**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Научный руководитель образовательной программы «Программная инженерия» доцент департамента программной инженерии факультета компьютерных наук    \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.В. Вознесенская  «15» \_\_мая\_\_\_ 2022 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия», кандидат технических наук  Автограф В В.В. Шилов  «16» мая 2022 г. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Интерактивная визуализация данных приемной кампании**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.10.03-01 81 01-1-ЛУ** | | |
|  |  | |
| Изображение выглядит как текст  Автоматически созданное описаниеИсполнитель:  студент группы БПИ203  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Я.Ю. Щербаков /  «15» \_\_\_\_\_\_\_\_\_мая\_\_\_\_\_\_\_\_\_\_ 2022 г. | |
|  | | |
| **Москва 2022** | |  |

|  |  |  |  |
| --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.10.03-01 81 01-1-ЛУ |  |  | |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл*** |  | | **Интерактивная визуализация данных приемной кампании**  **Пояснительная записка**  **RU.17701729.10.03-01 81 01-1**  **Листов 33** | | | | |
|  |  | | | |
|  | | | |
|  | | | | |
| **Москва 2022** | | |  | |

СОДЕРЖАНИЕ

[1. ВВЕДЕНИЕ 3](#_Toc103624489)

[1.1. Наименование разработки 3](#_Toc103624490)

[1.2. Основание для разработки 3](#_Toc103624491)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ РАЗРАБОТКИ 4](#_Toc103624492)

[2.1. Функциональное назначение 4](#_Toc103624493)

[2.2. Эксплуатационное назначение 4](#_Toc103624494)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc103624495)

[3.1. Постановка задачи на разработку программы 5](#_Toc103624496)

[3.2. Описание алгоритма и функционирования программы 5](#_Toc103624497)

[3.3. Конфигурация базы данных 9](#_Toc103624498)

[3.4. Описание и обоснование выбора метода организации входных и выходных данных…………………………………………………………………………………………………10](#_Toc103624499)

[3.4.1. Описание метода организации входных и выходных данных 10](#_Toc103624500)

[3.4.2. Обоснование выбора метода организации входных и выходных данных 10](#_Toc103624501)

[3.5. Описание и обоснование выбора и состава технических и программных средств…………………………………………………………………………………………………10](#_Toc103624502)

[3.5.1. Описание технических и программных средств 10](#_Toc103624503)

[3.5.2. Обоснование выбора технических и программных средств 10](#_Toc103624504)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 11](#_Toc103624505)

[4.1. Ориентировочная экономическая эффективность 11](#_Toc103624506)

[4.2. Предполагаемая потребность 11](#_Toc103624507)

[4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 11](#_Toc103624508)

[5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 12](#_Toc103624509)

[ПРИЛОЖЕНИЕ 1. 13](#_Toc103624510)

[ПРИЛОЖЕНИЕ 2. 14](#_Toc103624511)

[ПРИЛОЖЕНИЕ 3. 15](#_Toc103624512)

[ПРИЛОЖЕНИЕ 4. 18](#_Toc103624513)

[ПРИЛОЖЕНИЕ 5. 19](#_Toc103624514)

[ПРИЛОЖЕНИЕ 6. 24](#_Toc103624515)

[ПРИЛОЖЕНИЕ 7. 25](#_Toc103624516)

[ПРИЛОЖЕНИЕ 8. 28](#_Toc103624517)

[ПРИЛОЖЕНИЕ 9. 29](#_Toc103624518)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 33](#_Toc103624519)

1. **ВВЕДЕНИЕ**

## Наименование разработки

**Наименование программы на русском языке:** «Интерактивная визуализация данных приёмной кампании».

**Наименование программы на английском языке: «**Interactive visualization of admission campaign data».

## Основание для разработки

Данная разработка выполнена в рамках написания курсовой работы. Основанием для её выполнения является учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и тема курсового проекта, которая утверждена академическим руководителем программы.

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ РАЗРАБОТКИ**
   1. **Функциональное назначение**

Приложение предназначено для облегчения процесса обработки данных приемной компанией НИУ ВШЭ путем предоставления централизованного места хранения и визуализации этих данных при помощи интерактивной карты мира

* 1. **Эксплуатационное назначение**

Программа была задумана преимущественно для помощи сотрудникам приемной компании НИУ ВШЭ, обладающих доступом к персональному компьютеру (в рабочих целях).

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

В рамках курсовой работы требуется реализовать программу для работы с данными приемной комиссией НИУ ВШЭ. Отдельно моим напарником по курсовому проекту выполняется часть визуализации, которая отправляет запросы данному приложению и получает соответствующие ответы. Разрабатываемая программа должна обеспечивать возможность выполнения следующих функций:

* хранение и аккумулирование информации об абитуриентах;
* возможность изменения информации об абитуриентах;
* возможность удаления абитуриента;
* возможность добавления информации об абитуриенте в формате CSV и при помощи запросов в формате JSON;
* возможность фильтрации абитуриентов;
* возможность вывода статистики по регионам.
  1. **Описание алгоритма и функционирования программы**

Для разработки приложения была изучена и использована технология opensource-фреймворка компании Microsoft ASP.NET Core, с помощью которой было создано веб-приложение. Основное преимущество приложения на ASP.NET Core заключается в его кроссплатформенности. Это означает, что веб-приложение можно запускать не только на ОС Windows, а также на Mac OS и Linux.

Применяемая в данном проекте технология построения приложения Web API ASP.NET направлена на выполнение работы в стиле REST. Аббревиатура переводится, как «передача состояния представления». Применение архитектуры данного стиля предполагает использование таких методов/типов запросов HTTP, как PUT, POST, DELETE, GET.

Как правило, стиль REST подходит для создания таких приложений, в которых задействованы javascript-фреймворки (React, Angular или Vue.js), например для Single Page Application. То есть WEB API является веб-сервисом, к которому обращаются сторонние приложения, представляющие любую технологию или платформу (мобильные, десктопные приложения).

На сегодняшний день для разработки приложений используется огромное количество разных типов архитектур, и одной из наиболее популярных является именно трехуровневая система. Соответственно, она предполагает разделение программы на три уровня. В данном курсовом проекте мы использовали трехуровневую архитектуру приложения ASP.NET MVC.

Стоит уточнить, что n-layer и n-tier – не совсем связанные между собой понятия, тем не менее, оба применяются в качестве обозначения многоуровневой системы. Laver и Tier означают «уровень», однако это уровни разного порядка.

Tier является физическим уровнем. Таким образом, в применяемой нами трехуровневой архитектуре n-tier приложение может содержать такие уровни, как браузер пользователя, веб-приложение и сервер базы данных, где каждый уровень – обособленный физический процесс, даже если все они содержатся на одном компьютере. В том случае, если в качестве альтернативы клиентом бы выступало мобильное приложение, это был бы уже новый физический уровень.

Layer – логический уровень. В нашем распоряжении могут быть уровень сервисов, уровень представления, уровень бизнес-логики, уровень доступа к данным и т. д. Важно отметить, что физические и логические уровни никогда не совпадают.

Таким образом, уровень предоставления в приложении ASP.NET зачастую включает в себя представления (отображаются в веб-браузере) и контроллеры (отвечают за обработку ввода) – то есть подразделяется на два физических уровня.

В этом курсовом проекте применяются логические уровни – n-layer архитектура.

В ходе курсового проекта мы создаем веб-приложение, разделенное на три уровня:

• Уровень представления (AdmissionСampaign.API).

На данном уровне происходит взаимодействие с пользователем. Данное Web-приложение реализовано с помощью технологии WEB API. Реализован контроллер для направления и обработки запросов. Предоставлены следующие конечные точки для запросов от пользователя:

Таблица 1. Конечные точки запросов от пользователя.

|  |  |
| --- | --- |
| Конечная точка | Назначение |
| {host}/Entrant/AddEntrant | Добавление абитуриента в систему |
| {host}/Entrant/UploadDocument | Добавление списка абитуриентов с помощью файла в формате CSV. |
| {host}/Entrant/UpdateEntrant | Обновление абитуриента в системе |
| {host}/Entrant/RemoveEntrant/{id} | Удаление абитуриента в системе |
| {host}/Entrant/RemoveAllEntrants | Удаление всех абитуриентов из системы |
| {host}/Entrant/GetByEntrantId/{id} | Получение полной информации об абитуриенте |
| {host}/Entrant/GetAllEntrants | Постраничное получение абитуриентов |
| {host}/Entrant/FilterEntrants | Получение списка отфильтрованных абитуриентов |
| {host}/Entrant/GetEntrantsByEntryYear | Постраничное получение абитуриентов по году подачи документов |
| {host}/Entrant/GetStaticticByRegions | Получение статистики по регионам |

Документация для приложения была создана с помощью Swagger (см. рис. 1). Она позволяет не только просмотреть спецификацию, но и отправлять запросы (см. рис. 2, 3, 4).

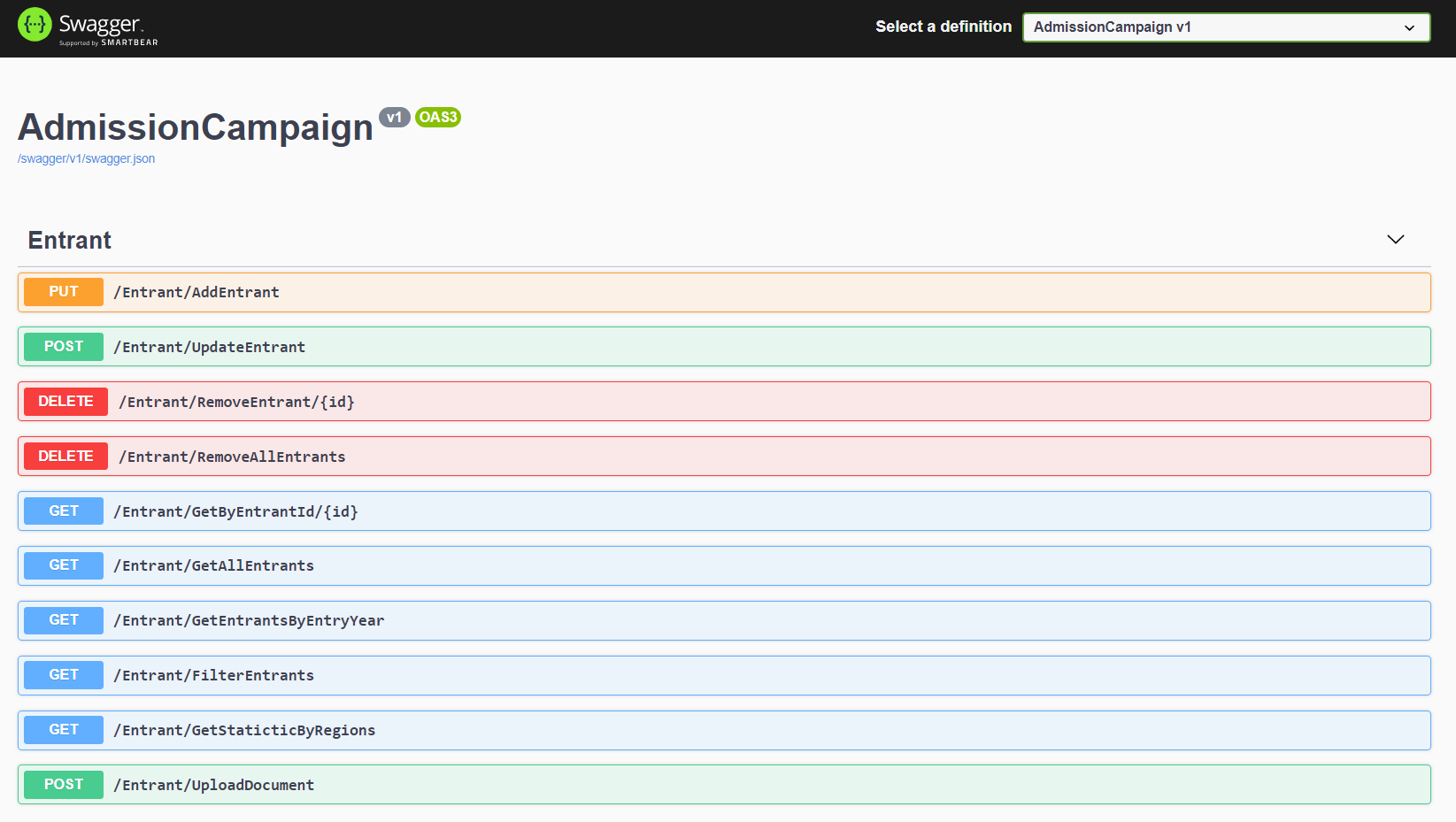


Рисунок 1. Swagger.

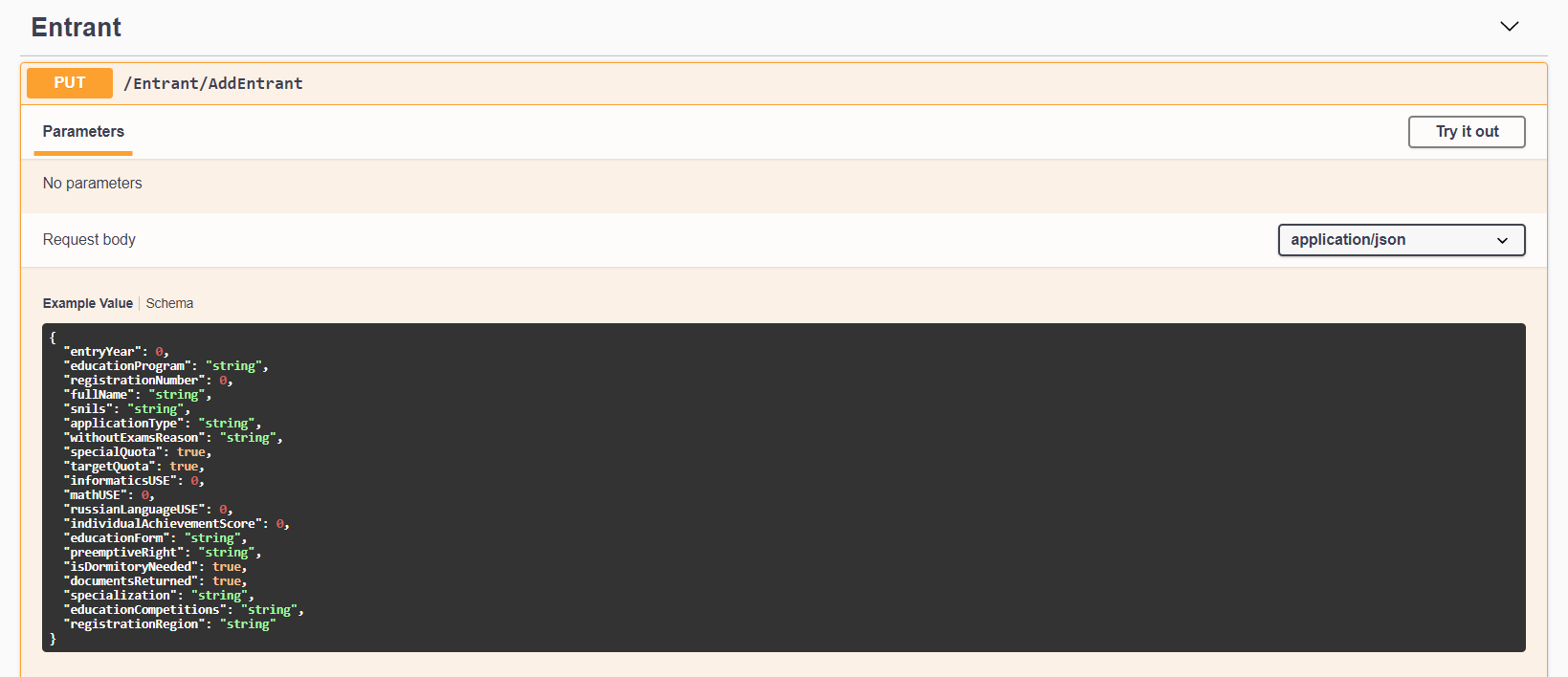


Рисунок 2. Пример запроса.

Изображение выглядит как стол

Автоматически созданное описание

Рисунок 3. Пример модели запроса.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4. Пример ответа на запрос.

• Уровень бизнес-логики (AdmissionСampaign.Services).

На уровне бизнес-логики находятся компоненты, не только реализующие логику приложения, необходимые вычисления и взаимодействие с репозиториями, но и обработку данных уровня представления, а также обратную передачу обработанных данных этому уровню. Реализован сервис для обработки данных от котроллера. Также реализована обработка CSV файла для дальнейшего сохранения данных в базу данных.

• Уровень доступа к данным (AdmissionСampaign.DataAccess).

Данный уровень отвечает за хранение моделей, которые содержат описание использованных сущностей. Кроме того, здесь содержатся специфичные классы, позволяющие работать с различными технологиями доступа к данным (в данном приложении используется Entity Framework и MS QSL база данных). Также здесь содержатся репозитории, которые позволяют взаимодействовать уровню бизнес-логики с базой данных, то есть добавлять, удалять, обновлять абитуриентов в базе данных, а также осуществлять выборку с использованием различных фильтров.

Важно отметить, что в рассматриваемом типе архитектуры невозможно прямое взаимодействие крайних уровней. Например, уровень представления не может напрямую связаться с базой данных или с уровнем доступа данных – взаимодействие возможно только через уровень бизнес-логики.

Цепочка зависимостей уровней выглядит следующим образом: уровень представления зависит от уровня бизнес-логики, который, в свою очередь, зависит от уровня доступа к данным, а последний не зависим от любых других уровней. Запрос от пользователя обрабатывается соответственно сверху вниз – контроллер, сервис, репозиторий, и затем обратным путем возвращается ответ. Схема взаимодействия компонентов изображена на рисунке 5:

Изображение выглядит как текст, стол

Автоматически созданное описание

Рисунок 5. Схема взаимодействия компонентов.

Все приложение представляет решение в Visual Studio, где отдельные уровни – это проекты. Однако, уровень не обязательно соответствует какому-то конкретному отдельному проекту – мы можем разделить один уровень на несколько проектов, но таким образом, чтобы его функции представляли из себя единое логическое звено.

* 1. **Конфигурация базы данных**

Данное Web-приложение манипулирует таблицей Entrants. Она имеет следующий вид:

Таблица 2. Схема Entrants.

|  |  |
| --- | --- |
| Название столбца | Описание |
| Id | bigint primary key not null auto\_increment |
| EntryYear | int not null |
| EducationProgram | nvarchar(max) null |
| RegistrationNumber | int not null |
| FullName | nvarchar(max) null |
| SNILS | nvarchar(max) null |
| ApplicationType | nvarchar(max) null |
| WithoutExamsReason | nvarchar(max) null |
| SpecialQuota | bit not null |
| TargetQuota | bit not null |
| InformaticsUSE | int not null |
| MathUSE | int not null |
| RussianLanguageUSE | int not null |
| IndividualAchievementScore | int not null |
| EducationForm | nvarchar(max) null |
| PreemptiveRight | nvarchar(max) null |
| IsDormitoryNeeded | bit not null |
| DocumentsReturned | bit not null |
| Specialization | nvarchar(max) null |
| EducationCompetitions | nvarchar(max) null |
| RegistrationRegion | nvarchar(max) null |

* 1. **Описание и обоснование выбора метода организации входных и выходных данных**

### 3.4.1. Описание метода организации входных и выходных данных

В качестве входных данных взяты таблицы абитуриентов, в виде файлов CSV. Также для входных и выходных данных был выбран формат данных JSON для получения запросов и отправки ответов Web-приложением.

### 3.4.2. Обоснование выбора метода организации входных и выходных данных

CSV-формат входных данных был выбран ввиду простоты структурного анализа. Учет абитуриентов ведется с помощью таблиц в формате EXCEL. Множество современных редакторов таблиц поддерживают конвертацию данных из EXCEL в CSV. Формат сериализации входных и выходных данных JSON был выбран из-за компактности, читаемости и распространенности данного формата.

* 1. **Описание и обоснование выбора и состава технических и программных средств**

### 3.5.1. Описание технических и программных средств

Для возможности использования программы требуется компьютер с доступом в интернет со следующими техническими характеристиками:

* Процессор Intel или AMD с 4 ядрами, 1.7 GHz или мощнее;
* Оперативная память 4 GB RAM или более;
* Графический процессор с памятью не менее 128 Мб;
* Место в памяти ПК: 3 ГБ или более.

Из программных характеристик требуется операционная система Windows 7 (64 бит) или более поздняя версия, но не позднее Windows 10.

### 3.5.2. Обоснование выбора технических и программных средств

Технические и программные средства выбраны соответствуют рекомендуемым параметрам к реализации программы.

1. **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**

## Ориентировочная экономическая эффективность

Данная работа не предполагает расчета экономической эффективности, поскольку она не является коммерческим проектом.

## Предполагаемая потребность

Программа была задумана, как инструмент для более комфортной и эффективной работы с базами данных приемной комиссией НИУ ВШЭ.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Анализ существующих на сегодняшний день решений не выявил аналогов данной программы в открытом доступе.

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
2. ГОСТ 19.102-77 Стадии разработки. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.101-77 Виды программ и программных документов. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103-77 Обозначения программ и программных документов. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104-78 Основные надписи. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105-78 Общие требования к программным документам. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению. Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603-78. Общие правила внесения изменений - ИПК Издательство стандартов.
10. ГОСТ 19.604-78. Правила внесения изменений в программные документы, выполненные печатным способом - ИПК Издательство стандартов.

# ПРИЛОЖЕНИЕ 1.

**ТЕРМИНОЛОГИЯ**

ASP.NET Core - кроссплатформенный фреймворк для создания веб-приложений на платформе .NET.

CSV – текстовый формат данных, в котором поля значений разделены запятыми.

JSON – текстовый формат для обмена данными, основанный на JavaScript.

HTTP – протокол передачи данных клиент-серверного взаимодействия.

Swagger – язык описания интерфейса для описания RESTful API, выраженных с помощью JSON.

# ПРИЛОЖЕНИЕ 2.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ ПРОЕКТА ADMISSIONCAMPAIGN.API**

Таблица 2.1

Описание и функциональное назначение классов проекта AdmissionCampaign.API

|  |  |
| --- | --- |
| **Класс** | **Функциональное назначение** |
| **EntrantController** | Контроллер абитуриента, отвечающий за обработку входящих запросов, выполнение операций и выбор представлений для визуализации пользователю. После получения входящего запроса контроллер посылает обратно результат обработки. |
| **SwaggerFileOperationFilter** | Фильтр для загрузки файлов через Swagger UI. |
| **HttpStatusCodeExceptionMiddleware** | Промежуточный слой для обработки исключений. |
| **Program** | Главный класс, начинающий выполнение приложения. Код этого класса запускает и настраивает веб-хост, в пределах которого разворачивается приложение. |
| **Startup** | Класс, который является входной точкой в приложение ASP.NET Core и содержит логику обработки входящих запросов. В нём производится конфигурация приложения, настройка сервисов, которые приложение будет использовать, установка компонентов для обработки запроса и промежуточных слоев. |

# ПРИЛОЖЕНИЕ 3.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ ПРОЕКТА ADMISSIONCAMPAIGN.API**

Таблица 3.1

Описание и функциональное назначение полей, методов и свойств класса EntrantController

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля:** | | | | |
| **Имя** | **Модификатор доступа** | **Тип** | **Назначение** | |
| \_entrantService | private | EntrantService | Содержит логику обработки абитуриента. | |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| EntrantController | public | EntrantController | EntrantService | Конструктор для инициализации контроллера и внедрения зависимостей. |
| AddEntrant | public | IActionResult | AddEntrantViewModel | Позволяет добавить абитуриента в систему. |
| UpdateEntrant | public | IActionResult | UpdateEntrantViewModel | Позволяет редактировать абитуриента в системе. |
| RemoveEntrant | public | IActionResult | long | Позволяет удалить абитуриента из системы по индивидуальному порядковому номеру. |
| RemoveAllEntrants | public | IActionResult | без параметров | Позволяет удалить всех абитуриентов из системы. |
| GetByEntrantId | public | IActionResult | long | Позволяет получить абитуриента по индивидуальному номеру в системе. |
| GetAllEntrants | public | IActionResult | PaginationViewModel | Позволяет получить всех абитуриентов из системы. |
| GetEntrantsByEntryYear | public | IActionResult | GetEntrantsByEntryYearViewModel | Позволяет получить всех абитуриентов по конкретному году поступления из системы. |
| FilterEntrants | public | IActionResult | FilterEntrantsViewModel | Позволяет отфильтровать абитуриентов. |
| GetStaticticByRegions | public | IActionResult | GetStaticticByRegionsViewModel | Позволяет получить статистику по регионам. |
| UploadDocument | public | IActionResult | int, string, IFormFile | Позволяет загрузить данные об абитуриентах при помощи документа в специальном формате СSV. |

Таблица 3.2

Описание и функциональное назначение методов класса SwaggerFileOperationFilter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор доступа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| Apply | public | void | OpenApiOperation, OperationFilterContext | Определяет метод загрузки файлов через пользовательский интерфейс Swagger с выбором файла на локальном компьютере. |

Таблица 3.3

Описание и функциональное назначение полей и методов класса HttpStatusCodeExceptionMiddleware

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип** | **Назначение** | |
| \_next | private | RequestDelegate | Содержит ссылку на следующий делегат запроса в конвейере обработки запроса. | |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| HttpStatusCodeExceptionMiddleware | public | HttpStatusCodeExceptionMiddleware | RequestDelegate | Конструктор, который позволяет сохранить ссылку делегата запроса в конвейере обработки запроса. |
| InvokeAsync | public | Task | HttpContext | Данный метод обрабатывает поступающие запросы и отлавливает исключения. |
| HandleDefaultException | public | Task | HttpContext, HttpStatusCode, Exception | Обрабатывает поступающие исключения. Отвечает на перехваченное исключение, устанавливая сообщения об ошибках. |

Таблица 3.4

Описание и функциональное назначение методов класса Program

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| Main | public | void | string[] | Запускает метод CreateHostBuilder от переданных параметров. |
| CreateHostBuilder | public | IHostBuilder | string[] | Создаёт и запускает хост, в рамках которого разворачивается веб-приложение, после чего веб-сервер приступает к считыванию всех входящих HTTP-запросов. |

Таблица 3.5

Описание и функциональное назначение методов и свойств класса Startup

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| Startup | public | Startup | IConfiguration | Конструктор, в котором задаётся начальная конфигурация приложения. |
| ConfigureServices | public | void | IServiceCollection | Регистрирует сервисы, которые используются в приложении. |
| Configure | public | void | IApplicationBuilder, IWebHostEnvironment | Определяет, каким образом приложение будет обрабатывать запрос. Получает информацию о среде запуска приложения и позволяет взаимодействовать с ней. |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Configuration | public | IConfiguration | get | Хранит в себе начальные конфигурации приложения. |

# ПРИЛОЖЕНИЕ 4.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ ПРОЕКТА ADMISSIONCAMPAIGN.DATAACCESS**

Таблица 4.1

Описание и функциональное назначение классов проекта AdmissionCampaign.DataAccess

|  |  |
| --- | --- |
| **Класс** | **Функциональное назначение** |
| **Entrant** | Класс, который содержит и описывает сущность абитуриента. |
| **FilteredEntrantsModel** | Класс представляет модель отфильтрованных абитуриентов. |
| **FilterEntrantsModel** | Класс представляет модель для фильтрации абитуриентов. |
| **GetStaticticByRegionsModel** | Класс представляет модель запроса статистики об абитуриентах по регионам. |
| **PaginationModel** | Класс представляет модель механизма постраничного отображения данных. |
| **RegionStaticticModel** | Модель запроса статистики по регионам, описывающая структуру и логику используемых данных. |
| **EntrantRepository** | Класс представляет репозиторий абитуриентов. |
| **ApplicationContext** | Класс определяет контекст данных, используемый для взаимодействия с базой данных. |
| **ApplicationContextFactory** | Класс фабрики для создания контекста базы данных. |
| **StartupExtension** | Класс содержит методы расширения для первоначальной настройки Services проекта. |

# ПРИЛОЖЕНИЕ 5.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ ПРОЕКТА ADMISSIONCAMPAIGN.DATAACCESS**

Таблица 5.1

Описание и функциональное назначение свойств класса Entrant

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Id | public | long | get, set | Индивидуальный порядковый номер абитуриента в базе данных. |
| EntryYear | public | int | get, set | Год поступления абитуриента. |
| EducationProgram | public | string | get, set | Образовательная программа. |
| RegistrationNumber | public | int | get, set | Регистрационный номер абитуриента. |
| FullName | public | string | get, set | Полное имя абитуриента. |
| SNILS | public | string | get, set | СНИЛС абитуриента. |
| ApplicationType | public | string | get, set | Заявление абитуриента о согласии на зачисление. |
| WithoutExamsReason | public | string | get, set | Право поступления без вступительных испытаний. |
| SpecialQuota | public | bool | get, set | Наличие особой квоты для поступления. |
| TargetQuota | public | bool | get, set | Наличие поступления по целевой квоте. |
| InformaticsUSE | public | int | get, set | Балл ЕГЭ абитуриента по информатике. |
| MathUSE | public | int | get, set | Балл ЕГЭ абитуриента по математике. |
| RussianLanguageUSE | public | int | get, set | Балл ЕГЭ абитуриента по русскому языку. |
| IndividualAchievementScore | public | int | get, set | Балл абитуриента за индивидуальные достижения. |
| EducationForm | public | string | get, set | Предполагаемая форма обучения. |
| PreemptiveRight | public | string | get, set | Наличие преимущественного права при поступлении. |
| IsDormitoryNeeded | public | bool | get, set | Необходимость общежития на время обучения. |
| DocumentsReturned | public | bool | get, set | Возврат документов. |
| Specialization | public | string | get, set | Специализация. |
| EducationCompetitions | public | string | get, set | Все конкурсы. |
| RegistrationRegion | public | string | get, set | Регион регистрации абитуриента. |

Таблица 5.2

Описание и функциональное назначение свойств класса FilteredEntrantsModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Entrants | public | List<Entrant> | get, set | Хранит список отфильтрованных абитуриентов. |
| Total | public | int | get, set | Общее количество отфильтрованных абитуриентов. |

Таблица 5.3

Описание и функциональное назначение свойств класса FilterEntrantsModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| EntryYear | public | int? | get, set | Год поступления абитуриента. |
| EducationProgram | public | string | get, set | Образовательная программа. |
| NameSearch | public | string | get, set | ФИО для поиска. |
| RegistrationRegion | public | string | get, set | Регион регистрации абитуриента. |
| IsDormitoryNeeded | public | bool? | get, set | Необходимость общежития на время обучения. |
| HasTargetQuota | public | bool? | get, set | Наличие поступления по целевой квоте. |
| HasSpecialQuota | public | bool? | get, set | Наличие особой квоты для поступления. |
| IsDocumentsReturned | public | bool? | get, set | Возврат документов. |
| MinTotalUse | public | int? | get, set | Минимальный суммарный балл ЕГЭ. |
| MaxTotalUse | public | int? | get, set | Максимальный суммарный балл ЕГЭ. |
| MinRussianLanguageUSE | public | int? | get, set | Минимальный балл ЕГЭ по русскому языку. |
| MaxRussianLanguageUSE | public | int? | get, set | Максимальный балл ЕГЭ по русскому языку. |
| MinInformaticsUSE | public | int? | get, set | Минимальный балл ЕГЭ по информатике. |
| MaxInformaticsUSE | public | int? | get, set | Максимальный балл ЕГЭ по информатике. |
| MinMathUSE | public | int? | get, set | Минимальный балл ЕГЭ по математике. |
| MaxMathUSE | public | int? | get, set | Максимальный балл ЕГЭ по математике. |

Таблица 5.4

Описание и функциональное назначение свойств класса GetStaticticByRegionsModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| RegionStatictic | public | List<RegionStaticticModel> | get, set | Статистика по регионам. |
| Total | public | int | get, set | Общее количество регионов. |

Таблица 5.5

Описание и функциональное назначение свойств класса PaginationModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Skip | public | int | get, set | Происходит отбор части данных: данное свойство пропускает переданное параметром количество данных. |
| Take | public | int | get, set | Происходит отбор части данных: данное свойство извлекает переданное параметром количество данных. |

Таблица 5.6

Описание и функциональное назначение свойств класса RegionStaticticModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Region | public | string | get, set | Название региона. |
| Count | public | int | get, set | Количество регионов. |

Таблица 5.7

Описание и функциональное назначение полей и методов класса EntrantRepository

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип** | **Назначение** | |
| \_dbSet | protected | DbSet<Entrant> | Представляет набор абитуриентов, которые хранятся в базе данных. | |
| \_context | protected | ApplicationContext | Определяет контекст данных, используемый для взаимодействия с базой данных. | |
| \_transaction | private | IDbContextTransaction | Определяет объект транзакции базы данных. | |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| EntrantRepository | public | EntrantRepository | ApplicationContext | Конструктор для инициализации репозитория и внедрения зависимостей. |
| BeginTransaction | public | void | без параметров | Определяет начало транзакции базы данных. |
| RollbackTransaction | public | void | без параметров | Отменяет изменения, сделанные после начала транзакции в базе данных. |
| CommitTransaction | public | void | без параметров | Подтверждает успешное завершение транзакции и сохраняет результаты изменений в базе данных. |
| AddEntrant | public | long | Entrant | Добавление абитуриента в базу данных. |
| AddRange | public | void | List<Entrant> | Добавление списка абитуриентов в базу данных. |
| UpdateEntrant | public | void | Entrant | Редактирование абитуриента в базе данных. |
| RemoveEntrant | public | void | Entrant | Удаление абитуриента из базы данных. |
| FindEntrantById | public | Entrant | long | Поиск абитуриента в базе данных по индивидуальному порядковому номеру. |
| GetAllEntrants | public | FilteredEntrantsModel | PaginationModel | Представляет всех абитуриентов из базы данных. |
| RemoveAllEntrants | public | void | без параметров | Удаляет всех абитуриентов из базы данных. |
| GetEntrantsByEntryYear | public | FilteredEntrantsModel | int, PaginationModel | Представляет абитуриентов по указанному году поступления. |
| GetStaticticByRegions | public | GetStaticticByRegionsModel | FilterEntrantsModel, PaginationModel | Представляет статистику по регионам. |
| FilterEntrants | public | FilteredEntrantsModel | FilterEntrantsModel, PaginationModel | Определяет фильтр абитуриентов. |
| AddDocumentsReturnedFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, bool? | Добавляет фильтр по возврату документов. |
| AddEntryYearFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, int?, int? | Добавляет фильтр по году поступления. |
| AddRussianLanguageUSEFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, int?, int? | Добавляет фильтр по баллу ЕГЭ по русскому языку. |
| AddMathUSEFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, int?, int? | Добавляет фильтр по баллу ЕГЭ по математике. |
| AddInformaticsUseFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, int?, int? | Добавляет фильтр по баллу ЕГЭ по информатике. |
| AddTotalUseFilter | private | IQueryable<Entrant> | IQueryable<Entrant>, int?, int? | Добавляет фильтр по суммарному баллу ЕГЭ. |

Таблица 5.8

Описание и функциональное назначение свойств и методов класса ApplicationContext

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Entrants | public | DbSet<Entrant> | get, set | Представляет набор абитуриентов, которые хранятся в базе данных. |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| ApplicationContext | public | ApplicationContext | DbContextOptions | Определяет контекст данных, используемый для взаимодействия с базой данных. |
| OnConfiguring | protected | void | DbContextOptionsBuilder | Определение конфигурация подключения. |

Таблица 5.9

Описание и функциональное назначение методов класса ApplicationContextFactory

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| CreateDbContext | public | ApplicationContext | string[] | Создает контекст базы данных. |

Таблица 5.10

Описание и функциональное назначение методов класса StartupExtension

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| AddDBContext | public | void | IServiceCollection, string | Добавление контекста в базу данных. |
| AddRepositories | public | void | IServiceCollection | Настройка внедрения зависимостей для репозитория. |

# ПРИЛОЖЕНИЕ 6.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ ПРОЕКТА ADMISSIONCAMPAIGN.SERVICES**

Таблица 6.1

Описание и функциональное назначение классов проекта AdmissionCampaign.Services

|  |  |
| --- | --- |
| **Класс** | **Функциональное назначение** |
| **CsvParser** | Класс для конвертации CSV файлов. |
| **EntrantMapper** | Класс, позволяющий проецировать одну модель абитуриента на другую. |
| **EntrantService** | Класс, содержащий логику обработки абитуриентов. |
| **StartupExtension** | Класс содержит методы расширения для первоначальной настройки DataAccess проекта. |

# ПРИЛОЖЕНИЕ 7.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ ПРОЕКТА ADMISSIONCAMPAIGN.SERVICES**

Таблица 7.1

Описание и функциональное назначение методов класса CsvParser

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| Parse | public | List<string> | string | Конвертирует файлы формата CSV. |

Таблица 7.2

Описание и функциональное назначение методов класса EntrantMapper

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| MapEntrantViewModelToEntity | public | void | AddEntrantViewModel, Entrant | Проецирует модель представления абитуриента в сущность. |
| MapEntrantEntityToView | public | EntrantView | Entrant | Проецирует сущность абитуриента в модель представления. |

Таблица 7.3

Описание и функциональное назначение полей и методов класса EntrantService

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Поля:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип** | **Назначение** | |
| \_entrantRepository | private | EntrantRepository | Репозиторий абитуриентов. | |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| EntrantService | public | EntrantService | EntrantRepository | Конструктор для инициализации сервиса и внедрения зависимостей. |
| AddEntrant | public | long | AddEntrantViewModel | Добавляет абитуриента в систему. |
| RemoveAllEntrants | public | void | без параметров | Удаляет всех абитуриентов из системы. |
| UpdateEntrant | public | void | UpdateEntrantViewModel | Редактирует абитуриента в системе. |
| RemoveEntrant | public | void | long | Удаляет абитуриента из системы. |
| FindEntrantById | public | EntrantView | long | Находит абитуриента в системе по индивидуальному порядковому номеру. |
| GetAllEntrants | public | FiltredEntrantsView | PaginationViewModel | Предоставляет всех абитуриентов системы. |
| GetPaginationModel | private | PaginationModel | PaginationViewModel | Предоставляет модель пагинации. |
| GetEntrantsByEntryYear | public | FiltredEntrantsView | GetEntrantsByEntryYearViewModel | Предоставляет абитуриентов из системы по указанному году поступления. |
| FilterEntrants | public | FiltredEntrantsView | FilterEntrantsViewModel | Фильтр абитуриентов. |
| GetStaticticByRegions | public | GetStaticticByRegionsView | GetStaticticByRegionsViewModel | Предоставляет статистику абитуриентов по регионам из системы. |
| UploadDocument | public | void | int, string, Stream | Загружает документ в формате CSV в систему. |
| ParseEntrantFromCsv | private | Entrant | string | Конвертирует студента из специального файла в формате CSV. |
| ParseStringToInt | private | int | string | Конвертирует строку в целочисленный тип. |
| ParseStringToBool | private | bool | string | Конвертирует строку в логический тип. |

Таблица 7.4

Описание и функциональное назначение методов класса StartupExtension

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Методы:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Типы параметров** | **Назначение** |
| AddServices | public | void | IServiceCollection, string | Настройка внедрения зависимостей для сервисов. |

# ПРИЛОЖЕНИЕ 8.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ ПРОЕКТА ADMISSIONCAMPAIGN.VIEWMODELS**

Таблица 8.1

Описание и функциональное назначение классов проекта AdmissionCampaign.ViewModels

|  |  |
| --- | --- |
| **Класс** | **Функциональное назначение** |
| **AddEntrantViewModel** | Класс представляет модель запроса для добавления абитуриента. |
| **EntrantView** | Класс представляет ответ абитуриента на запрос. |
| **FilterEntrantsViewModel** | Класс представляет модель запроса для фильтрации абитуриентов. |
| **FiltredEntrantsView** | Класс представляет модель ответа на запрос фильтрации абитуриентов. |
| **GetEntrantsByEntryYearViewModel** | Класс представляет модель запроса для абитуриентов, поступивших в указанный год. |
| **GetStaticticByRegionsView** | Класс представляет модель ответа на запрос статистики абитуриентов по регионам. |
| **GetStaticticByRegionsViewModel** | Класс представляет модель запроса статистики абитуриентов по регионам. |
| **PaginationViewModel** | Класс представляет модель запроса отбора нужной части данных и создания механизма их постраничного вывода. |
| **RegionStaticticView** | Класс представляет модель ответа на запрос статистики по регионам. |
| **UpdateEntrantViewModel** | Класс представляет модель запроса для редактирования данных абитуриента. |

# ПРИЛОЖЕНИЕ 9.

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ МЕТОДОВ И СВОЙСТВ ПРОЕКТА ADMISSIONCAMPAIGN.VIEWMODELS**

Таблица 9.1

Описание и функциональное назначение свойств класса AddEntrantViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| EntryYear | public | int | get, set | Год поступления абитуриента. |
| EducationProgram | public | string | get, set | Образовательная программа. |
| RegistrationNumber | public | int | get, set | Регистрационный номер абитуриента. |
| FullName | public | string | get, set | Полное имя абитуриента. |
| SNILS | public | string | get, set | СНИЛС абитуриента. |
| ApplicationType | public | string | get, set | Заявление абитуриента о согласии на зачисление. |
| WithoutExamsReason | public | string | get, set | Право поступления без вступительных испытаний. |
| SpecialQuota | public | bool | get, set | Наличие особой квоты для поступления. |
| TargetQuota | public | bool | get, set | Наличие поступления по целевой квоте. |
| InformaticsUSE | public | int | get, set | Балл ЕГЭ абитуриента по информатике. |
| MathUSE | public | int | get, set | Балл ЕГЭ абитуриента по математике. |
| RussianLanguageUSE | public | int | get, set | Балл ЕГЭ абитуриента по русскому языку. |
| IndividualAchievementScore | public | int | get, set | Балл абитуриента за индивидуальные достижения. |
| EducationForm | public | string | get, set | Предполагаемая форма обучения. |
| PreemptiveRight | public | string | get, set | Наличие преимущественного права при поступлении. |
| IsDormitoryNeeded | public | bool | get, set | Необходимость общежития на время обучения. |
| DocumentsReturned | public | bool | get, set | Возврат документов. |
| Specialization | public | string | get, set | Специализация. |
| EducationCompetitions | public | string | get, set | Все конкурсы. |
| RegistrationRegion | public | string | get, set | Регион регистрации абитуриента. |

Таблица 9.2

Описание и функциональное назначение свойств класса EntrantView

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** |  |  |  |  |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Id | public | long | get, set | Индивидуальный порядковый номер абитуриента в базе данных. |
| EntryYear | public | int | get, set | Год поступления абитуриента. |
| EducationProgram | public | string | get, set | Образовательная программа. |
| RegistrationNumber | public | int | get, set | Регистрационный номер абитуриента. |
| FullName | public | string | get, set | Полное имя абитуриента. |
| SNILS | public | string | get, set | СНИЛС абитуриента. |
| ApplicationType | public | string | get, set | Заявление абитуриента о согласии на зачисление. |
| WithoutExamsReason | public | string | get, set | Право поступления без вступительных испытаний. |
| SpecialQuota | public | bool | get, set | Наличие особой квоты для поступления. |
| TargetQuota | public | bool | get, set | Наличие поступления по целевой квоте. |
| InformaticsUSE | public | int | get, set | Балл ЕГЭ абитуриента по информатике. |
| MathUSE | public | int | get, set | Балл ЕГЭ абитуриента по математике. |
| RussianLanguageUSE | public | int | get, set | Балл ЕГЭ абитуриента по русскому языку. |
| IndividualAchievementScore | public | int | get, set | Балл абитуриента за индивидуальные достижения. |
| CompetitionPoints | public | int | get, set | Суммарный конкурсный балл. |
| EducationForm | public | string | get, set | Предполагаемая форма обучения. |
| PreemptiveRight | public | string | get, set | Наличие преимущественного права при поступлении. |
| IsDormitoryNeeded | public | bool | get, set | Необходимость общежития на время обучения. |
| DocumentsReturned | public | bool | get, set | Возврат документов. |
| Specialization | public | string | get, set | Специализация. |
| EducationCompetitions | public | string | get, set | Все конкурсы. |
| RegistrationRegion | public | string | get, set | Регион регистрации абитуриента. |

Таблица 9.3

Описание и функциональное назначение свойств класса FilterEntrantsViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| EntryYear | public | int? | get, set | Год поступления абитуриента. |
| EducationProgram | public | string | get, set | Образовательная программа. |
| NameSearch | public | string | get, set | ФИО для поиска. |
| RegistrationRegion | public | string | get, set | Регион регистрации абитуриента. |
| IsDormitoryNeeded | public | bool? | get, set | Необходимость общежития на время обучения. |
| HasTargetQuota | public | bool? | get, set | Наличие поступления по целевой квоте. |
| HasSpecialQuota | public | bool? | get, set | Наличие особой квоты для поступления. |
| IsDocumentsReturned | public | bool? | get, set | Возврат документов. |
| MinTotalUse | public | int? | get, set | Минимальный суммарный балл ЕГЭ. |
| MaxTotalUse | public | int? | get, set | Максимальный суммарный балл ЕГЭ. |
| MinRussianLanguageUSE | public | int? | get, set | Минимальный балл ЕГЭ по русскому языку. |
| MaxRussianLanguageUSE | public | int? | get, set | Максимальный балл ЕГЭ по русскому языку. |
| MinInformaticsUSE | public | int? | get, set | Минимальный балл ЕГЭ по информатике. |
| MaxInformaticsUSE | public | int? | get, set | Максимальный балл ЕГЭ по информатике. |
| MinMathUSE | public | int? | get, set | Минимальный балл ЕГЭ по математике. |
| MaxMathUSE | public | int? | get, set | Максимальный балл ЕГЭ по математике. |

Таблица 9.4

Описание и функциональное назначение свойств класса FiltredEntrantsView

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Entrants | public | List<EntrantView> | get, set | Список отфильтрованных абитуриентов. |
| Total | public | int | get, set | Общее количество отфильтрованных абитуриентов. |

Таблица 9.5

Описание и функциональное назначение свойств класса GetEntrantsByEntryYearViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| EntryYear | public | int | get, set | Год поступления абитуриента. |

Таблица 9.6

Описание и функциональное назначение свойств класса GetStaticticByRegionsView

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| RegionStatictic | public | List<RegionStaticticView> | get, set | Статистика по регионам. |
| Total | public | int | get, set | Общее количество регионов. |

Таблица 9.7

Описание и функциональное назначение свойств класса GetStaticticByRegionsViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| EntryYear | public | int? | get, set | Год поступления. |
| EducationProgram | public | string | get, set | Образовательная программа. |

Таблица 9.8

Описание и функциональное назначение свойств класса PaginationViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Page | public | int | get, set | Номер страницы для представления. |
| ItemsPerPage | public | int | get, set | Количество элементов на одной странице. |

Таблица 9.9

Описание и функциональное назначение свойств класса RegionStaticticView

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Region | public | string | get, set | Название региона. |
| Count | public | int | get, set | Общее количество регионов. |

Таблица 9.10

Описание и функциональное назначение свойств класса UpdateEntrantViewModel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Свойства:** | | | | |
| **Имя** | **Модификатор дотсупа** | **Тип возвращаемого значения** | **Доступ** | **Назначение** |
| Id | public | long | get, set | Индивидуальный порядковый номер в базе данных для редактирования информации об абитуриенте. |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированх |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |