# Odyssey Gaming Candidate Assessment Documentation

## Initial Setup

Initial setup is simple – Visual Studio 2022 is required as well as .NET 6. Simply pull down the repository, open the solution and run it. When first building the project, you will need to approve an SSL certificate to view the Swagger docs that are opened. This will allow for extensive testing of the API endpoint.

## Assumptions and/or Design Choices

The assumption was made that the input would be between 0 and the maximum value allowed by the decimal type in C# (79,228,162,514,264,337,593,543,950,335).

The assumption was also made that the input is required, as this was not specified in the documentation.

It is also assumed that we are using standard rounding practices in C# (i.e., Math.Round)

The code was written to be as concise as possible while still maintaining readability – avoiding unnecessary code duplication and over-complicating simple tasks.

## Testing and validation

The quickest way to test and validate the service is to build the solution and enter some values in the Swagger docs – as this will make it clear when an invalid input is provided, as well as provide the output when valid input is provided.

To see the raw output, such as error responses, hit the endpoint /api/DecimalToString/{input} from either the browser or a CURL request.

## Bugs/Issues

One minor issue is regarding the minimum value for input. It is set at 0.01, even though the minimum input before rounding could technically be much lower.

## Security/Longevity

Obviously this is incredibly insecure, as there is no form of authentication before this endpoint can be accessed. If this were deployed in a production environment, authentication would be a necessity (especially assuming the data involved is regarding cheques).

For longevity, I would probably migrate this to asynchronous code, as if this were to be deployed at any sort of scale it would struggle to keep up, given the fact it is synchronous.