Quantified Student

Technical Documentation—API Design

Kane Petra (456964)

Koen Janssen (451394)



# 

# Abstract

# This technical

# 

# Version History

| **Version** | **Author** | **Date** | **Description** | **Reviewers** |
| --- | --- | --- | --- | --- |
| 0.1 | K. Petra | 10.05.2022 |  | K. Janssen |
| 0.2 | K. Janssen | 12.05.2022 | Added argumentation of the framework choice.  Added updated version of Interface section. |  |

# 

# Table of Contents

[**Version History**](#_8mkqauacshyo) **2**

[**Table of Contents**](#_sjm8gpclnyo1) **2**

[**Preface**](#_1248eu7394b5) **3**

[Documents](#_z618b2q0o7v1) 3

[Standards](#_1szl4n3q7wgq) 3

[**System Architecture**](#_xho1soja9s1t) **4**

[Architecture](#_nmvx5kibv1tu) 4

[Router](#_5rsvl7igzlld) 4

[Third Party Controller](#_zd8m0qhwotae) 5

[User Controller](#_4n72in75j4yx) 5

[Interface](#_ff82czjdo0ax) 5

[Data storage](#_g5o4ey51d5sp) 5

[**Data Model**](#_ple17qpv2sj4) **6**

[Database](#_ws26vrwts5b3) 6

[**Object Model**](#_1g53l5yqd9fw) **6**

[Objects](#_bkzhz7vkngsg) 6

[UserController](#_r6ddlrbxsy92) 6

[MoreClasses…](#_7g4b92w9ium) 6

[**System Functions**](#_y5xz2ubiuehg) **7**

# Preface

The Quantified Student (QS) API …

## Documents

These documents were used when creating this one and will be referenced later on.

* List
* Used
* Documents
* Here

## Standards

This document uses the following standards and expects the reader to understand them:

* Application Programming Interfaces (API)
* Entity Relationship Diagram (ERD)
* Context, Container, Components & Code model (C4 Model)

# 

# System Architecture

## Architecture

The QS API consists of multiple components that communicate with each other. An overview can be found below and a description under that.

## 

### Router

The main entry point of the API servers, this component will handle routing to the different controllers and does not contain any logic beyond this.

### Third Party Controller

The third party controller handles data aggregation and access to the data stored in the data warehouse. Endpoints require authentication before being usable, to authenticate see “User Controller” below.

### User Controller

The user controllers handles authentication using Canvas’s identity server, user preferences are stored locally on a PostgreSQL database.

## Interface

For interfacing with the API, we chose the JSON data format. This allows us to easily parse the returned data into objects. Further, according to Microsoft Docs, "**JSON** is probably the most common data format for web APIs" (APA ref here: https://docs.microsoft.com/en-us/azure/architecture/best-practices/api-design) Making it easy for other developers to use and interact with the API, as the use of JSON is so commonplace in REST APIs.

## Data storage

Dropped

The API itself will only store data in a PostgreSQL database and will not produce it anywhere else.

## Framework

The framework for the API that we decided on is ASP.NET. This choice is largely based on the Team's experience with ASP.NET, .NET and the C# programming language. As well as, these technologies are covered within the Fontys curriculum in the first year. For these reasons, we believe that it would be a good choice as the framework. This should enable everybody in the team to easily work on its other groups that will be working on Quantified student in future iterations.

## 

# Data Model

Link Canvas API

https://canvas.instructure.com/doc/api/

## Database

The API will only store data in a PostgreSQL database, the model will look like this:

# Object Model

Diagram here

ThirdPartyController: also research what data we want to save in our own API that is related to the date warehousing or RDBMS.

## 

## 

## Objects

### UserController

### MoreClasses…

# System Functions

## Post processing

# 

# Glossary

| *Term* | *Explanation* |
| --- | --- |
| *API* | *Application Programming Interface* |