Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра Информатики

Дисциплина: Объектно-ориентированное программирование

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему

Программное средство «Электронный атлас»

БГУИР КП I – 40 04 01

Выполнил

студент гр. 053504 Гурский А.Д.

Проверила: Луговская А.М.

Минск 2022

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой Информатика

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Волорова Н. А. 2022г.

ЗАДАНИЕ

по курсовому проектированию

Студенту *Гурскому Артемию Дмитриевичу* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема работы *Программное средство «Электронный атлас» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­­­­­­­\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок сдачи законченной работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Исходные данные к работе *Среда программирования C#. Использование баз данных MS SQL для хранения сущностей четырёх предметных областей. Графическое отображение в WinForms данных сущностей четырёх предметных областей с использованием наведения мыши на заданные экранные координаты сущности и отображения её данных. Возможность переключаться между четырьмя предметными областями.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые

подлежат разработке)

*Введение*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*1 Анализ аналогов и постановка задачи\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2 Используемые технологии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*3 Проектирование приложения \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*4 Программная реализация \_\_\_\_*

*5 Тестирование приложения \_\_\_\_\_*

*Заключение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Список использованных источников\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

\_*Приложения А - Текст программы* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

*Схема алгоритма в формате А4*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Консультант по курсовой работе Луговская А. М.

7.Дата выдачи задания *16.02.2022г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

*Раздел 1. Введение к 28.02.2022г. – 10 % готовности работы;\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 2 к 15.03.2022г. – 30% готовности работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 3 к 15.04.2022г. – 60% готовности работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 4 к 10.05.2022г. – 80% готовности работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Раздел 5. Заключение. Приложения к 30.05.2022г. - 90% готовности работы;*

*оформление пояснительной записки и графического материала к 07.06.2022г. – 100% готовности работы.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Защита курсового проекта с 30.05.2022г. по 09.06.2022г.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

РУКОВОДИТЕЛЬ *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Луговская А. М.*

*(подпись)*

Задание принял к исполнению *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Гурский А. Д. 16.02.2022г.*

*(дата и подпись студента)*

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc104806559)

[1. АНАЛИЗ АНАЛОГОВ И ПОСТАНОВКА ТРЕБОВАНИЙ К ПРИЛОЖЕНИЮ 5](#_Toc104806560)

[1.1 Описание предметной области 5](#_Toc104806561)

[1.2 Требование к разрабатываемому проекту 5](#_Toc104806562)

[2. ОБЗОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ 6](#_Toc104806563)

[2.1 Используемые технологии 6](#_Toc104806564)

[2.2 Среда реализации задачи 6](#_Toc104806565)

[2.2.1 C# 6](#_Toc104806566)

[2.2.2 Entity Framework 6](#_Toc104806567)

[2.2.3 Windows Forms 7](#_Toc104806568)

[2.2.3 MS SQL 7](#_Toc104806569)

[3. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ 7](#_Toc104806570)

[3.1 АРХИТЕКТУРНЫЕ РЕШЕНИЯ 7](#_Toc104806571)

[3.1.1 МНОГОУРОВНЕВАЯ АРХИТЕКТУРА 7](#_Toc104806572)

[3.1.2 ДОМЕННАЯ МОДЕЛЬ 9](#_Toc104806573)

[3.1.3 ДИАГРАММА БАЗ ДАННЫХ 10](#_Toc104806574)

[3.1.4 MVC и MVVM 11](#_Toc104806575)

[4. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ 12](#_Toc104806576)

[4.1 Реализация MVC и MVVM 12](#_Toc104806577)

[4.2 Логика рисования 17](#_Toc104806578)

[5. ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ 17](#_Toc104806579)

[ЗАКЛЮЧЕНИЕ 17](#_Toc104806580)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 18](#_Toc104806581)

[ПРИЛОЖЕНИЕ А – ТЕКСТ ПРОГРАММЫ 18](#_Toc104806582)

# ВВЕДЕНИЕ

С древних времён люди рисовали карты-путеводители, звёздные карты, морские карты. С прошедшим временем в 20 веке с покорением космоса для человека планета стала полностью доступна для её исследования, а космос стремительно начали открывать телескопы и спутники.

Всё новые и новые карты составляли с каждым открытием и огромное количество карт по какой-либо теме, по какому-либо региону начали соединять в атласы для удобства полного просмотра всей картографии

В эпоху развития компьютерных технологий знания человечества постепенно начали переносить на цифровые устройства, появились приложения для их представления и просмотра. А с появлением интернета всё начало мигрировать в него. Однако данный проект будет из себя представлять desktop-приложение.

# 1. АНАЛИЗ АНАЛОГОВ И ПОСТАНОВКА ТРЕБОВАНИЙ К ПРИЛОЖЕНИЮ

## 1.1 Описание предметной области

**Электронный атлас** – это целостное электронное картографическое произведение в форме собрания электронных карт, выполненных по единой программе.

Классификация электронных карт строится на традиционных принципах классификации атласов географических по их содержанию, назначению и территориальному охвату, а также в соответствии с их функциональными возможностями: формами визуализации, возможностью оперирования атрибутивной частью данных, ввода новых пространственных объектов, встраивания некартографических элементов содержания электронного атласа (поддержки многосредности), генерации нефиксированного (произвольного) набора карт по множеству исходных данных с использованием развитых средств их графического оформления, привлечения операций пространственного анализа.

Помимо картографического изображения, обычно включают обширные текстовые комментарии, табличные данные. Как правило, создаются для справочно-информационных и общеобразовательных целей средствами географических информационных систем (ГИС) или автоматических картографических систем.

В данном проекте используются атласы из нескольких предметных областей: астрономия (звёздное небо северного полушария), биология (ареал обитания конкретных видов животных, география (политическая карта и карта географических объектов). К примеру для создания карты звёздного неба рассматривался такой сайт, как http://meteoweb.ru/astro/skaymaps1.php.

## 1.2 Требование к разрабатываемому проекту

Готовый проект должен удовлетворять следующим требованиям:

⎯ Надёжность разработанного проекта

⎯ Максимально точное отображение и соответствие информации

Задача данного курсового проекта сводится к реализации оконного приложения с подключённой базой данных и работой с ними с помощью средств языка C# и среды Visual Studio 2022.

# 2. ОБЗОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ

## 2.1 Используемые технологии

Для проектной реализации был использован .NET Framework и язык C#. Работа выполнялась с использованием библиотек .NET 5, Entity Framework, Windows Form. Также использовалась база данных MS SQL.

## 2.2 Среда реализации задачи

Проектная реализация выполнена в среде разработки Visual Studio 2022 с использованием языка C# и его библиотек.

### 2.2.1 C#

**C#** — это объектно-ориентированный язык программирования. Разработан в 1998-2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework и .NET Core. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с С-подобным синтаксисом, из них его синтаксис наиболее близок к С++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов, (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыкания, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков С++, Delphi, Smalltalk, Modula и, в особенности, Java — C#, опираясь на практику их использования исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличии от С++ не поддерживает множественное наследование классов (между тем допускается множественная реализация интерфейсов).

### 2.2.2 Entity Framework

**ADO.NET Entity Framework** (EF) — объектно-ориентированная технология доступа к данным, является [object-relational mapping](https://ru.wikipedia.org/wiki/ORM) (ORM) решением для [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework) от [Microsoft](https://ru.wikipedia.org/wiki/Microsoft). Предоставляет возможность взаимодействия с объектами как посредством [LINQ](https://ru.wikipedia.org/wiki/LINQ) в виде LINQ to Entities, так и с использованием Entity SQL..

### 2.2.3 Windows Forms

**Windows Forms** — [интерфейс программирования приложений](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9) (API), отвечающий за [графический интерфейс пользователя](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F) и являющийся частью [Microsoft](https://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft) [.NET Framework](https://ru.wikipedia.org/wiki/.NET_Framework). Данный интерфейс упрощает доступ к элементам интерфейса [Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows) за счет создания обёртки для существующего [Win32 API](https://ru.wikipedia.org/wiki/Windows_API) в [управляемом коде](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4). Причём управляемый код — классы, реализующие [API](https://ru.wikipedia.org/wiki/API) для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, С++, так и на VB.Net, J# и др.

С одной стороны, Windows Forms рассматривался как замена более старой и сложной библиотеке [MFC](https://ru.wikipedia.org/wiki/Microsoft_Foundation_Classes), изначально написанной на языке [C++](https://ru.wikipedia.org/wiki/C%2B%2B). С другой стороны, WF не предлагает парадигмы, сравнимой с [MVC](https://ru.wikipedia.org/wiki/Model-View-Controller). Для исправления этой ситуации и реализации данной функциональности в WF существуют сторонние библиотеки. Одной из наиболее используемых подобных библиотек является [User Interface Process Application Block](http://www.microsoft.com/downloads/details.aspx?familyid=98C6CC9D-88E1-4490-8BD6-78092A0F084E&displaylang=en), выпущенная специальной группой Microsoft, занимающейся примерами и рекомендациями, для бесплатного скачивания. Эта библиотека также содержит исходный код и обучающие примеры для ускорения обучения.

### 2.2.3 MS SQL

**Microsoft SQL Server** — [система управления реляционными базами данных (РСУБД)](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94), разработанная корпорацией [Microsoft](https://ru.wikipedia.org/wiki/Microsoft" \o "Microsoft). Основной используемый язык запросов — [Transact-SQL](https://ru.wikipedia.org/wiki/Transact-SQL" \o "Transact-SQL), создан совместно Microsoft и [Sybase](https://ru.wikipedia.org/wiki/Sybase" \o "Sybase). Transact-SQL является реализацией стандарта [ANSI](https://ru.wikipedia.org/wiki/%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B0%D0%BD%D1%81%D0%BA%D0%B8%D0%B9_%D0%BD%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%81%D1%82%D0%B8%D1%82%D1%83%D1%82_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BE%D0%B2)/[ISO](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D0%B4%D1%83%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D0%B0%D1%8F_%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%BF%D0%BE_%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8) по структурированному языку запросов ([SQL](https://ru.wikipedia.org/wiki/SQL)) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

# 3. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

## 3.1 АРХИТЕКТУРНЫЕ РЕШЕНИЯ

Для создания проекта были взяты следующие архитектурные паттерны: многоуровневая архитектура (N-layer architecture), смесь паттернов MVC и MVVM, доменная модель (из концепции Domain-Driven Design), а также представление отношений баз данных.

### 3.1.1 МНОГОУРОВНЕВАЯ АРХИТЕКТУРА

В данном курсовой работе отношение между уровнями таково: пользователь обращается к уровню презентации, который вкулючает в себя:

* форму приложения;
* View (из шаблона MVC);
* ViewModel из шаблона MVVM.

При обращении пользователя к уровню презентации (Presentation Layer), выполняет вызов требуемой бизнес логики из слоя бизнес логики (Business Logic Layer) посредством контроллера (Controller модели MVC).

Затем Controller для обработки запроса обращается к уровню доменной модели, где и будут обработаны данные, которые вызывает запрос пользователя, в DomainModel.

Для доступа к данным, бизнес-логика обращается к доменной модели (DomainModel), которпя, в свою очередь, обращается к уровню доступа к данным (Data Access Layer – DAL). Слой доступа к данным использует модель сущностей базы данных MS SQL, сгенерированную с помощью библиотеки Entity Framework - класс Model1, который выполняет чтение данных из уровня данных (Data Layer), где данные хранятся в реляционной базе данных MS SQL.

Суть использования данного паттерна в том, чтобы логические слои системы можно было разрабатывать и поддерживать независимым образом. Недостаток заключается в увеличении уровней: чем больше уровней в проекте, тем дольше идёт запрос и уменьшается производительность проекта, однако в нашем случае, уровни помогают нам в удобном проектировании двух следующих паттернов и для данного проекта высокая производительность не требуется.

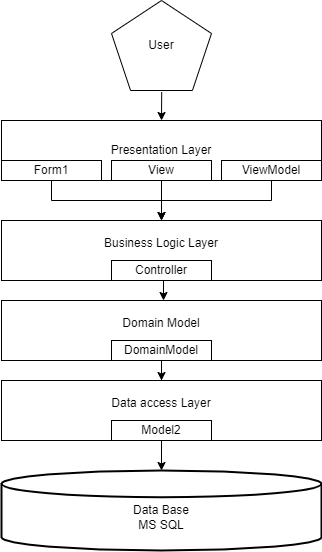
****

Рисунок 1. Логическая диаграмма многоуровневой архитектуры

### 3.1.2 ДОМЕННАЯ МОДЕЛЬ

Доменная модель курсовой работы реализована совокупностью классов и интерфейсов.

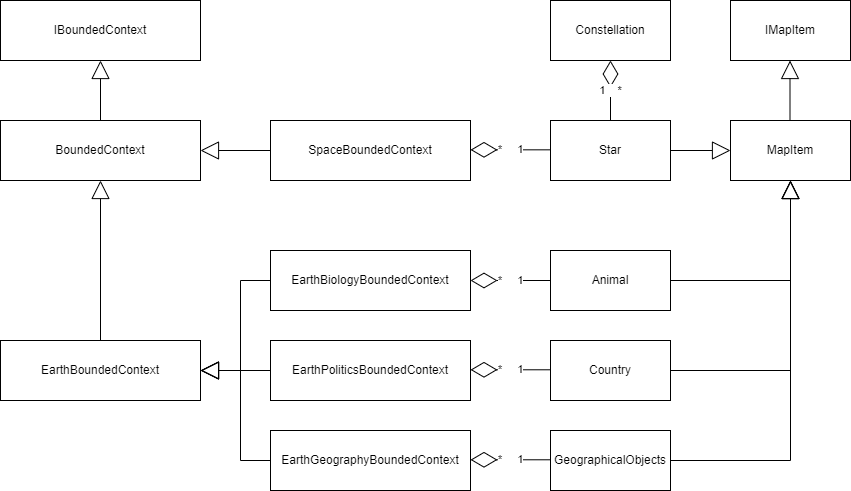
****

Рисунок 2. Диаграмма отношения классов

Доменная модель разбита на ограниченные контексты (bounded context) для группировки сущностей из разных областей. В проекте используется 4 ограниченных контекста:

* SpaceBoundedContext – для группировки сущностей, моделирующих предметную область Космос;
* EarthAnimalBoundedContext – для группировки сущностей, моделирующих предметную область Животный Мир;
* EarthPoliticsBoundedContext – для группировки сущностей, моделирующих предметную область Страны;
* EarthGeographyBoundedContext – для группировки сущностей, моделирующих предметную область География;

Все классы сущностей предметных областей наследуются от класса MapItem для реализации общей функциональности отрисовки этих сущностей на экране (определение координат, вывод подсказок и т.д).

В проекте находится 25 классов.

### 3.1.3 ДИАГРАММА БАЗ ДАННЫХ

Данная диаграмма представляет отношения между таблицами данных Stars, ConstellationStars и Constellations, и отображает поля таблиц. Stars и Constellations связаны между собой таблицей ConstellationStars для рисования звёзд соединённых линиями, образуя их в созвездия.

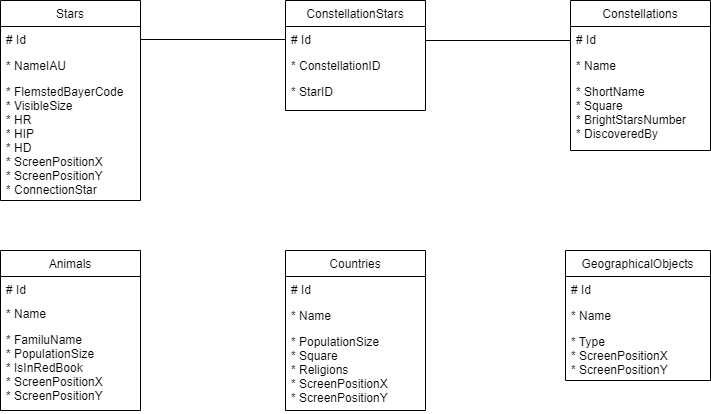
****

Рисунок 3. Диаграмма баз данных

### 3.1.4 MVC и MVVM

Данная диаграмма последовательности представляет совместную работу двух паттернов: MVC и MVVM.

MVC — это архитектурный шаблон, обычно используемый для разработки пользовательских интерфейсов, который делит приложение на три взаимосвязанные части, называемые моделью, видом и контроллером.

MVVM — это программный архитектурный паттерн, который позволяет отделить разработку графических пользовательских интерфейсов от разработки бизнес-логики или серверной логики.

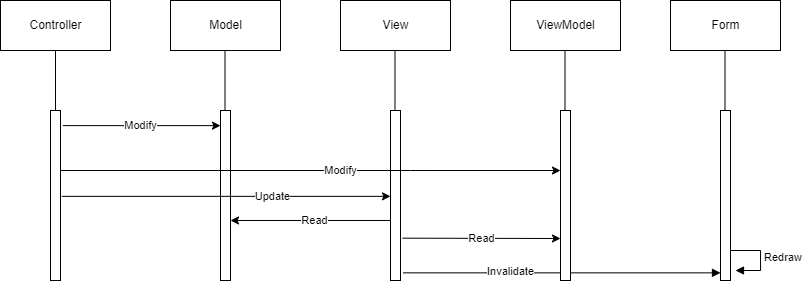


Рисунок 4. Диаграмма MVC и MVVM

# 4. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

## 4.1 Реализация MVC и MVVM

В курсовой работе используется смесь паттернов MVC и MVVM.

За отрисовку формы отвечает класс View, который агрегирует форму и управляет ей.

Состояние View моделируется классом ViewModel. Этот класс хранит переменные, описывающие параметры визуализации данных для пользователя, выбранные настройки визуализации и т.д.

За выполнение бизнес-логики отвечает Controller, который содержит код с бизнес-логикой приложения.

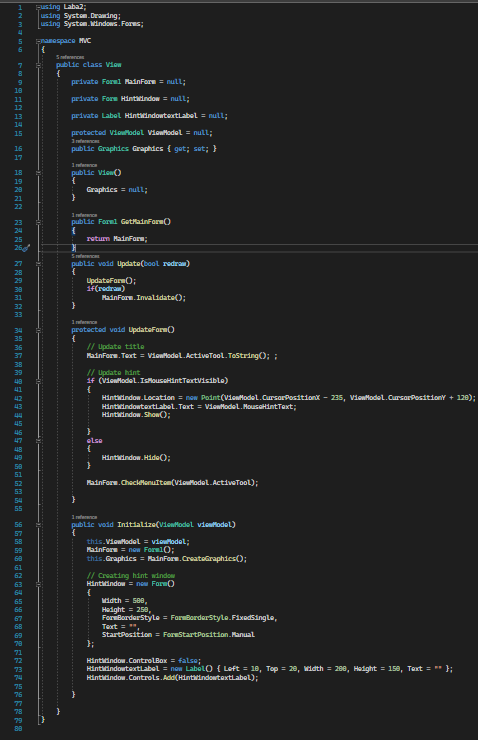


Рисунок 5. Программная реализация View

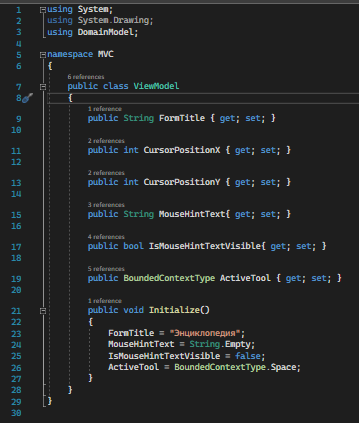


Рисунок 6. Программная реализация ViewModel

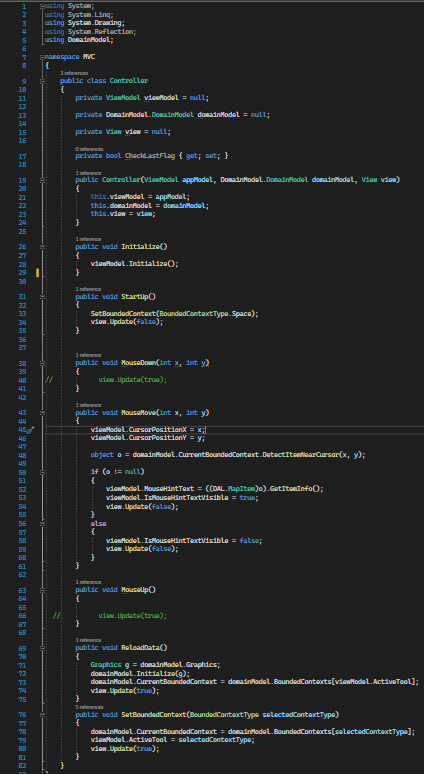


Рисунок 7. Программная реализация Controller

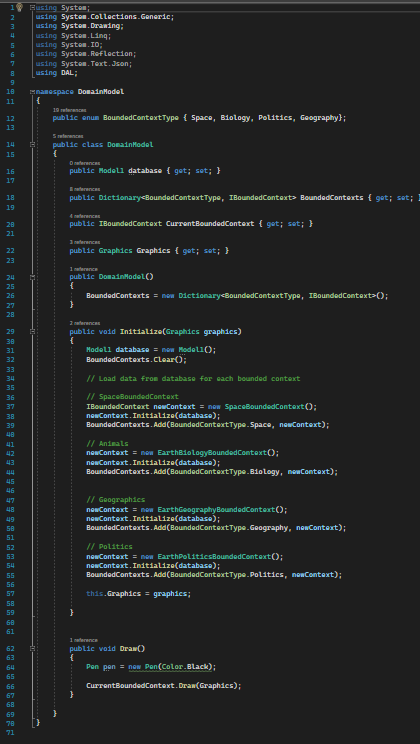


Рисунок 8. Программная реализация DomainModel

## 4.2 Логика рисования

Для рисования сущностей выбранного пользователем ограниченного контекста, испольщуется метод Draw(), который рисует точки сущностей контекста по их экранным координатам.

Это становится возможным, поскольку для каждой сущности предменой области хранятся экранные координаты (ScreenPositionX и ScreenPositionY).

Для вывода справочной информации по выбранному пользователем объекту, используется окно, реализуемое HintWindow. А для заполнения данными этого окна, Controller обращается к DomainModel, что обращается к данным контекста, из которого нам надо взять данные.

При обнаружении сущности рядом с курсором мыши, Controller поместит информацию об этой сущности во ViewModel, после чего , затем вызывается View, который отрисует форму по заданным экранным координатам и переданному тексту для окна из ViewModel.

Триггером для рисования окна являются успешная проверка наличия элементов данных на экране в радиусе 15 точек вокруг курсора мыши. Такая проверка выполняется для каждого события MouseMove.

# 5. ТЕСТИРОВАНИЕ ПРИЛОЖЕНИЯ

Тестирование проводилось вручную с помощью отладки. В следующей версии проекта планируется проводить тестирование с помощью unit-тесты.

# ЗАКЛЮЧЕНИЕ

В рамках работы над курсовым проектом был разработан проект “Электронный Атлас”, представленный в виде desktop-приложения на базе Windows Form.

В процессе написания были использованы: C# и его библиотеки Windows Form и Entity Framework и база данных MS SQL. В проекте были применены архитектурный многоуровневый паттерн, что представляет основу проекта и паттерны MVC, и MVVM, встроенные в эту основу и взаимодействующие между собой.

В следующей версии проекта планируется перенести структуры приложения в сеть, проводить отладку через unit-тесты, а также расширить базу данных и улучшить интерфейс пользователя.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Сайт Министерство Культуры Российской Федерации  [Электронный ресурс]/Большая Российская Энциклопедия – Режим доступа: <https://bigenc.ru/geography/text/4929743>

2. C# [Электронный ресурс]/ статья в Википедии – Режим доступа: https://ru.wikipedia.org/wiki/C\_Sharp

3. .Net [Электронный ресурс]/статья в Википедии – Режим доступа: <https://en.wikipedia.org/wiki/.NET>

4. Entity Framework [Электронный ресурс]/статья в Википедии – Режим доступа: <https://ru.wikipedia.org/wiki/ADO.NET_Entity_Framework>

5. Windows Forms [Электронный ресурс]/статья в Википедии – Режим доступа: <https://ru.wikipedia.org/wiki/Windows_Forms>

6. Microsoft SQL [Электронный ресурс]/статья в Википедии – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_SQL_Server>

7. Самые важные архитектурные паттерны, которые нужно знать [Электронный ресурс]/ Хабр – Режим доступа: <https://habr.com/ru/company/alconost/blog/522662/>

8. В чём разница между MVC и MVVM [Электронный ресурс]/ Strephosays – Режим доступа: <https://ru.strephonsays.com/what-is-the-difference-between-mvc-and-mvvm#:~:text=MVC%20-%20это%20архитектурный%20шаблон%2C,разработки%20бизнес-логики%20или%20серверной%20логики>

# ПРИЛОЖЕНИЕ А – ТЕКСТ ПРОГРАММЫ

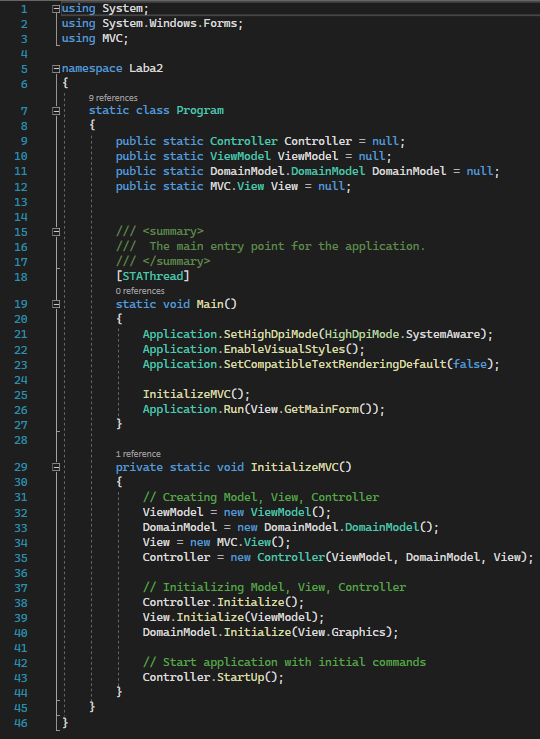


Рисунок 9. Программная реализация Program

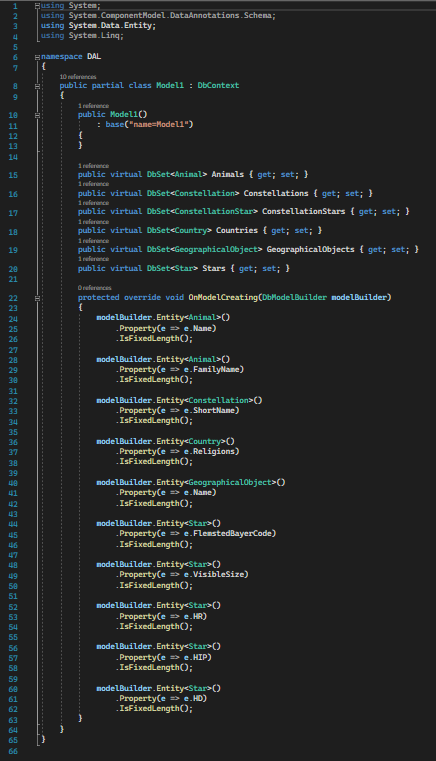


Рисунок 10. Программная реализация Model1

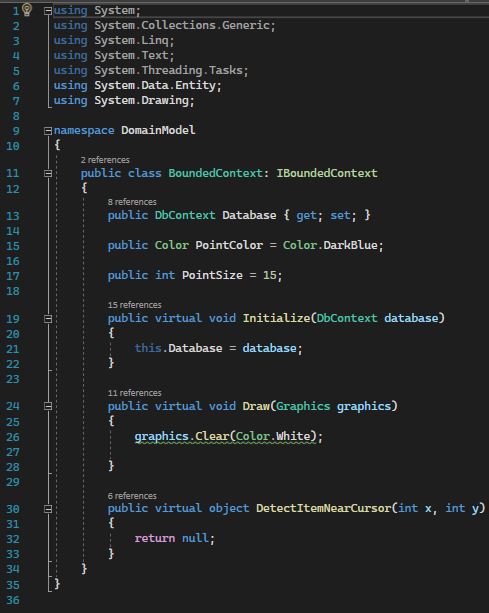


Рисунок 11. Программная реализация BoundedContext

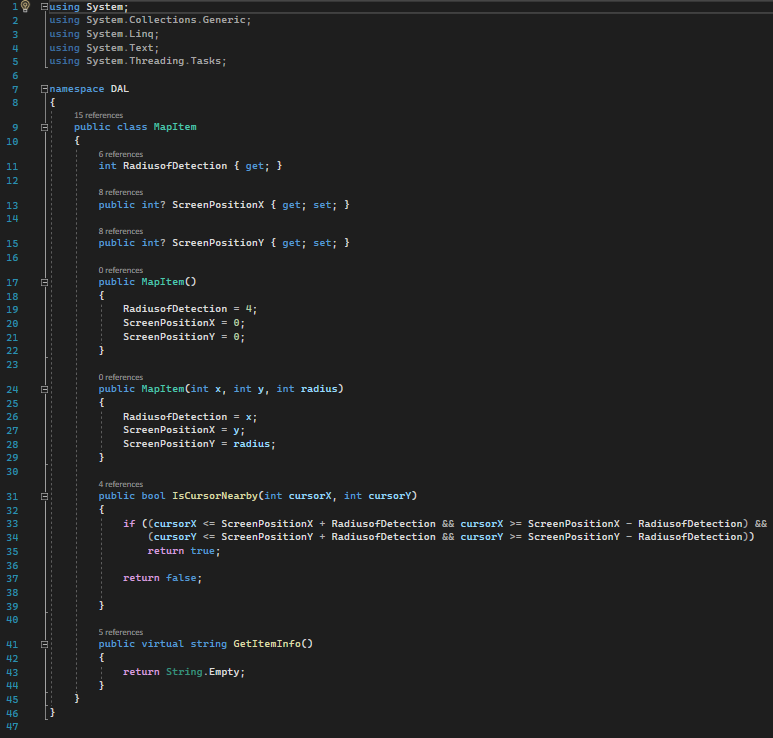


Рисунок 12. Программная реализация MapItem