**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ ГОРОДА МОСКВЫ**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ГОРОДА МОСКВЫ**

**«МОСКОВСКИЙ КОЛЛЕДЖ УПРАВЛЕНИЯ, ГОСТИНИЧНОГО БИЗНЕСА И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ «ЦАРИЦЫНО»**

**ДИПЛОМНЫЙ ПРОЕКТ**

Тема: Разработка информационной системы для автоматизации управления персоналом на предприятиях с почасовой формой оплаты труда.

Пояснительная записка

ДП.09.02.07.41.16.2022.01 ПЗ

Руководитель проекта: Воронина С.Ю. Дипломник: Мозырский А.Д.

Рецензент: Адоньев М.С.

**Допущен к защите на заседании**

**Государственной экзаменационной комиссии**

Учебно-методический отдел отделения УИТ

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2022 г. \_\_\_\_\_\_\_\_\_\_ Т.Н. Михайлова

Москва, 2022

|  |  |
| --- | --- |
| **Согласовано**  На заседании Кафедры  информационных технологий  Протокол № 11 от «30» марта 2022 г.  Заведующий кафедрой  \_\_\_\_\_\_\_\_\_\_\_\_Е.Ф. Писчасова | **Утверждаю**  **УМО ОУИТ**  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2022г.  \_\_\_\_\_\_\_\_\_\_ Т.Н. Михайлова |

**ЗАДАНИЕ НА ДИПЛОМНОЕ ПРОЕКТИРОВАНИЕ**

Студенту Мозырскому Андрею Дмитриевичу

группы ИС4-1 специальности 09.02.07 Информационные системы и программирование

Дата выдачи задания: «19» апреля 2022 г.

Срок сдачи проекта: «27» мая 2022 г.

**Тема дипломного проекта:**

Разработка информационной системы для автоматизации управления персоналом на предприятиях с почасовой формой оплаты труда.

**Техническое задание**

1. **Исходные данные** информация по персонале
2. **Задачи**:

* Разработать программный модуль для распределение плановых рабочих часов на сотрудников по отделам
* Разработать программный модуль для формирования дополнительных соглашений.
* Разработать алгоритм регистрации и авторизации пользователей;
* Разработать алгоритм приема и увольнения сотрудников;
* Разработать алгоритм расчета заработной платы сотрудников в соответствии с отработанными часами;
* Провести тестирование информационной системы;

|  |  |
| --- | --- |
| Раздел пояснительной записки | Объем, % |
| Введение | 6 |
| 1. Теоретическая часть  1.1. Исследование предметной области  1.2. Выбор средств разработки ПО | 30 |
| 2. Практическая часть  2.1. Разработка базы данных  2.2. Алгоритм работы Web Api  2.3. Разработка пользовательского интерфейса  2.4. Тестирование и отладка | 62 |
| Заключение | 1 |
| Список используемой литературы | 1 |
| Приложение 1 |  |

**Графическая часть:**

Мультимедиа презентация.

Дипломник: /Мозырский А.Д./

Руководитель проекта: /Воронина С.Ю./

Заведующий кафедрой: /Писчасова Е.Ф./

Содержание

*Изм.*

*Лист*

*№ докум.*

*Подпись*

*Дата*

*Лист*

4

**ДП.09.02.07.41.16.2022.01 ПЗ**

*Разраб.*

Мозырский А.Д.

*Пров.*

Воронина С.Ю.

*Реценз.*

Адоньев М.С.

*Н. контр.*

Воронина С.Ю.

*Утв.*

Писчасова Е.Ф.

Разработка информационной системы для автоматизации управления персоналом на предприятиях с почасовой формой оплаты труда

Пояснительная записка

*Лит.*

*Листов*

*135*

**ГБПОУ Колледж «Царицыно»**

Д

[Введение 5](#_Toc104534543)

[1 Теоретическая часть 8](#_Toc104534544)

[1.1 Исследование предметной области 8](#_Toc104534545)

[1.2 Выбор средств разработки ПО 9](#_Toc104534546)

[2 Практическая часть 20](#_Toc104534547)

[2.1 Разработка базы данных 20](#_Toc104534548)

[2.2 Алгоритм работы Web Api 35](#_Toc104534549)

[2.3 Разработка пользовательского интерфейса 43](#_Toc104534550)

[2.4 Тестирование и отладка 46](#_Toc104534551)

[Заключение 50](#_Toc104534552)

# Введение

Рассматривая структуру любого предприятия с почасовой формой оплаты труда, можно выделить распространенные причины убытков: отставания от планов и графиков, постоянную текучку персонала, необходимого для обеспечения работоспособности компании и излишнее количество человеко-часов, уделяемых на составление однообразной документации, а именно, дополнительных соглашений с сотрудниками на каждый месяц.

Поскольку прибыль компании напрямую зависит от отработанных сотрудниками человеко-часов, полнота и правильность распределения рабочих часов между сотрудниками является ключевым фактором дохода компании.

Объектом исследования в данном дипломном проекте является процесс управления персоналом на предприятиях. Предметом является универсальная информационная система, позволяющая автоматизировать управление персоналом.

В настоящее время существует множество приложений, распространяемых как свободно, так и на коммерческой основе, позволяющих оптимизировать управление рабочей нагрузкой на работников компании. Среди них можно выделить: «1C-CRM», «Peopleforce», «Talantix» и т.д. Использование этих приложений возможно, однако, во-первых, они не способны оптимизировать процесс формирования дополнительных соглашений, а во-вторых, имеют ряд существенных ограничений.

При использовании перечисленных выше систем процесс тайм-менеджмента персонала лишь немного оптимизирован, но не автоматизирован полностью, оперативное изменение распределения рабочих часов невозможно. Также стоит отметить, что при создании базы данных, приведенные системы запрашивают очень большой объем информации, которая, как правило, при работе с информационной системой затем не используется.

Учитывая все недостатки, складывается необходимость разработки информационной системы для автоматизации управления персоналом на предприятиях с почасовой формой оплаты труда, которая при минимальном наборе данных будет адекватно распределять рабочие часы между отделами и сотрудниками компании, с возможностью внесения изменений в нестандартных ситуациях. Поэтому целью дипломного проекта является проектирование базы данных для универсальной платформы управления персоналом предприятия и разработка масштабируемого WebAPI реализующего базовый функционал HRM (Human Resource Management), а также обеспечивающего распределение рабочей нагрузки и формирование дополнительных соглашений.

Качество работы универсальной информационной системы определяется возможностью эффективного распределения рабочей нагрузки между отделами и персоналом организации, автоматическими расчетами заработной платы сотрудников, оперативным реагированием на изменение стандартов рабочего времени. Руководители такой компании ведут учет персонала, структуры отделов и их рабочей нагрузки. Решают, сколько часов нужно отработать компании за определенный период времени для получения прибыли, сколько должен будет отработать каждый отдел и сотрудник для формирования дополнительных соглашений. Сотрудники имеют возможность просматривать историю своих дополнительных соглашений, получать их на почту или сохранять на устройстве в виде .pdf документа.

Исходя из цели, в рамках дипломного проекта необходимо реализовать следующие задачи:

1. Исследование методов распределения рабочей нагрузки
2. Разработать универсальную базу данных управления персоналом предприятия
3. Разработать алгоритмы работы WebAPI
4. Реализовать взаимодействие WebAPI с базой данных
5. Разработать пользовательский интерфейс клиентского приложения
6. Провести комплексное тестирование основных функций WebAPI

# Теоретическая часть

## 1.1 Исследование предметной области

При рассмотрении объекта исследования и статистики различных организаций был сделан вывод о том, что, несмотря на специфику различных предприятий, руководители компаний стремятся усреднить рабочую нагрузку на персонал.

В зависимости от должностных обязанностей диапазон рабочей нагрузки варьируется от 80 до 250 часов в месяц. Однако статистика показывает, что наиболее частая рабочая нагрузка составляет 150 ч. +- 5 ч.

Вследствие этого было решено, что модуль распределения рабочей нагрузки разрабатываемой системы будет иметь следующие ограничения:

1. Минимальная рабочая нагрузка не должна быть меньше, чем 4 ч. в день \* кол-во рабочих дней в текущем месяце
2. Максимальная рабочая нагрузка не должна превышать 10 ч. в день \* кол-во рабочих дней в текущем месяце

Распределение предполагает, что все сотрудники конкретного отдела получат равноценную рабочую нагрузку. Различные отделы могут иметь различные требования к персоналу, поэтому для каждого отдела можно задать свои стандарты рабочего времени.

Возможно подтверждение статистики в виде графиков

## 1.2 Выбор средств разработки ПО

Описание современных СУБД

СУБД представляет собой совокупность специальных языковых и программных средств, облегчающих пользователям выполнение всех операций, связанных с организацией хранения данных, их корректировкой и доступом к ним.

Современные реляционные СУБД обеспечивают:

1. Набор средств для поддержки таблиц и соотношений между связанными таблицами;
2. Развитый пользовательский интерфейс, позволяющий вводить и модифицировать информацию, выполнять поиск;
3. Средства программирования для разработки собственных приложений.

Для выполнения данного проекта рассматриваются несколько СУБД:

MS SQL

Довольно популярная СУБД, которая является программным средством, разработанным компанией Microsoft. Данная СУБД, располагается на как на облачных, так и на локальных серверах, причем возможно комбинировать применяемые сервера одновременно. Еще программный продукт Microsoft, имеет лицензированную поддержку даже для бесплатной версии, большое сообщество, которое может быть полезным в решении возникших трудностей и задач, а также большую базу документации, также переведенной на русский язык.

Актуальная версия Microsoft SQL сервер имеет возможность поддержки ddm (динамическую маскировку данных), которая гарантирует, что только авторизованные пользователи с установленными правами будут видеть данные. MS SQL Server имеет умный мастер импорта, также можно сформировать скрипт базы данных. В процессе изучения СУБД были выявлены следующие преимущества:

1. СУБД довольно проста;
2. Актуальная версия достаточно быстра и стабильна;
3. Доступна в бесплатном варианте для физических лиц;

И недостатки:

1. Высокая цена за платную версию;
2. Занимает все доступные аппаратные ресурсы.

Идеально подходит для крупных организаций, которые уже используют ряд продуктов Microsoft.

PostgreSQL

PostgreSQL это бесплатная и довольно популярная СУБД, которая зачастую используется для создания и ведения баз данных web-сайтов. Это одна из первых СУБД, и поэтому сейчас ее функциональность очень хорошо развита, что позволяет пользователям управлять как структурированными, так и неструктурированными данными. Отлично показывает себя с задачами по импорту данных из других типов баз с помощью собственных возможностей.

Наиболее актуальная версия PostgreSQL дает возможность обработки огромных объемов данных и значительное увеличение одновременно задействованных пользователей. В процессе изучения СУБД были выявлены следующие преимущества:

1. Имеет возможность к масштабированию и способен обрабатывать большое количество данных;
2. Поддерживает формат json, что расширяет возможности;
3. Имеет множество предопределенных возможностей и функций;
4. Доступен ряд разнообразных интерфейсов.

И недостатки:

1. Мало официальной документации;
2. Скорость работы может “прыгать” время от времени;
3. Сложный в освоении интерфейс.

Очень хорошо подходит для пользователей с малым бюджетом. Однако необходим специалист высокого уровня, для возможности выбрать интерфейс и использовать json, без потери в скорости.

MySQL

MySQL - наверное, наиболее популярная СУБД с богатой, бесплатной функциональностью. Хоть это и бесплатная СУБД, но обновления к ней приходят постоянно, расширяя ее возможности и модернизируя систему безопасности. В ней так же есть разнообразные платные версии, предназначенные для коммерческого пользования. Бесплатная версия программы производит основной упор в надежность СУБД и ее скорость, а не на вариативность функциональных возможностей.

СУБД MySQL обеспечивается поддержкой большого количества разнообразных типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Кроме того, MySQL предоставляется с особыми типами таблиц EXAMPLE. СУБД также имеет несложный интерфейс, обычные функции которого не требуют специфичной подготовки. Система надежна и не стремится использовать под себя все доступные ресурсы. В процессе изучения СУБД были выявлены следующие преимущества:

1. Бесплатное распространение;
2. Хорошо документирована на разных языках;
3. Различное множество функции;
4. Имеет возможность поддержки разнообразных пользовательских интерфейсов.

И недостатки:

1. Нет предустановленных возможностей решения простых задач;
2. В бесплатной версии отсутствует поддержка.

Подходит для тех, кому необходим бесплатный, но надежный инструмент управления базами данных.

Обоснование выбранной СУБД

Исходя из целей данной работы, необходимо реализовать хранение часто меняющегося списка сотрудников компании, для обеспечения целостности данных, в соответствующей таблице должен быть предусмотрен атрибут id, содержащий в себе уникальное целочисленное значение. Для автоматизации заполнения данного атрибута в MS SQL предусмотрено свойство IDENTITY, которое по умолчанию запрещает пользователю изменять значение атрибута. Похожая функция есть и в других СУБД. В PostgreSQL для этого используется отдельный тип данных serial, что накладывает множество ограничений. В MySQL существует свойство AUTO\_INCREMENT, который скорее выполняет роль функции, подставляющей значение на основе уже заданных в данном столбце, то есть не ограничивает пользовательский ввод атрибута и сбрасывает структуру нумерации во время удаления строк. Эта особенность может негативно повлиять на качество базы данных.

При работе HR, нельзя исключать вероятность того, что при подаче резюме для работы, кандидат может случайно отправить его дважды или заполнить не полностью, рассмотрим для примера обработку ввода данных в атрибут email. Для обеспечения уникальности введенных значений язык SQL содержит ограничение UNIQUE, который по-разному работает в разных СУБД. При использовании MS SQL, данное ограничение гарантирует не только уникальность данных, но и то, что несколько строк не будут содержать NULL значения, в отличии от MySQL, в котором это допускается.

Исходя из данных особенностей работы разных СУБД, для реализации данного проекта самым подходящим вариантом является MS SQL. Так как разрабатываемое ПО подлежит внедрению в уже существующие компании, необходим быстрый и качественный импорт информации, которая ранее хранилась вне базы данных, в частности из файлов с расширением .xlsx и .csv. MS SQL предоставляет возможность импортировать и экспортировать данные из любых источников с помощью встроенного программного решения.

Описание современных сред разработки

Среда разработки представляет из себя соединённые в одно решение программные средства, целью которой является создание и отладка программного продукта.

Среда разработки объединяет в себе:

1. Компилятор – программа, предназначенная для считывания исходного кода, который она преобразует в программный продукт.
2. Интерпретатор – программа, задачей которой является считывание команд, скрытых в исходном коде, и их последующее выполнение.
3. Отладчик – программа, ищущая ошибки в разрабатываемых программный продуктах, SQL-запросах и других разновидностях кода.
4. Среда автоматизации сборки – этап в процессе разработки ПП, который представляет из себя автоматизацию большого спектра задач, решаемых разработчиками.
5. Редактор текста – программа, функциональными возможностями которой являются создание и редактирование текстовых данных при разработке ПП, а также создания обычных текстовых файлов.

Если в среде разработки реализованы все вышеперечисленные компоненты, то тогда ее можно назвать интегрированной. Интегрированная среда разработки позволяет разработчику выбрать подходящий для данной задачи язык программирования (из языков, поддерживаемых данной средой).

Visual Studio

Visual Studio — это IDE, позволяющая: отлаживать, редактировать и создавать код. Кроме стандартного отладчика и редактора, которые имеют место в большинстве IDE, Visual Studio имеет в своем арсенале средства выполнения кода, компиляторы, графические конструкторы и большое количество других функций для повышения комфорта разработки. В процессе изучения интегрированных средств разработки были выявлены следующие преимущества:

1. Учитывая то, что IDE создана в Microsoft очевидно, что она хорошо взаимодействует с другими продуктами данной компании. Кроме того, в определенных ситуациях без Visual Studio не обойтись — например, при реализации проекта WPF.
2. Функционала «Community edition» (бесплатная версия) для обычного пользователя будет достаточно.
3. Visual Studio имеет огромное количество разнообразных плагинов. Используя их, можно расширить функциональные возможности, а также подключать другие языки программирования.
4. Имеет поддержку платформы .NET. Данная среда разработки обладает широкими возможности по созданию приложений для Windows, в том числе в .NET-сегменте.

И недостатки:

1. При долгом использовании могут появляться баги, не срабатывать нажатия, нарушаться работа с сервером.
2. Самоличное освоение Visual Studio новичком является довольно непростой задачей — большое количество функциональных возможностей и скрытых настроек.

Подходит для опытных пользователей, а также для тех, кому нужен огромный функционал и постоянная поддержка компании разработчика.

JetBrains Rider

JetBrains Rider — кроссплатформенная интегрированная среда разработки программных продуктов для платформы .NET. Поддерживает разнообразные языки программирования, среди которых: C#, VB.NET и F#. Данная среда разработки поддерживает .NET Framework, новые кроссплатформенные .NET Core и моно-проекты. Это позволяет создавать широкий спектр приложений, включая: службы и библиотеки, игры Unity, приложения Xamarin, ASP.NET. В процессе изучения интегрированных средств разработки были выявлены следующие преимущества:

1. Особенность продуктов JetBrains, воссозданная в Project Rider. С ним вы сможете организовать весь цикл создания ПП: от идеи до поддержки.
2. Данная среда разработки дает возможность подключить MSBuild и XBuild, работать с CLI-проектами и организовать отладку приложений .NET и Mono.
3. Возможность поддержания нескольких запущенных программ.
4. Project Rider работает с Windows, Linux и MacOS.

И недостатки:

1. Так как проект новый, то часть обещанного функционала еще в разработке, а также не все стартовые баги исправлены.
2. Project Rider в самой дешевой сборке обойдется в довольно большую сумму за первый год использования. Хоть для данной среды разработки и есть триал-версия, но она обладает сильно урезанной функциональностью.

Может использоваться как опытными пользователями, так и новичками, однако слабо подходит для тех, кто прежде всего ценит бесперебойность работы.

Code::Blocks

Code::Blocks — представляет из себя кроссплатформенную среду разработки, которая существует между мощными средами по созданию больших проектов, типа Visual Studio, и слабыми по функционалу, но удобными блокнотами типа Sublime, причем преимущества и тех, и других сочетаются и позволяют использовать данную среду, как для создания небольших ПП для встраиваемых приложений, так и для программирования простых приложений для РС. В процессе изучения интегрированных средств разработки были выявлены следующие преимущества:

1. Проект полностью бесплатный (open-source).
2. Среда Code::Blocks проста в освоении, необходимо лишь знать один из предлагаемых языков.
3. Данная IDE возможно запустить на любой десктопной ОС.

И недостатки:

1. Довольно слабая функциональность для IDE. Для создания комплексных приложений Code::Blocks практически не подходит.
2. Довольно часто происходят ошибки в работе всего проекта.

Данный продукт подойдет для разработки небольших приложений и простого написания кода, как и хорошо подходит для обучения новых пользователей. Однако бесполезен при разработке крупных проектов.

Обоснование выбранной среды разработки

Для реализации данного проекта самым подходящим вариантом среды разработки является Visual Studio. Разрабатываемое приложение должно выполнять взаимодействие с базой данных, для чего необходим функционал тонкой отладки и дебагинга, который может обеспечить только Visual Studio.

В процессе разработки Web Api должна использоваться система контроля версий, Visual Studio помогает интегрировать Git и отслеживать состояние проекта.

Так как Visual Studio является продуктом компании Microsoft, он содержит множество предустановленных библиотек для работы с такими платформами как MS SQL и ASP.NET CORE и шаблонов различных проектов, что существенно ускоряет процесс разработки.

Функциональные возможности бесплатной версии, являются наиболее оптимальными для решения поставленной задачи. Visual Studio обладает не только развернутой функциональностью для создания Web Api, но и упрощает взаимодействие с MS SQL.

Описание современных case-средств

CASE-средство представляет собой набор специальных программных методов и технологий, которые помогают обеспечить отсутствие ошибок, высокое качество при проектировании и облуживании программного продукта. Главные составляющие CASE-продукта таковы:

1. Методология, которая задает единый графический язык и правила работы с ним;
2. Графические редакторы, которые позволяют создавать диаграммы;
3. Генератор: по графическому представлению модели можно сгенерировать исходный код для различных платформ;
4. Репозиторий, своеобразная база данных для хранения результатов работы программистов.

draw.io

draw.io – это Интернет-ресурс, который обладает большой функциональностью, разнообразием видов диаграмм разнообразных форм-факторов. Обладает удобным и гибким интерфейсом, а также возможностью его настройки под себя. Кроме того, разнообразие шаблонов позволяет экономить большое количество времени. В процессе изучения CASE-средства были выявлены следующие преимущества:

1. Богатый функциональный набор;
2. Бесплатная веб-версия;
3. Универсальная система;
4. Большое количество разнообразных шаблонов.

И недостатки:

1. Нет интеграции с СУБД;
2. Бесплатной является только веб-версия.

Подходит для тех, кому необходимо быстрое и удобное CASE-средство.

Toad Data Modeler

Toad Data Modeler – это средство позволяющие проектировать базы данных приложений. Понятный интерфейс позволяет даже неопытному пользователю успешно использовать данное CASE-средство. Кроме того, оно поддерживает разнообразные СУБД, что облегчает интегрирование модели. В процессе изучения CASE-средства были выявлены следующие преимущества:

1. Существует бесплатная учебная версия;
2. Поддержка СУБД;
3. Есть возможность написания SQL-скриптов;
4. Большой функционал возможностей в сфере применения.

И недостатки:

1. Платная полная версия;
2. Узкая сфера применения;
3. Документация только на английском.

Подходит для тех, кому нужен узконаправленный инструмент создания моделей баз данных.

Microsoft Office Visio

Microsoft Office Visio — графическая среда, разработанная и поддерживающаяся Microsoft. Имеет богатые функциональные возможности для реализации бизнес-планов, схем и документов. Так как данное средство является продуктом Microsoft, она хорошо совместима с остальными ее продуктами. В процессе изучения CASE-средства были выявлены следующие преимущества:

1. Большое количество документации;
2. Возможность создания диаграмм потоковых данных;
3. Возможности интеграции с продуктами Microsoft.

И недостатки:

1. Отсутствие удобности интерфейса;
2. Полный функционал является платным.

Отлично подходит для пользователей, которые уже используют ряд продуктов Microsoft.

Обоснование выбранного case-средства

Минимальная необходимая функциональность CASE-средства ограничена графическим редактором для построения разнообразных диаграмм.

Учитывая поставленную в проекте задачу, в качестве основного CASE-средства, был выбран бесплатный интернет ресурс draw.io, так как он обладает необходимыми функциональными возможностями, удобным и простым интерфейсом, а также большим количеством импортируемых шаблонов.

# Практическая часть

## Разработка базы данных

Исходя из предметной области в проекте были выделены следующие сущности:

Authorization, Employee, Interview, Candidate, Document, Dismissal, Department, Period, Department\_work\_load, Employee\_work\_load, Personal\_achivements, Passport\_info, Contact\_data

Более подробная информация об описании и назначении сущностей представлена в таблице 1.

| Номер сущности | Имя сущности | Определение | Описание |
| --- | --- | --- | --- |
| E1 | Authorization | Аккаунт пользователя информационной системы | Новый экземпляр сущности появляется при регистрации сотрудника в информационной системе |
| E2 | Employee | Человек, являющийся сотрудником компании | Новый экземпляр сущности появляется в результате прохождения собеседования |
| E3 | Interview | Сервис найма или отказа потенциального сотрудников | Новый экземпляр сущности появляется при прохождении кандидатом собеседования |
| E4 | Candidate | Человек, являющийся потенциальным сотрудником компании | Новый экземпляр сущности появляется при подаче кандидатом резюме на работу в компании |
| E5 | Document | Электронный документ, содержащий не обязательные данные кандидата | Новый экземпляр сущности появляется при прикреплении кандидатом к своему резюме документа |
| E6 | Dismissal | Сервис увольнения сотрудников | Новый экземпляр сущности появляется при увольнении сотрудника |
| E7 | Department | Структура персонала компании | Новый экземпляр сущности появляется при расширении направленности компании |
| E8 | Period | Временной интервал, длительностью 1 месяц | Новый экземпляр сущности появляется при назначении ответственным лицом рабочей нагрузке на следующий период |
| E9 | Department\_  work\_load | Электронный документ, содержащий данные о рабочей нагрузке на отдел в определенный период времени | Новый экземпляр сущности появляется при назначении ответственным лицом рабочей нагрузке на следующий период |
| E10 | Employee\_  work\_load | Электронный документ, содержащий данные о рабочей нагрузке на сотрудника в определенный период времени | Новый экземпляр сущности появляется при назначении ответственным лицом рабочей нагрузке на следующий период |
| E11 | Personal\_  achivements | Электронный документ, содержащий данные о добровольной сверх рабочей нагрузке на сотрудника | Новый экземпляр сущности появляется при перевыполнении плана работ сотрудником компании |
| Е12 | Passport\_info | Электронный документ, содержащий данные о кандидате или сотруднике | Новый экземпляр сущности появляется при подаче кандидатом резюме |
| Е13 | Contact\_data | Электронный документ, содержащий данные о способах связи с кандидатом или сотрудником | Новый экземпляр сущности появляется при подаче кандидатом резюме |

Таблица 1. Описание множества сущностей.

В качестве примера данных для импорта в базу данных приведены тестовые экземпляры для каждой сущности:

Authorization/E1: 0, NULL, admin, 12345, Administrator

Employee/E2: 1, 1, 1, 1, 1, true, 05f04074-a0d2-4abd-a039-72e3e8336f40

Interview/E3: 1, 1, True, 2020-12-09

Candidate/E4: 1, 1, 1, СПО, 0

Document/E5: 1, 1, Sertificate, localhost://document/plziwontthisjob.pdf

Dismissal/E6: 1, 1, 2021-11-17, 2021-12-17, По собственному желанию, 1000

Department/E7: 1, 50, Тех. обслуживание, 250, 250

Period/E8: 1, 2021, 11, 10000

Department\_work\_load/E9: 1, 1, 1, 495, 495, True

Employee\_work\_load/E10: 1, 1, 1, 165, 165, 41250

Personal\_achivements/E11: 1, 1, 1, Выход на рабочее место в нерабочий день, 500

Passport\_info/E12: 1, 4819, 462281, Милана, Яскунова, Денисовна, Самарская область, Россия, г. Домодедово, Максима Горького ул., 3, 2, 73,

Contact\_data/E13: 1, milana.yaskunova@rambler.ru, 89984254851

Для более наглядного обозначения зависимостей сущностей была разработана матрица связей, представленная в таблице 2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Authorization/E1 | Employee/E2 | Interview/E3 | Candidate/E4 | Document/E5 | Dismissal/E6 | Department/E7 | Period/E8 | Department\_wokload/E9 | Employee\_wok\_load/E10 | Personal\_achivements/E11 | Pasport\_info/E12 | Contact\_data/E13 |
| Authorization/E1 |  | R1 |  |  |  |  |  |  |  |  |  |  |  |
| Employee/E2 | R1 |  | R2 |  |  |  | R6 |  |  | R11 | R12 | R13 | R14 |
| Interview/E3 |  | R2 |  | R3 |  |  |  |  |  |  |  |  |  |
| Candidate/E4 |  |  | R3 |  | R4 |  |  |  |  |  |  | R15 | R16 |
| Document/E5 |  |  |  | R4 |  |  |  |  |  |  |  | R5 |  |
| Dismissal/E6 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Department/E7 |  | R6 |  |  |  |  |  |  | R10 |  |  |  |  |
| Period/E8 |  |  |  |  |  |  |  |  | R7 | R8 | R9 |  |  |
| Department\_work  \_load/E9 |  |  |  |  |  |  | R10 | R7 |  |  |  |  |  |
| Employee\_work  \_load/E10 |  | R11 |  |  |  |  |  | R8 |  |  |  |  |  |
| Personal  \_achivements/E11 |  | R12 |  |  |  |  |  | R9 |  |  |  |  |  |
| Passport\_info/E12 |  | R13 |  | R15 |  | R5 |  |  |  |  |  |  |  |
| Contact\_data/E13 |  | R14 |  | R16 |  |  |  |  |  |  |  |  |  |

Таблица 2. Матрица связей.

Далее, в таблице 3 представлено подробное описание связей сущностей и способы их реализации.

| Номер связи | Родительская сущность | Дочерняя сущность | Имя связи | Тип связи | Мощность | Описание |
| --- | --- | --- | --- | --- | --- | --- |
| R1 | Employee/E2 | Authorization/E1 | FK\_Authorization\_Employee | Определенная  Не идентифицирующая  Необязательная | Z | Один сотрудник может зарегистрироваться как пользователь или не делать этого, у каждого сотрудника может быть только 1 аккаунт |
| R2 | Interview/E3 | Employee/E2 | FK\_Employee\_Interview | Определенная  Не идентифицирующая  Обязательная | Z | По результатам одного интервью может быть нанят 1 сотрудник или не нанят никто |
| R3 | Candidate/E4 | Interview/E3 | FK\_Interview\_Candidate | Определенная  Не идентифицирующая  Обязательная | P | Один кандидат может несколько раз приходить на интервью, при этом одному сотруднику всегда соответствует одно пройденное интервью |
| R4 | Candidate/E4 | Document/E5 | FK\_Document\_Candidate | Определенная  Не идентифицирующая  Обязательная |  | Один кандидат может отправить различные документы, но может и не отправлять ничего |
| R5 | Passport\_info/E12 | Dismissal/E6 | FK\_Dismissal\_Passport\_info | Определенная  Не идентифицирующая  Обязательная | Z | Один сотрудник может быть уволен 1 раз, но может и не быть уволен |
| R6 | Department/E7 | Employee/E2 | FK\_Employee\_Department | Определенная  Не идентифицирующая  Обязательная |  | Один отдел может содержать множество сотрудников, но сотруднику не обязательно состоять в каком-либо отделе |
| R7 | Period/E8 | Department\_work\_load/E9 | FK\_Department\_work\_load\_Period | Определенная  Не идентифицирующая  Обязательная | P | Каждому периоду соответствует по одной нагрузке на каждый отдел, одной нагрузке на отдел соответствует один период |
| R8 | Period/E8 | Employee\_wok\_load/E10 | FK\_Employee\_work\_load\_Period | Определенная  Не идентифицирующая  Обязательная | P | Каждому периоду соответствует по одной нагрузке на каждого сотрудника, одной нагрузке на сотрудника соответствует один период |
| R9 | Period/E8 | Personal\_achivements/E11 | FK\_Personal\_achievements\_Period | Определенная  Не идентифицирующая  Обязательная |  | За каждый период может быть насколько достижений сотрудников, а может и не быть вообще, каждое достижение сотрудника привязано к конкретному периоду |
| R10 | Department/E7 | Department\_work\_load/E9 | FK\_Department\_work\_load\_Department | Определенная  Не идентифицирующая  Обязательная | P | Один отдел получает множество нагрузок (по 1 за период), нагрузка направленна на конкретный отдел |
| R11 | Employee/E2 | Employee\_wok\_load/E10 | FK\_Employee\_work\_load\_Employee | Определенная  Не идентифицирующая  Обязательная | P | Один сотрудник получает множество нагрузок (по 1 за период), нагрузка направленна на конкретного сотрудника |
| R12 | Employee/E2 | Personal\_achivements/E11 | FK\_Personal\_achievements\_Employee | Определенная  Не идентифицирующая  Обязательная |  | Каждый сотрудник может иметь насколько достижений, а может и не иметь вообще, каждое достижение привязано к конкретному сотруднику |
| R13 | Passport\_info/E12 | Employee/E2 | FK\_Employee\_Passport\_info | Определенная  Не идентифицирующая  Обязательная | Z | Каждому паспорту может соответствовать один сотрудник, а может 0 |
| R14 | Contact\_data/E13 | Employee/E2 | FK\_Employee\_Contact\_data | Определенная  Не идентифицирующая  Обязательная | P | Каждому паспорту соответствует один кандидат, одному кандидату соответствует один паспорт |
| R15 | Passport\_info/E12 | Candidate/E4 | FK\_Candidate\_Passport\_info | Определенная  Не идентифицирующая  Обязательная | Z | Каждому набору контактных данных может соответствовать один сотрудник, а может 0 |
| R16 | Contact\_data/E13 | Candidate/E4 | FK\_Candidate\_Contact\_data | Определенная  Не идентифицирующая  Обязательная | P | Каждому набору контактных данных соответствует один кандидат, одному кандидату соответствует один набор контактных данных |

Таблица 3. Описание связей

На диаграмме уровня сущностей (см. Рисунок 4) представлено графическое представление описания сущностей.

На диаграмме уровня ключей (см. Рисунок 5) представлена логика взаимодействия сущностей.

Полноатрибутная диаграмма представляет собой наиболее полное описание структуры базы данных (см. Рисунок 6)

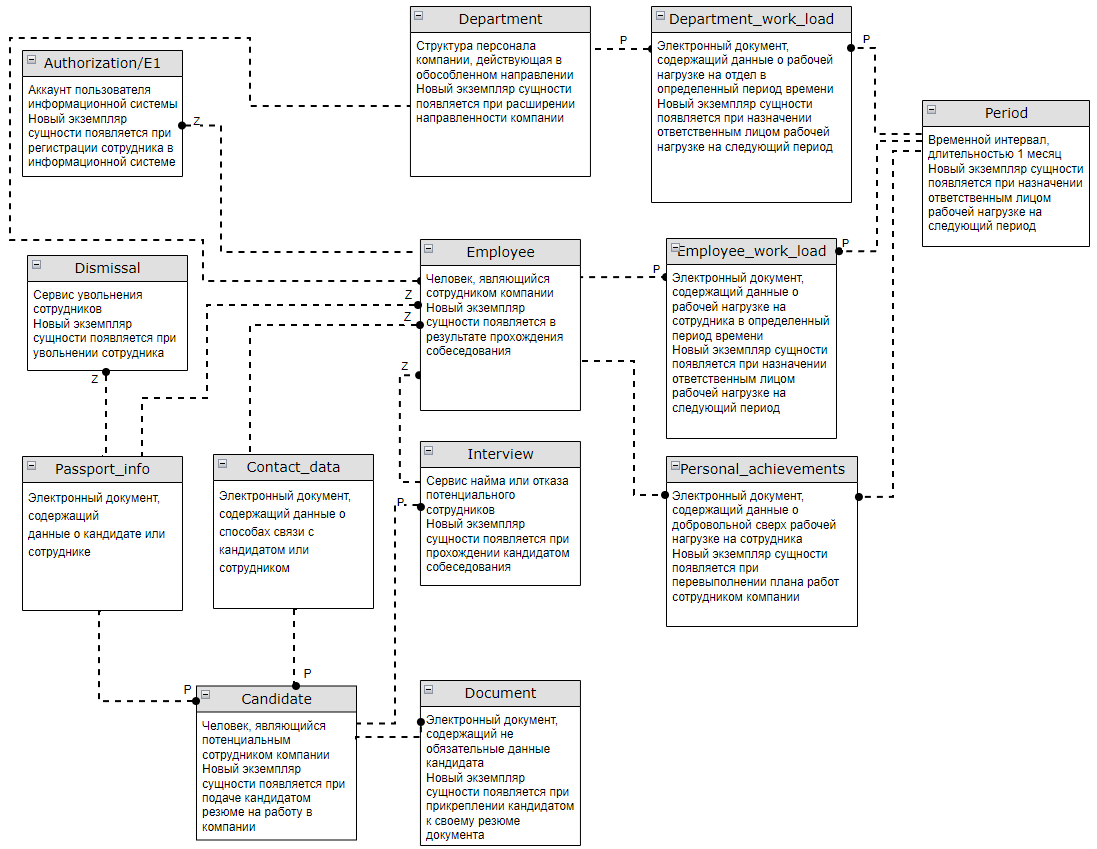


Рис. 4 Диаграмма уровня сущностей.

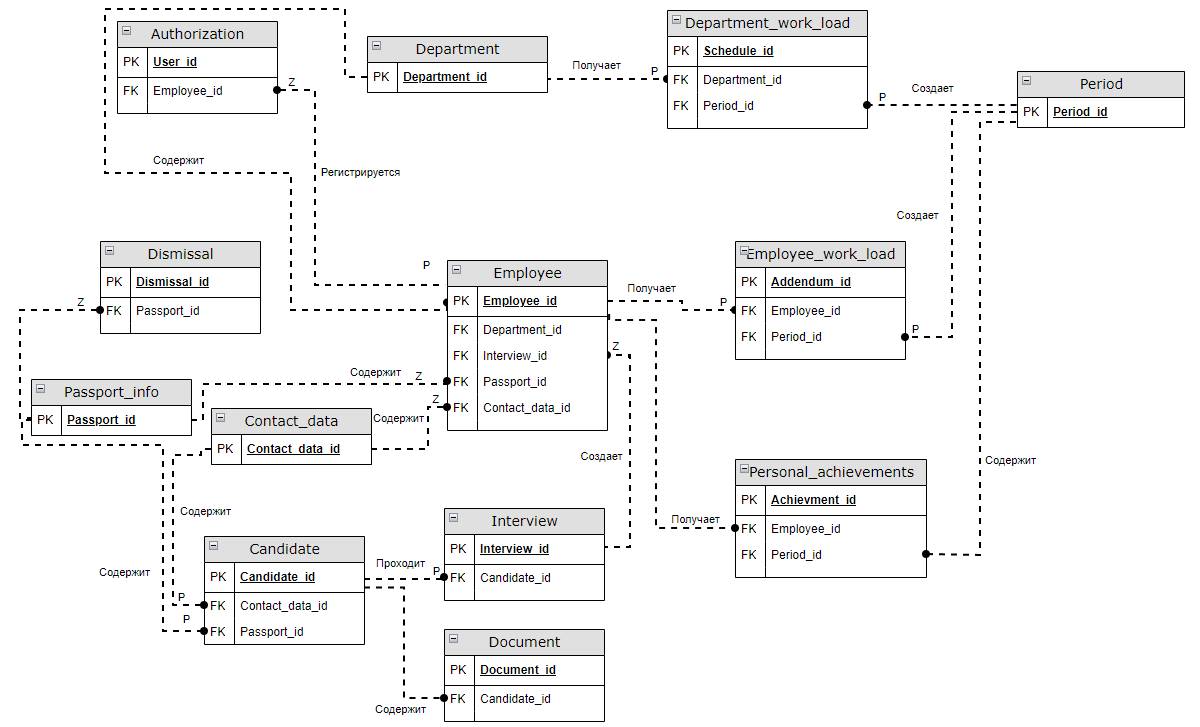


Рис. 5 Диаграмма уровня ключей.

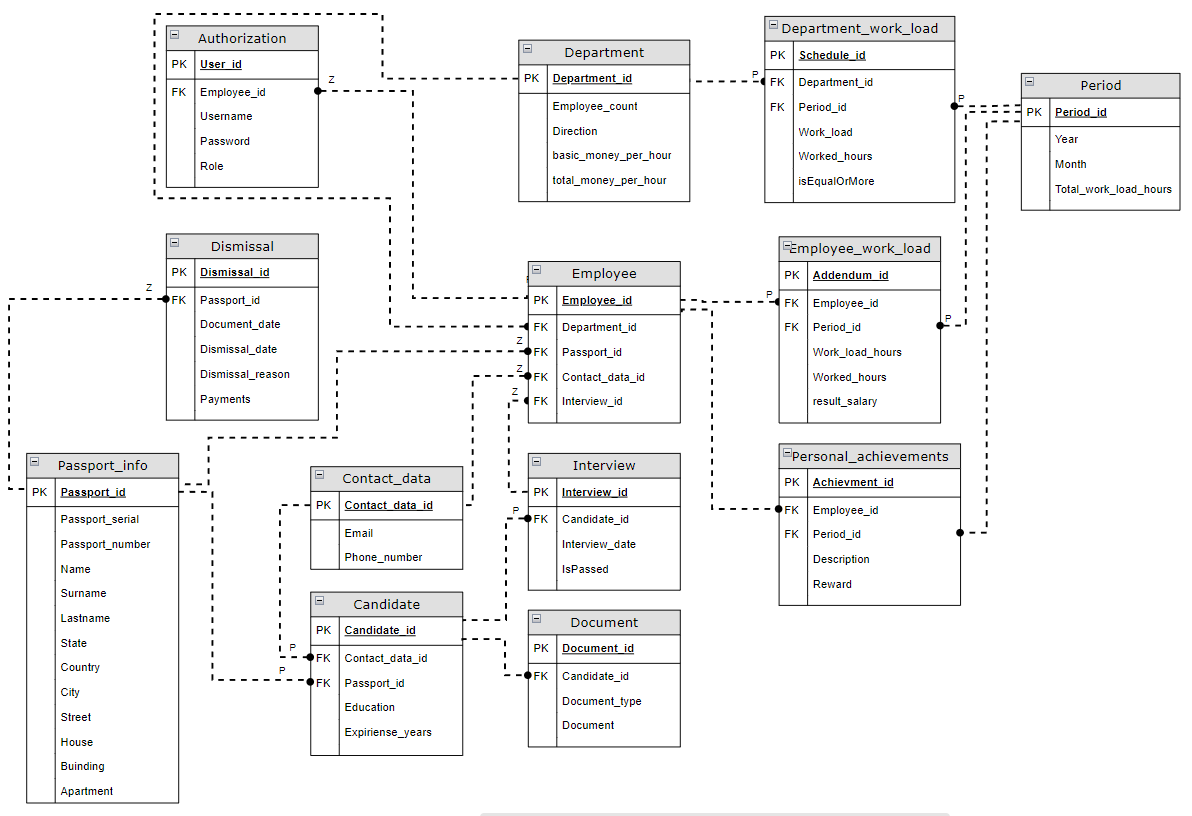


Рис. 6 Полноатрибутная диаграмма.

## Алгоритм работы Web Api

На начальном этапе разработки была сформирована диаграмма активности (рис. \*) на основе которой, проводилась разработка информационной системы.

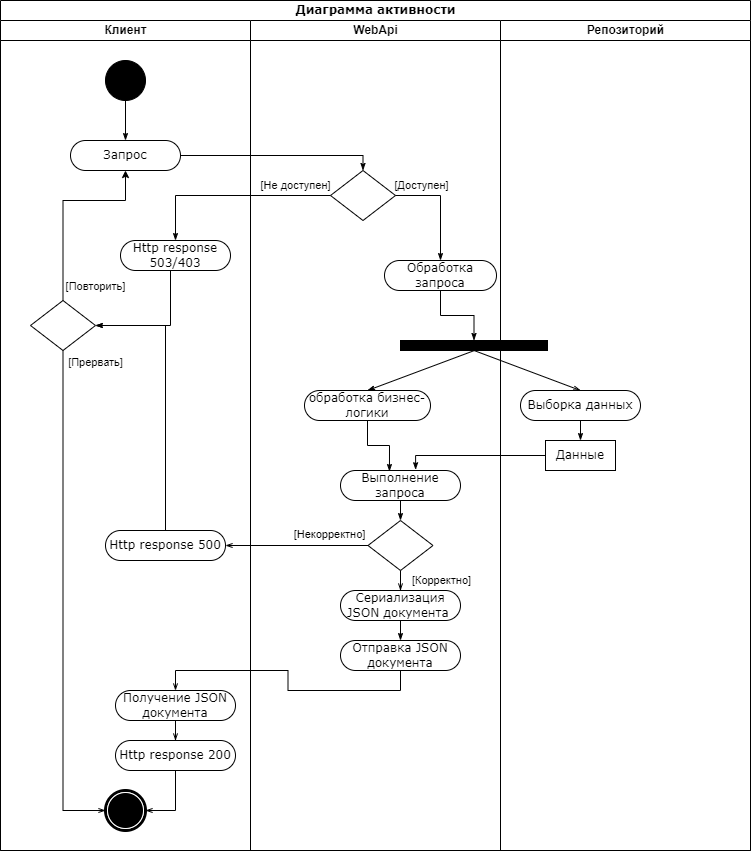


Рис. \* диаграмма активности

Для обособленной реализации логики и данных, Web Api построена по принципу чистой архитектуры (рис \*), что позволяет стандартизировать, обезопасить и ускорить выполнение реализованного функционала.



Рис. \* Чистая архитектура

Разработка производилась от внутренних уровней к внешним (Рис. \* Структура решения). Уровень Domain содержит модели данных, соответствующие сущностям в базе данных. Уровень Application содержит интерфейсы, бизнес-логику и Dependency Injection, для конфигурации ASP Net.Core API.

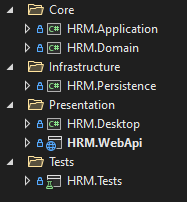


Рис. \* Структура решения

Далее идет уровень Persistence, который позволяет осуществлять работу с базой данных. Так, как он находится на ступень выше, то может реализовывать интерфейсы уровня Application. Для работы с БД используется Entity Framework 6 (EF), однако в данном проекте, некоторая часть бизнес-логики должна существовать непосредственно на уровне данных. Таким образом было принято решение следовать паттерну IRepository. Данный принцип не только помогает соблюдать подход чистой архитектуры, но и отличается в лучшую сторону в вопросах миграции с одной платформы на другую: к примеру, изначально разработка функционала уровня работы с данными производится на EF 6 под MS SQL, но внезапно возникает необходимость переноса базы данных в Postgre SQL и перестройки инфраструктуры для использования Dapper. В случае прямой разработки, этот процесс может растянуться на месяцы и даже годы, однако, с использованием IRepository, это можно сделать за 5 минут.

Репозиторий можно разделить на 3 уровня. Базой для каждого репозитория служит CRUD – базовый функционал создания, чтения, обновления и удаления данных. Следующий уровень наследует предыдущий и дополнительно реализует необходимый функционал конкретной модели (сущности), к примеру вывод списка не авторизованных в системе сотрудников. Верхний уровень представляет собой unit of work, содержащий все репозитории 2го уровня, методы доступа к ним и метод для сохранения всех изменений.

Для обеспечения минимальной нагрузки на систему, а так же с целью предотвращения зацикливания вызова данных, была разработана модель загрузки данных (см. рис. \*)

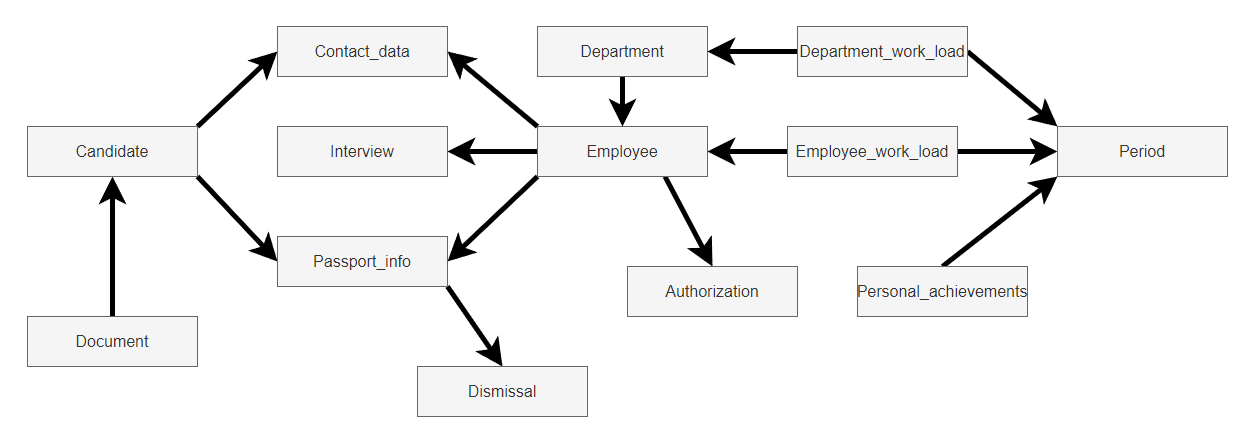


Рис.\* Модель загрузки данных

На данной модели прямоугольниками представлены сущности базы данных, а стрелочками зависимости загрузки. Для примера возьмем Документы кандидата: когда поступает запрос, на выборку из базы данных документа кандидата, вместе с ним, для дальнейшей обработки загружаются только кандидат, его контактные и паспортные данные. Модель разработана с расчетом на производительность и возможности дальнейшей реализации необходимого функционала.

Уровень Presentation представлен двумя проектами, первый: ASP.Net Core WebApi, здесь подключаются Dependency Injections из Application и Persistence уровней благодаря этому WebApi понимает, какую базу данных ему следует использовать, какие валидаторы и репозитории будут обрабатывать данные.

В этом проекте расположены MVC контроллеры, каждый контроллер реализует один или несколько Http методов: HttpGet, HttpPost, HttpPut, HttpDelete и другие.

В соответствии с их названиями, метод HttpGet вызывается автоматически при переходе по URL адресу контроллера и выводит тот набор данных (в виде JSON документа), который прописан в теле метода в контроллере.

Метод HttpPost вызывается при отправке формы (в виде JSON документа) на URL адрес контроллера с целью создания экземпляра какого-либо элемента.

Метод HttpPut вызывается при отправке формы (в виде JSON документа) при обращении к конкретному экземпляру какого-либо элемента контроллера с целью его изменения.

Метод HttpDelete вызывается при обращении к конкретному экземпляру какого-либо элемента с целью удалить его.

Подобное разделение называется Restful API, среди его достоинств можно отметить возможности по масштабированию и работе в условиях ограниченной пропускной способности.

Второй проект на этом уровне представляет собой один из вариантов клиентского приложения, в данном конкретном случае оно является приложением для ПК на базе Windows 8 и выше.

Клиентское приложение выполнено посредством технологии Windows Presentation Platform (WPF) .NET Framework 4.7.2 . За общение с WebApi отвечает 2 библиотеки: System.Net.Http и Newtonsoft.Json. Первая библиотека отвечает за получение и отправку Json документов на сервер WebApi, а вторая библиотека преобразует отправляемые запросы в Json документы и получаемые Json документы в результат.

Теперь, когда мы рассмотрели структуру проекта, остановимся по подробнее на некоторых модулях, обеспечивающих выполнение бизнес-логики.

Начнем с модуля распределения плановых рабочих часов на сотрудников по отделам. Для его реализации было написано 2 алгоритма:

1. Алгоритм статического распределения часов
2. Алгоритм динамического распределения часов

Первый алгоритм предполагает, что в систему будет внесено число, соответствующее общей рабочей нагрузке на компанию. При этом распределение будет производиться в первую очередь по сотрудникам, на основе этих данных будет произведен расчет нагрузки на отделы.

Второй алгоритм в качестве входных данных принимает список отделов и соответствующих им рабочих нагрузок. После назначения рабочей нагрузки отделам, происходит ее разделение на сотрудников. После прохождения всех этапов, выполняется перерасчет рабочих часов на отделах (в случае если введенные числа было невозможно поровну разделить между сотрудниками отдела), далее все суммируется и формирует общую нагрузку на компанию.

Модуль формирования дополнительных соглашений берет за основу шаблон, загружаемый в систему при ее настройке. Далее шаблон можно заменить на другой. Для уже существующего распределения рабочих часов можно сгенерировать набор дополнительных соглашений. После этого они хранятся в базе данных в виде массива битов до тех пор, пока не поступит запрос на загрузку. В зависимости от параметров запроса, файлы могут быть конвертированы в любой из представленных форматов. Если поступает запрос на загрузку 2х и более файлов, то они автоматически архивируются.

Модуль авторизации условно разделен на 3 алгоритма: регистрация, аутентификация и авторизация. Регистрация пользователя в системе происходит уже после прохождения всех этапов собеседования и зачисления в штат персонала. Для успешной регистрации пользователь должен указать в специальном поле свой идентификатор, который система автоматически присваивает всем сотрудникам. Узнать его он может у непосредственного начальства. Данная мера предосторожности исключает возможность появления нескольких учетных записей у одного сотрудника, возможность появления неопознанной учетной записи с доступом к внутренним данным компании и т.д. Аутентификация и авторизация выполнены стандартным набором средств: имя пользователя / логин и пароль.

Модуль приема и увольнения сотрудников интегрирован в структуру персонала. Ключевыми элементами данный структуры являются паспортные и контактные данные. Кандидат записывается в систему после того, как оставит по крайней мере часть из них для дальнейшего прохождения собеседования. Так же кандидат может присылать файлы (дипломы, грамоты, сертификаты и т.д.), они будут прикреплены к личному делу кандидата. По результатам собеседования претендент может быть нанят, в таком случае он предоставляет все необходимые персональные данные, либо не нанят. Информация о кандидате и попытке прохождения собеседования при любом исходе остаются в системе в качестве статистических данных.

При найме сотрудник получает свой идентификатор для авторизации в системе.

Процесс увольнения сотрудника выполняется в несколько этапов:

1. Заполнение соответствующей формы.
2. Назначение сотруднику часов на 2 последующие недели в случае, если причиной увольнения является собственное желание сотрудника.
3. Ожидание окончания рабочего периода.
4. Выплата заработной платы сотруднику в количестве, соответствующем отработанным часам.
5. Удаление данных об авторизации сотрудника из системы.
6. Фактическое увольнение сотрудника.

После увольнения сотрудника, в системе остается информация о его кандидатуре и прохождении им собеседований в качестве статистических данных.

Расчет заработной платы происходит автоматически в конце каждого месяца. Для определения заработной платы сотрудника используются следующие параметры:

1. Отработанные часы
2. Соответствие отработанных часов, запланированным
3. Дежурная ставка отдела
4. Временная ставка отдела
5. Премия

При этом за отработанные сверх плана часы начисляется именно премия, при заполнении менеджером отдела соответствующего пункта. Если сотрудник отработал меньше часов чем указано в его дополнительном соглашении, то с него так же может взиматься штраф, при заполнении менеджером отдела соответствующего пункта.

Дежурная либо временная ставка отдела указана в дополнительном соглашении и не меняется за указанный период времени. Временная ставка может отличатся от дежурной в случаях, если отдел показал хорошую результативность и высшее руководство решит временно повысить ставку.

## Разработка пользовательского интерфейса

Поскольку клиентское приложение создано на технологии WPF, пользовательский интерфейс будет написан на XAML.

При запуске приложения пользователь заходит на страницу авторизации (см. рис. \*), для продолжения использования приложения необходимо заполнить все поля и нажать кнопку «Войти», если сотрудник еще не зарегистрирован в системе, он может перейти на страницу регистрации (см. рис. \*).

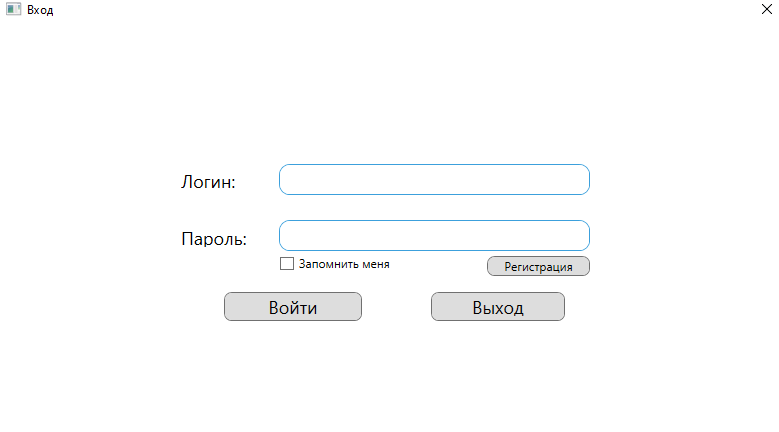


Рис \* Страница авторизации.

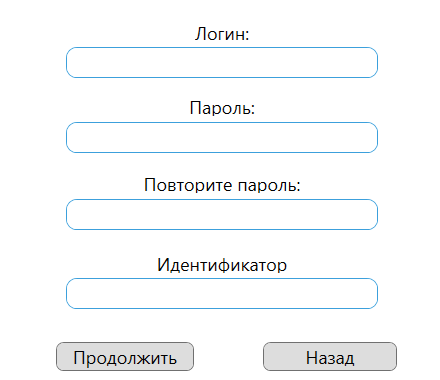


Рис. \* Страница регистрации

Для регистрации в системе, помимо логина и пароля, необходимо ввести ключ - идентификатор. Каждому сотруднику присвоен уникальный идентификатор в формате GUID, узнать свой идентификатор можно у менеджера своего отдела, либо у начальства.

После подтверждения личности, сотрудник переходит на главную страницу (см. рис. \*), далее навигация по приложению осуществляется 2мя способами: с помощью меню в верхней панели и кнопками на главной странице.

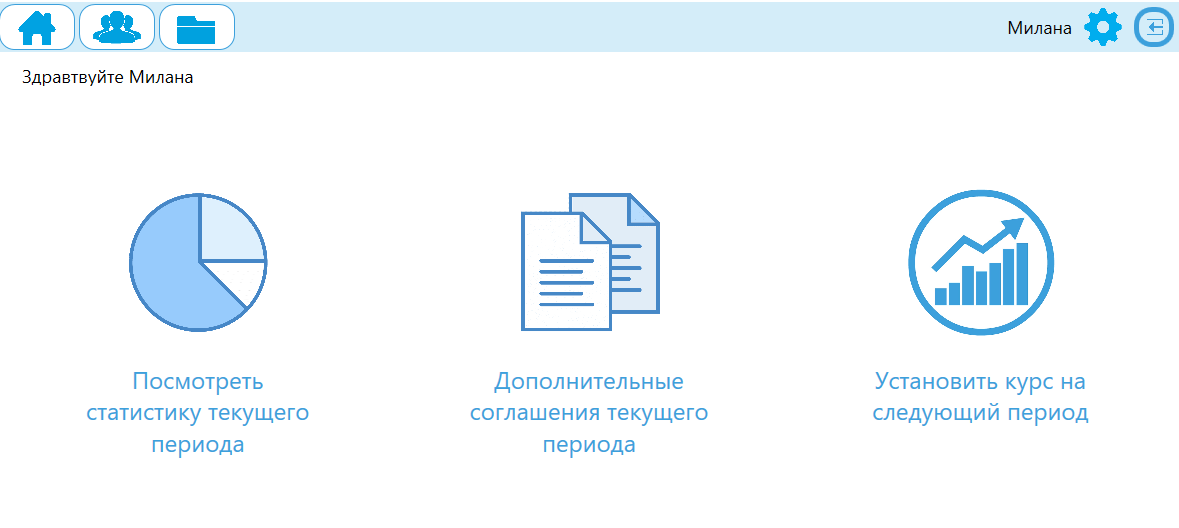


Рис. \* главная страница

Для менеджеров всех отделов и начальства предусмотрен следующий функционал: анализ распределения нагрузки по месяцам (см. рис. \*), загрузка (либо формирование) дополнительных соглашений для сотрудников по месяцам/отделам/конкретным сотрудникам (см. рис. \*), для начальства предусмотрен функционал изменения юридической информации предприятия (см. рис. \*), а так же распределения рабочей нагрузки на отделы и их сотрудников (см. рис. \*), менеджеры отделов могут составить отчет о личных достижениях своих сотрудников за данный период и назначить премию (см. рис. \*)

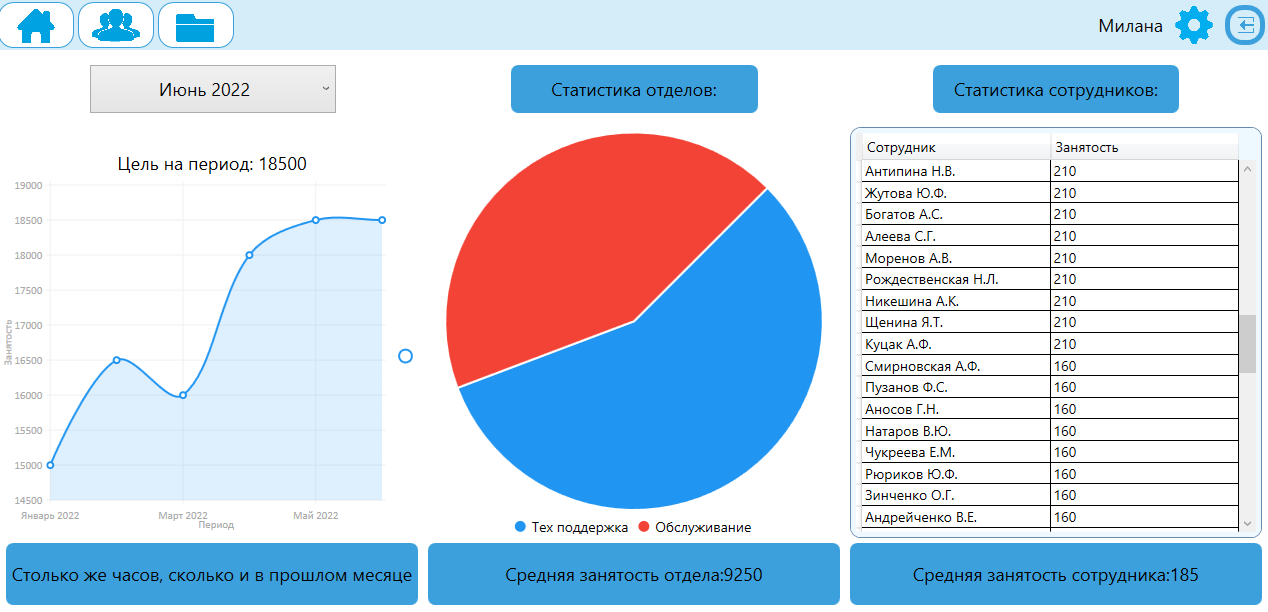


Рис. \* Страница статистики распределения рабочей нагрузки

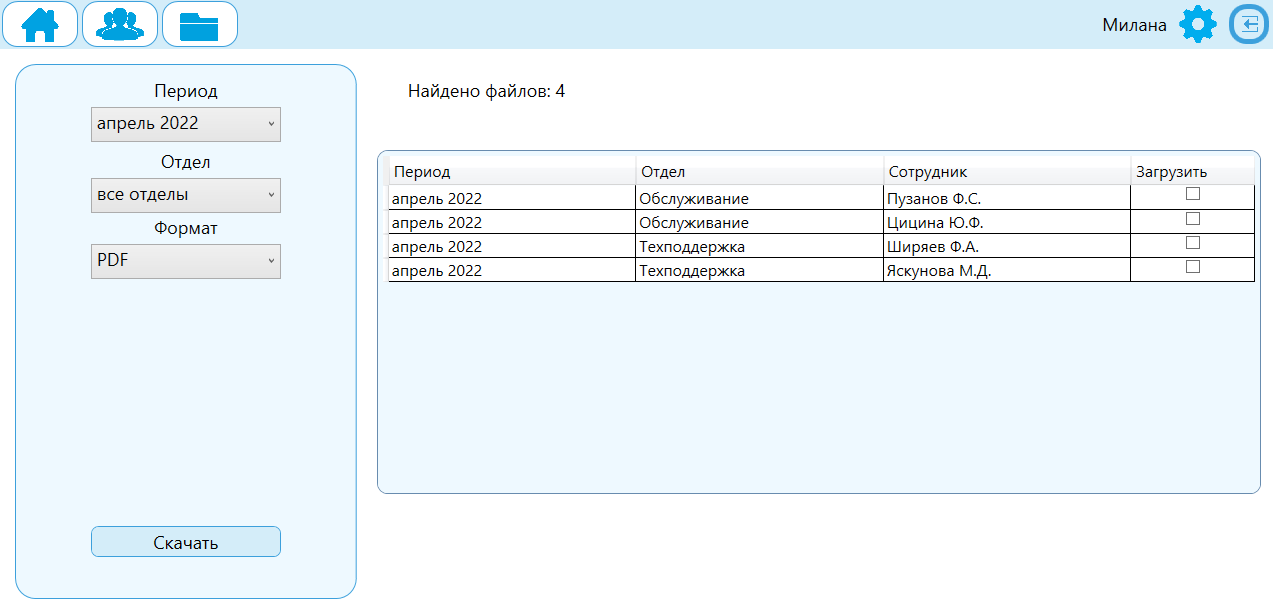


Рис. \* Страница загрузки доп. Соглашений

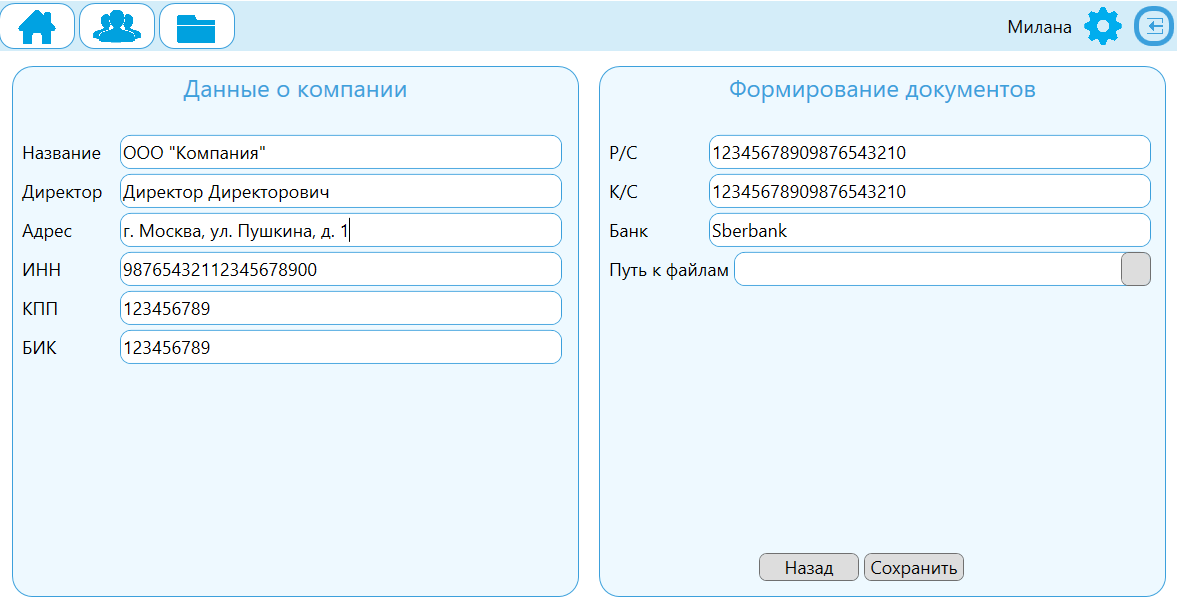


Рис. \* Страница юридической информации

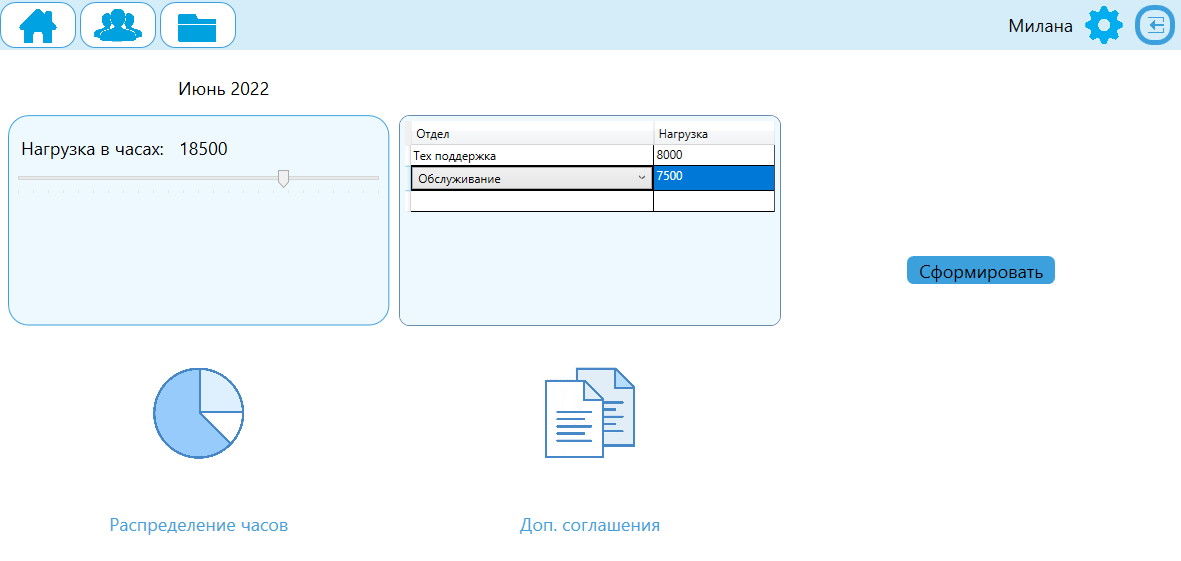


Рис. \* Страница распределения рабочей нагрузки

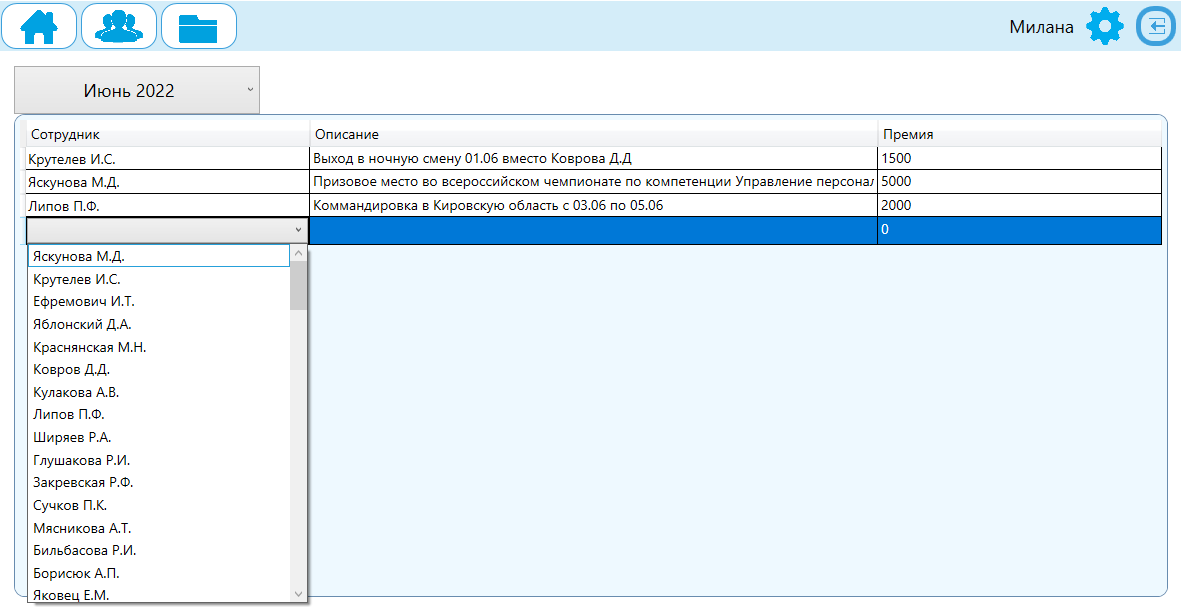


Рис. \* Страница заполнения отчета о достижениях сотрудника

Данный пример, наглядно иллюстрирует возможности по реализации клиентского приложения. Учитывая специфику конкретных сфер применения информационной системы, в качестве целевой платформы клиентского приложения могут выступать не только ПК, но и телефоны, планшеты и веб решения.

## Тестирование и отладка

Тестирование информационной системы можно разделить на 2 этапа: тестирование бизнес-логики и тестирование работы WebApi.

Для выполнения первого этапа тестирования был создан отдельный проект на xUnit, в нем реализована InMemory база данных, идентичная той, которую использует разрабатываемая система. Далее проведено заполнение базы тестовыми данными. После были написаны тесты для каждой критически важной функции. Среди них алгоритмы:

1. Статического распределения рабочих часов
2. Динамического распределения рабочих часов
3. Статического обновления распределения рабочих часов
4. Динамического обновления распределения рабочих часов
5. Генерации дополнительных соглашений
6. Прохождения собеседования
7. Увольнения сотрудника
8. Расчета заработной платы

Все произведенные тесты показали удовлетворительный результат (см. рис. \*)

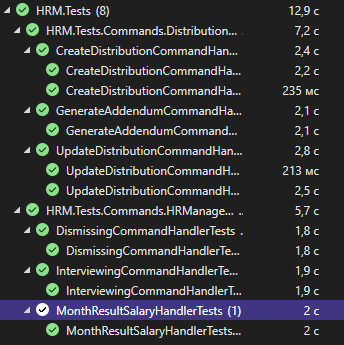


Рис. \* Результаты тестирования бизнес-логики

Тестирование работы WebApi реализовано с помощью Swagger и его визуального интерфейса (см. рис. \*)

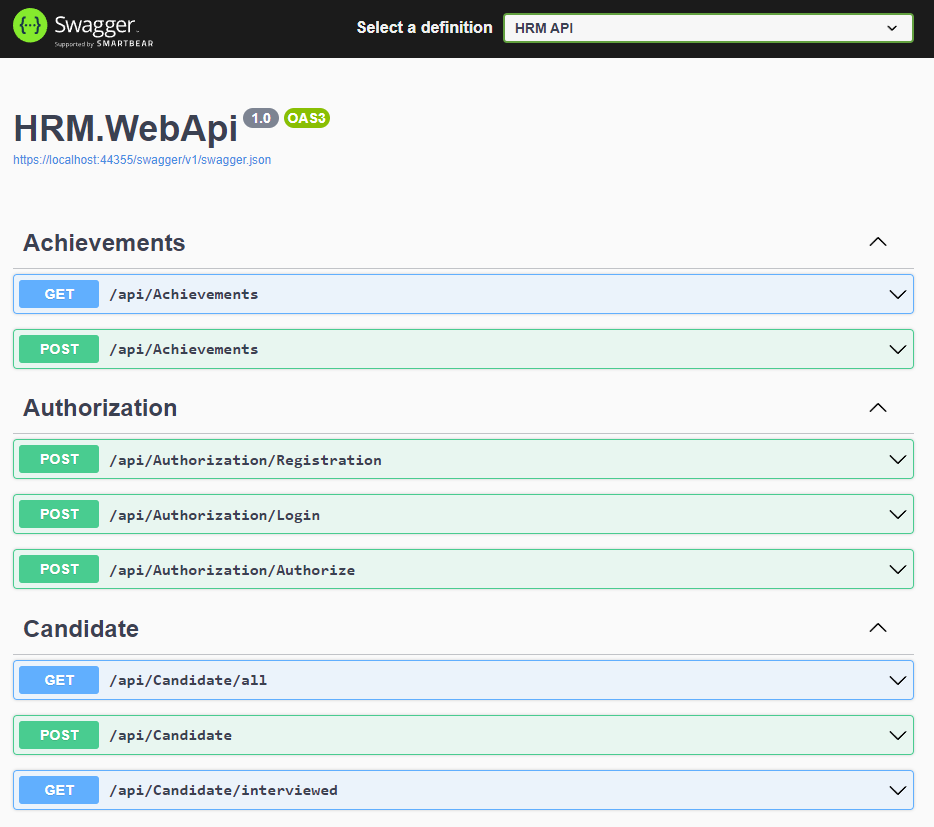


Рис. \* Визуальный интерфейс Swagger

Преимущества использования Swagger – это возможность редактировать JSON документы перед отправкой, а также всевозможная техническая информация при получении ответа (см. рис. \*, рис. \*).

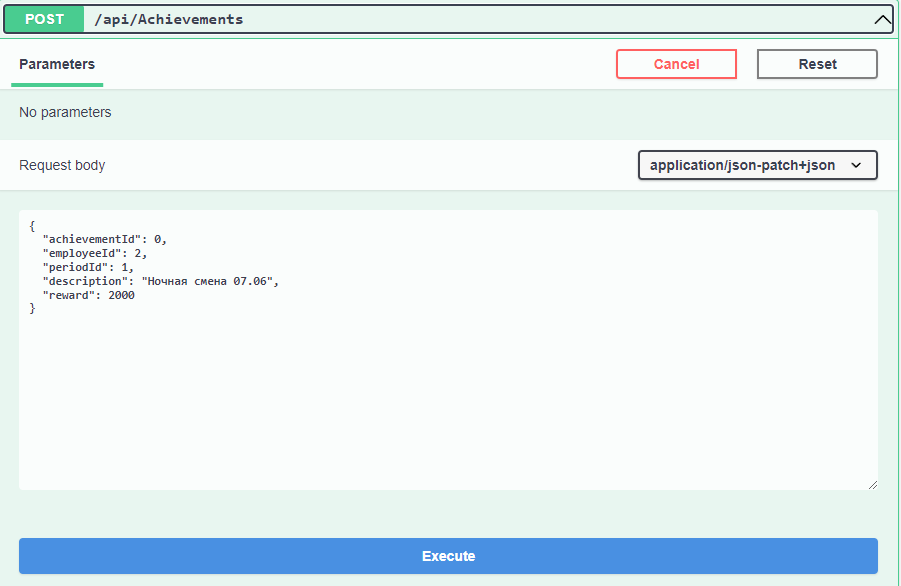


Рис. \* Пример создания запроса в интерфейсе Swagger

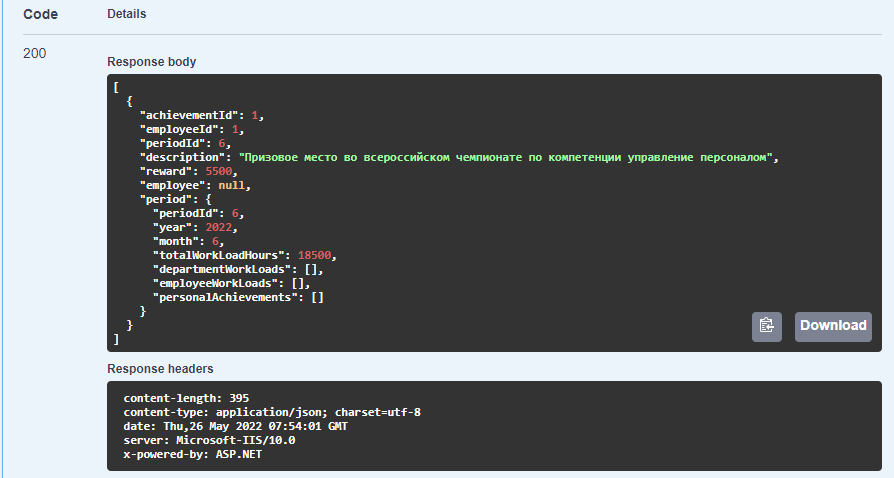


Рис. \* Пример полученного результата запроса в интерфейсе Swagger

Тестирование посредством Swagger использовалось на протяжении всей работы и помогло выявить и устранить множество дефектов.

# Заключение