data.MySqlClient;

Код, который может проверить подключение к базе

try

{

// create connection

string mysql = @"server=localhost;userid=root;password=root;database=test\_data";

MySqlConnection connect = new MySqlConnection(mysql);

connect.Open();

MessageBox.Show("Подключение прошло успешно!");

connect.Close();

}

catch

{

MessageBox.Show("Не удалось подключиться!");

}

finally

{

MessageBox.Show("Сеанс работы завершен");

}

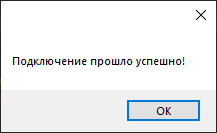


Рисунок 1 – Результат подключения к серверу MySQL

Теперь подумаем, как лучше организовать свою работу. Начать разработку с интерфейсной части или логики. Я начну с создания каркаса приложения, создам его структуру, внешнюю составляющую, потом буду делать так, чтобы все заработало.