# Triangle analyzer

## Introduction

The solution was implemented using an adapted version of the onion architecture. The setup revolves around the key assembly TriangleAnalyzer.Service. This assembly defines the business scope and logic of the application as well as the interfaces that the defines the infrastructure needed.

The idea is that with all other assemblies referencing only inwards in the onion a number of tasks become easy in maintaining the application.

These include testing, replacing and extending pieces of functionality etc.

Disclaimer: The application was implemented to showcase architecture and implementation details. The final steps to get the application up and running are left out of the implementation.

## The components

The solution consists of 7 assemblies. Here discussed in brief:

TriangleAnalyzer.Tests - Serves as a base of the testassemblies providing dependency injection capabilities.

TriangleAnalyzer.Service.Tests - Provides test coverage of the service assembly. Note that the test coverage is inadequate.

TriangleAnalyzer.Infrastructure.Tests - Provides test coverage of the infrastructure assembly. Note that the test coverage is inadequate.

TriangleAnalyzer.API - Exposes the functionality of the application.

TriangleAnalyzer.Batch - Handles offline processing of the analyzed triangles.

TriangleAnalyzer.Service - Defines the businesslogic of the application.

TriangleAnalyzer.Infrastructure - Defines the database access as well as other potential external dependancy.

## Use cases

Analyze triangle - The analyze method of the WEB API accept three integers representing the sides of a triangle. The values are passed on as an array.\* The analyze method of the service component accepts the array and validate the triangle. In case the validation fails http code 400 (Bad request) is passed back to the requester. In case the validation succeeds the triangle is analyzed to be either equilateral, isosceles or scalene. The triangle is then quickly offloaded to a processer queue to be included in the complete set of analysed triangles and then the result is passed back to the user.

Process incoming triangles - A simple batchjob (console application) is setup to process the triangles to make sure triangles only appear once in the complete list. The important point to make here is that the processing must be made to make the online (web api) run as fast as possible. The implementation could look differently based on detailed requirements.

Get all triagles - As the triangles are preprocessed the list is equal to the one stored in the database.

\*An assumption is made that a triangle is the same whether it is inverted or not.