**Документация**

База данных состоит из 5 таблиц:

* ActivityType;
* Company;
* Employee;
* OrganizationalType;
* Position;

И хранимых процедур:

* spAddActivityType (добавляет запись в таблицу ActivityType);
* spAddCompany (добавляет запись в таблицу Company);
* spAddEmployee (добавляет запись в таблицу Employee);
* spAddOrganizationalType (добавляет запись в таблицу OrganizationalType);
* spAddPosition (добавляет запись в таблицу Position);
* spDeleteActivityType (удаляет по id запись из таблицы ActivityType);
* spDeleteCompany (удаляет по id запись из таблицы Company);
* spDeleteEmployee (удаляет по id запись из таблицы Employee);
* spDeleteOrganizationalType (удаляет по id запись из таблицы OrganizationalType);
* spDeletePosition (удаляет по id запись из таблицы Position);
* spGetActivityType (возвращает запись по id из таблицы ActivityType);
* spGetCompany (возвращает запись по id из таблицы Company);
* spGetEmployee (возвращает запись по id из таблицы Employee);
* spGetOrganizationalType (возвращает запись по id из таблицы OrganizationalType);
* spGetPosition (возвращает запись по id из таблицы Position);
* spGetAllActivityTypes (возвращает все записи из таблицы ActivityType);
* spGetAllCompanies (возвращает все записи из таблицы Companies);
* spGetAllEmployees (возвращает все записи из таблицы Employees);
* spGetAllOrganizationalTypes (возвращает все записи из таблицы OrganizationalTypes);
* spGetAllPositions (возвращает все записи из таблицы Positions);
* spIsActivityTypeExist (проверяет существование записи в бд по id в таблице ActivityType);
* spIsCompanyExist (проверяет существование записи в бд по id в таблице Company);
* spIsEmployeeExist (проверяет существование записи в бд по id в таблице Employee);
* spIsOrganizationalTypeExist (проверяет существование записи в бд по id в таблице OrganizationalType);
* spIsPositionExist (проверяет существование записи в бд по id в таблице Position);
* spUpdateActivityType (обновляет запись по id из таблицы ActivityType);
* spUpdateCompany (обновляет запись по id из таблицы Company);
* spUpdate Employee (обновляет запись по id из таблицы Employee);
* spUpdate OrganizationalType (обновляет запись по id из таблицы OrganizationalType);
* spUpdate Position (обновляет запись по id из таблицы Position);

В приложении в папке View хранятся представления ActivityType, Companies, Employees, OrganizationalTypes, Positions, Home, Shared. В каждой из этих папок, кроме Home и Shared есть 3 представления:

* Create;
* Update;
* Название папки в множественном числе;

В папке Models хранятся классы сущностей:

* Company;
* Employee;
* CommonStructure(класс-родитель ActivityType, OrganizationalType, Position);
* ActivityType;
* OrganizationalType;
* Position;
* ErrorViewModel (создался автоматически);

В классе CommonStructure определены методы:

* SerializeToString, который сериализует экземпляр класса в строку с параметрами, отделенными между собой «~»;
* Desirialize, который десиреализует строку в экземпляр класса CommonStructure и возвращает его;

В классах OrganizationalType, Position, OrganizationalType реализованы методы DesirializeTo(Название класса), который принимает строку с параметрами и возвращает экземпляр класса.

В классах Company и Employee определены методы:

* SerializeToString, который сериализует экземпляр класса в строку с параметрами, отделенными между собой «~». Он отличается количеством параметров от метода SerializeToString класса CommonStructure;
* DesirializeToCompany (или Employee), который десиреализует строку в экземпляр класса Company (или Employee) и возвращает его;

В папке DataAccessLayers хранятся классы, которые взаимодействуют с бд:

* ActivityTypeDataAccessLayer;
* CompanyDataAccessLayer;
* EmployeeDataAccessLayer;
* OrganizationalTypeDataAccessLayer;
* PositionDataAccessLayer;

Все эти классы наследуются от класса DataAccessLayer.

В нем определены следующие методы:

* GetParticularData(int id, string storedProcedure), который получает id объекта и название хранимой процедуры возвращает объект из бд типа CommonStructure.
* ToActivityType(CommonStructure structure), который преобразует экземпляр класса CommonStructure в экземпляр класса ActivityType. Для OrganizationalType и Position определены такие же методы.
* ToActivityType(List<CommonStructure> structures), перегруженная версия метода ToActivityType. Для OrganizationalType и Position определены такие же методы.
* Delete(int id, string storedProcedure),который удаляет запись из бд экземпляра класса, наследованного от CommonStructure.
* Add(string value, string storedProcedure), который добавляет запись в бд экземпляра класса, наследованного от CommonStructure.
* Update(CommonStructure entity, string storedProcedure), который обновляет запись в бд экземпляра класса, наследованного от CommonStructure.
* GetAll(string storedProcedure), который возвращает все записи экземпляра класса, наследованного от CommonStructure.
* IsExist(int id, string storedProcedure), который проверяет, есть ли такой объект в бд.

И свойство ConnectionString, в котором хранится строка подключения к бд.

В папке Controllers хранятся классы, которые передают данные из классов, взаимодействующих с бд, в представления:

* ActivityTypeController;
* CompanyController;
* EmployeeController;
* HomeController;
* OrganizationalTypeController;
* PositionController;

В них реализованы методы:

* [HttpGet]Create, который возвращает представление для создания нового объекта;
* [HttpPost]Create, который принимает из представления параметры и создает новый объект;
* [HttpGet]Update, который возвращает представление для обновления объекта;
* [HttpPost]Update, который принимает из представления параметры и обновляет объект;
* [HttpGet]Delete, который принимает из представления параметры и удаляет объект;
* [HttpGet]НазваниеКлассаВоМножественномЧисле, который возвращает представление для отображения всех записей бд определенного класса;