MonsterCradTradingGame – Protocol

Link: https://github.com/Davidhoechtl/SWEN\_Semesterprojekt\_2022.git

# Design

The solution is divided into 4 separate projects. The MonsterTradingCardGame\_Hoechtl.csproj, MTCG.DAL, MTCG.Logic and MTCG.Tests.

## Server Architecture

Ein Bild, das Text, Screenshot, festlegen, mehrere enthält.

Automatisch generierte Beschreibung

The MonsterTradingCardGame\_Hoechtl.csproj is the Server of the Application. It Builds the complete Application via Autofac and starts the HttpListener which listens for user requests.

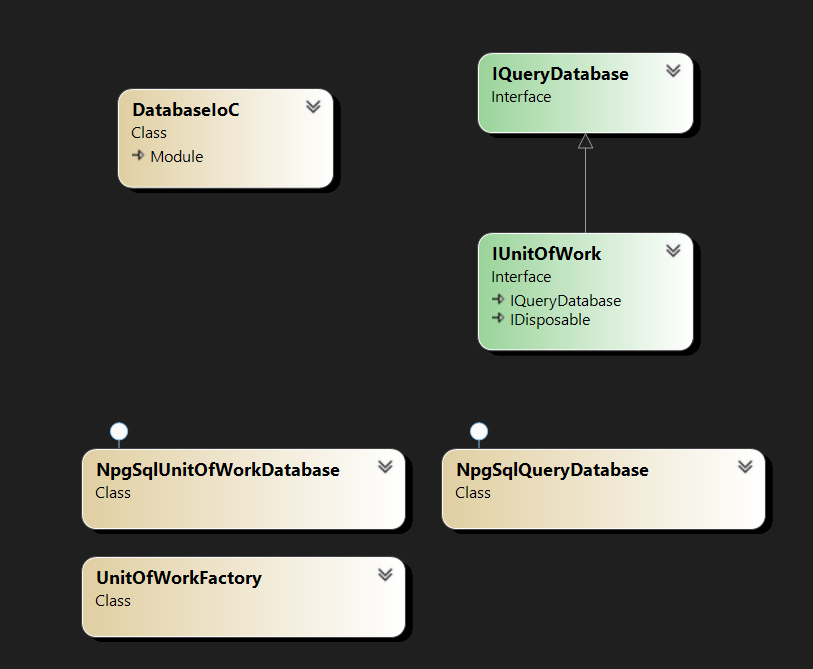
**IHandler** is the interface that is used to create Endpoints of the server. The request is broken down into its individual parts. From the url the module name is extracted and matched with all handler names.

**HandlerMethodResolver** is used to calculate the correct method of the found handler. This happens using HttpMethodAttributes and Reflexion. There can only be two Parameters.

* SessionContext -> Tells which user sent the request,
* object requestPayload -> sent JSON content.

**SessionContext** is used to handle token-based security. Here every user\_id is mapped with a generated GUID, which becomes the API key the user can user after login.

## Data Access Layer

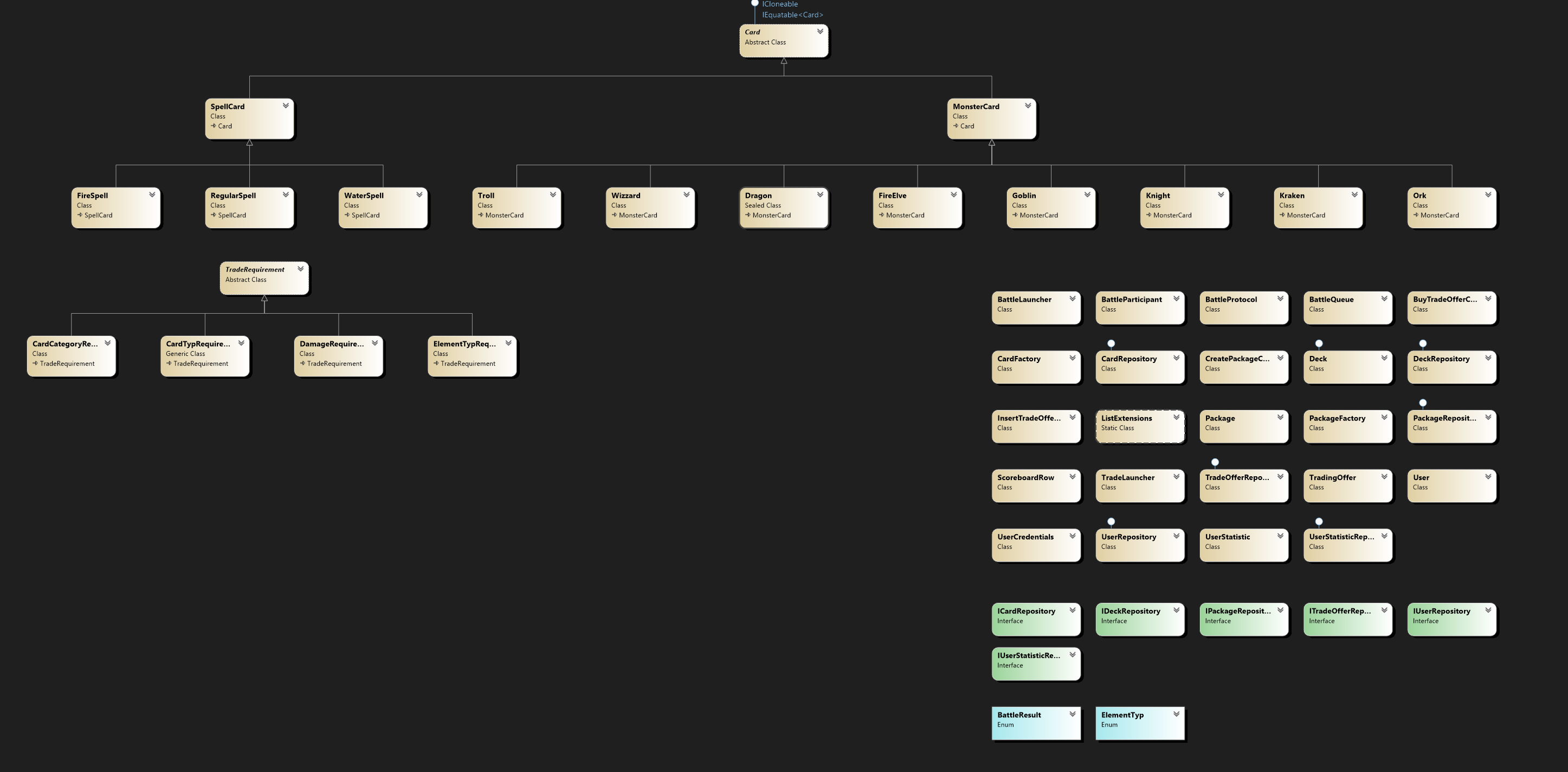


The project uses a Postgres SQL database. Npgsql Extension is used to communicate with it using C#.  
The Abstract Interface for the database implementation is called **IQueryDatabase**. Here methods like GetItem and GetItems are defined in a generic pattern.

**IUnitOfWork** is created via **UnitOfWorkFactory** and is used for transaction-based data access.

NpgSqlParameter and prepared Statements are used for avoiding SQL injection.

## Game Architecture



# Lessons Learned

# Unit-Tests Decisions

# Unique Features

# Tracked Time