Swen Protokoll

Paul Panosch (if22b108)
GitHub-Repo: MonsterCardGame

Design Beschreibung

Aufbau der Endpunkte

Das Design der Software lehnt sich stark an ASP.Net an und benutzt ebenfalls Reflections, um die einzelnen Endpunkte, Middlewares und Controllers zu erstellen. Controller, welche eine Klasse sind mit dem Attribut „ApiController“, beinhalten alle Endpunkte.

Ein Endpunkt ist eine Methode innerhalb eines Controllers, geschmückt mit einem der vier Attribute: „HttpGet“, „HttpPost“, „HttpPut“, „HttpDelete“. Diese Attribute bestimmen, um welchen http Anfragetyp es sich bei dem Endpunkt handelt. Dazu kann ein „ApplyMiddleware“ hinzugefügt werden. Dies bezweckt das Ausführen einer Middelware bevor die Methode des Endpunktes ausgeführt wird.

Eine Middleware ist eine Klasse mit dem Attribut „ApiMiddleware“ und einer Methode namens „Invoke“. Middlewares dienen zum Manipulieren des HttpRequestObjects, bevor es zu einem Endpunkt gelangt.

Der Pfad eines Endpunktes wird über das Controllerattribut „ApiController“ und eines der vier Endpunktattribute gesetzt. Der Pfad innerhalb des „ApiController“ Attributs ist der Basispfad. Die Endpunktattribute können diesen Basispfad erweitern und ebenfalls Pfadvariablen enthalten. (Pfadvariable: „/das/ist/mein/pfad/{id}“)

Vor dem Aufruf einer Endpunktmethode wird mittels Reflections eruiert, ob die Methode Parameter besitzt, welche dynamisch hineingeladen gehören. Dies ermöglicht das einfache Parsen von Daten aus dem RequestObject.

Authentifizierung

Die Authentifizierung implementiert eine eigene Version des Json-Web-Tokens, den PJWT (Pauls Json Web Token). Die Handhabung des PJWT wird durch eine Middleware gelöst. Alle Endpunkte, welche nur als authentifizierter User angesprochen werden dürfen, müssen diese Middleware vorher einbinden.

Der PJWT ist ein String, der aus drei Bestandteilen besteht: Header (besitzt die „Time To Live“ des Tokens), Content (Kann alles sein, in diesem Fall die UserId), Checksum (Schützt vor Client-Manipulation). Die Bestandteile sind in Base64 kodiert und werden in dem String durch einen Punkt getrennt.

Der PJWT muss, als „Bearer“ Token über die „Authorization“ Header übermittelt werden.

Battle

Es ist ein Warteraum-System implementiert, welches darauf wartet, dass zwei Benutzer einem Battle beitreten wollen. Wenn ein Warteraum voll ist, wird das Battle gestartet. Nach dem erfolgreichen Ende eines Battles wird an die jeweiligen Threads der HttpRequest das Battlelog zurückgeliefert.

Gewonnene Erkenntnisse

- Umgang mit Reflections.
- Bauen eines eigenen Http-Servers über einen TCP-Socket.
- Unit Testing mit FakeIt.
- Umgang mit ADO.Net
- Mehrere unabhängige Threads dazu zu bringen auf das Ende eines Ereignisses zu warten.
- Tieferes Verständnis für Json-Web-Tokens.

Unit-Test Entscheidungen

Verwendung von FakeItEasy über Moq, wegen des E-Mail scraping von Moq. Repositories, Services und Utilities müssen über den Constructor einer Klasse manuell gesetzt werden können. Ermöglicht das Isolieren von Funktionen in diesen Klassen, da die Repositories, Services und Utilities dadurch auch Dummies von FakeItEasy sein können.

Unique Feature

Der Casino-Coinflip-Endpunkt gibt Spielern die Möglichkeit ihren Einsatz von Münzen durch eine 50/50 Chance zu verdreifachen oder den Einsatz zu verlieren.

Aufwand an Zeit

Datum	Zeit
14.09.2023	10m
19.09.2023	4h
30.09.2023	2h
01.10.2023	1h 30m
10.10.2023	3h
11.10.2023	2h 30m
14.10.2023	3h
28.10.2023	5h 30m
30.10.2023	6h
04.11.2023	4h 30m
20.11.2023	1h
05.12.2023	2h
28.12.2023	5h
29.12.2023	7h
30.12.2023	3h

Summe: 50h 10m