Система расчёта научного рейтинга

Военного факультета

*Руководство разработчика*

# Общие сведения

Git репозиторий: <https://github.com/v-zubritsky/MilitaryFaculty> (Fork and work)

Рекомендуется использовать Reshrper.

# Проект расширений (Common)

Цель: Хранение методов расширений для всего приложения

Используемые библиотеки:

Описание и сведения:

Содержит методы расширений используемые во всем приложении.

# Проект обращения к данным (Data)

Цель: Работа с базой данных

Используемые библиотеки: EntityFramework

Ключевые классы:

**EntityContext:** Наследует класс DbContext.

**\*Configuration:** Класс конфигурации code first генерации базы данных. Наслудует класс EntityTypeConfiguration.

**Repository:** Generic класс. Реализует интерфейс IRepository. Предоставляет доступ к определенному DbSet из EntityContext.

Описание и сведения:

Проект служит для организации доступа к данным.

# Проект домена данных (Domain)

Цель: Предоставление сущностей, необходимых для работы приложения

Используемые библиотеки:

Ключевые классы:

**UniqueEntity:** Реализует интерфейс IUniqueEntity. Исходные класс для всех сущностей в приложении.

Описание и сведения:

Проект содежит все сущности используемые в приложении.

# Проект приложения (Application)

Цель:

Используемые библиотеки:

Ключевые классы:

Описание и сведения:

# Проект … (Presentation)

Цель:

Используемые библиотеки:

Ключевые классы:

Описание и сведения:

# Проект генерации отчета (Reporting)

Цель: Генерация отчета по имеющимся данным

Используемые библиотеки: ELW.Library.Math, EPPlus

Ключевые классы:

**DataProvider:** Абстрактный класс, реализует интерфейс *IDataProvider*, наследники которого содержат методы с атрибутами (каждый атрибут описывает, какую из переменных в формулах вычисляет данный метод). Также наследники реализуют методы ограничения (данные, необходимые для преподавателя, кафедры). Реализует несколько общих методов для вычисления значений.

**DataProviderContainer:** Агрегирует классы, которые наследованы от DataProvider. Служит для установки ограничений информации по типу заданной сущности (научный сотрудник, кафедра) и по интервалу времени, в которое были выпущены научные работы.

**ReportDataProvider**: Агрегирует классы, которые наследованы от DataProvider. Служит для подсчета значений по имени заданного атрибута (совпадает с переменной из формул).

**FormulaInfo:** Содержит информацию по формуле в текстовом виде.

**FormulaProvider:** Реализует интерфейс IFormulaProvider. Десериализует список формул. В итоге содержит их в виде коллекции FormulaInfo**.**

**Characteristic:** Предназначен для вычисления текстовых формул используя классы FormulaInfo и ReportDataProvider. Использует библиотеку ELW.Library.Math.

**ReportTableProvider:** Реализует интерфейс IReportTableProvider. Десериализует структуру таблиц отчета. В итоге содержит их в виде коллекции XReportTable.

**Report:** Объект отчета. Структура: коллекция ReportTable -> коллекция ReportGroup -> коллекция ReportRow -> коллекция Results. Для инициализации требует: ReportTableProvider, FormulaProvider, ReportDataProvider. И в соответствии с исходными данными при построении инициализируется итоговыми значениями. Есть возможность объединения двух объектов отчета с одинаковой структуры командой Unify (для генерации отчета для нескольких сущностей).

**ReportGenerator:** Создает объект отчета. Предоставляет объекту отчета необходимые входные данные в зависимости от типа сущности, относительно которой он генерируется, а также интервала времени, за который учитываются научные работы.

**ExcelReportingService:** Реализует интерфейс IExcelReportingService. Предназначен для генерации отчета в формате .xslx относительно объекта отчета. Использует библиотеку EPPlus.

**ExcelWriter:** Содержит методы для построения отдельных блоков отчета, их стили. Использует библиотеку EPPlus.

Описание и сведения:

Проект содержит инструменты для обработки файла с формулами, генерации объекта отчета, а также генерации .xlsx файлов.

Формулы для подсчета рейтинга хранятся в xml файле, рядом с ними в xml файлах хранится структура отчета для каждой сущности, по которым возможно построение отчета.

Домен для десериализации xml файлов находится в папке Structure/XmlDomain.

Десериализация проходит во время инициализации провайдеров.

# Дополнительные сведения

Порядок действий для добавления новой сущности:

* Описать сущность в Domain проекте, наследовать от UniqueEntity
* Создать \*Configuration класс в Data проекте
* Добавить в EntityContext
* \*Добавление на форму\*
* Добавить в xml файл формул новые формулы
* Добавить id новых формул в xml файл структуры таблицы отчета
* Создать \*DataProvider класс в Reporting проекте, наследовать от DataProvider, описать