T2A2: API Webserver

9/29/2024

84 Points Possible

Attempt 1

In Progress

NEXT UP: Submit assignment

Add comment

Unlimited Attempts Allowed

7/22/2024 to 10/4/2024

Details

T2A2: API WEBSERVER

Overview

Title	T2A2: API Webserver
ID	T2A2
Type	Assessment - Individual
Start Date	Week 7, Term 2
Due Date	Week 10, Term 2
Course	Diploma of Information Technology - Bootcamp Delivery Mode
Subject	CMP1001 - Principles of Information Systems PGM1003 - Database Design and Development
Weighting	CMP1001: 35% PGM1003: 50%
Marks	Total Marks: 84 • CMP1001: 42 • PGM1003: 42
Assessment Policies	• Written assessments are usually due on Sunday at 11:55pm on the week they are due. ***Please check the Assignments page in Canvas for the exact due date and time of each assessment event.*** • You will have a maximum of 5 days (including weekend days and public holidays) to submit if the due assessments are 'late'. • 5% points will be deducted for each day the assessment was late, up to a maximum of 5 days. After the 5 day period the assessment will need to be submitted to the academic teacher via their official AIT email and will be marked for feedback purposes, but will receive a grade of zero (0). • Refer to the latest version of the **Late Submission Policy** **(https://f.hubspotusercontent20.net/hubfs/493379/Assessment%20Late%20Submission%20Assessment%20Policy_09112020_V1.0.pdf)** online.

Introduction

Web servers can come in many shapes and contain different levels of complexity. At their core, they always involve server concepts such as routing, and handling the communication of data between users and a data storage medium.

To solidify your knowledge of core web server concepts and show your ability to work with web servers at a fundamental level, you should be able to write code to create a functioning web API server.

Brief

In order to demonstrate your understanding of fundamental programming concepts and database design, you are required to develop an API that works with a database, implementing functionality suitable for a realistic API concept.

Requirements

Requirements for this project are divided into two major parts,

1. Code
2. Documentation

The sections below contain more information about each of the major parts.

Documentation Requirements

Documentation for this project must be supplied as a single markdown file named README.md. This file should contain:

No.	Requirement
R1	Explain the problem that this app will solve, and explain how this app solves or addresses the problem.
R2	Describe the way tasks are allocated and tracked in your project.
R3	List and explain the third-party services, packages and dependencies used in this app.
R4	Explain the benefits and drawbacks of this app's underlying database system.
R5	Explain the features, purpose and functionalities of the object-relational mapping system (ORM) used in this app.
R6	Design an entity relationship diagram (ERD) for this app's database, and explain how the relations between the diagrammed models will aid the database design. This should focus on the database design BEFORE coding has begun, eg. during the project planning or design phase.
R7	Explain the implemented models and their relationships, including how the relationships aid the database implementation. This should focus on the database implementation AFTER coding has begun, eg. during the project development phase.
R8	Explain how to use this application's API endpoints. Each endpoint should be explained, including the following data for each endpoint: • HTTP verb • Path or route • Any required body or header data • Response

Design Requirements

- The web server must:
  - function as intended
  - store data in a persistent data storage medium (eg. a relational database)
  - appropriately validate & sanitise any data it interacts with
  - use appropriate HTTP web request verbs - following REST conventions - for various types of data manipulation
  - cover the full range of CRUD functionality for data within the database
- The database manipulated by the web server must accurately reflect the entity relationship diagram created for the Documentation Requirements.
- The database tables or documents must be normalised
- API endpoints must be documented in your readme
- Endpoint documentation should include
  - HTTP request verbs
  - Required data where applicable
  - Expected response data
  - Authentication methods where applicable

Code Requirements

- The web server must:
  - use appropriate functionalities or libraries from the relevant programming language in its construction
  - use appropriate model methods to query the database
  - catch errors and handle them gracefully
  - returns appropriate error codes and messages to malformed requests

- use appropriate functions or methods to sanitise & validate data
- use D.R.Y coding principles

- All queries to the database must be commented with an explanation of how they work and the data they are intended to retrieve

Deliverables

Following is a list of deliverables (which should cover all the above requirements) to include in your submission. The table also details how the information should be organised.

Deliverable	Description	Location (in Zip file)
`README.md`	General project documentation is to be compiled as a single markdown file named `README.md`.	**/** (Root folder of your zip file)
Resources	All files linked by the `README.md` file must be included in a folder named `docs`. All resources included in this folder must be in either `png`, `jpeg`, `pdf`, or markdown (`md`) format	**docs/**
Source Code	Source code for your entire project	**src/**

> Note: **All links to online material should have corresponding screenshots included in submission.**

Submission

- This project must be submitted via **Canvas**
- Your submission is to be a single zip file (organised as described in the *Deliverables* section above)

FILE

Follow the steps below to correctly prepare file(s) to submit,

1. Create a **directory** called,

```
{Fullname}_T2A2
```

> Where,
> - `{Fullname}` is to be replaced by your Fullname
> - T2A2 is this Assignment's ID.

2. **Organize** all the files for submission in this directory

> Follow the guidelines provided in the *Deliverables* section below

3. Create a **zip** file of this directory called,

```
{Fullname}_T2A2.zip
```

> A sample command that can be run in Mac OS X/Linux terminal to achieve creation of the zip file is as follows,
>
> ```
> zip -r {Fullname}_T2A2.zip {Fullname}_T2A2
> ```

4. Submit this zip file in Canvas

Example

> If your name is `Luke Skywalker`,
>
> 1. The directory you create will be called `LukeSkywalker_T2A2`

```
mkdir LukeSkywalker_T2A2
```

2. Organize all the files for submission in this directory
3. Create a zip file of this directory. The zip file will be named `LukeSkywalker_T2A2.zip`

```
zip -r LukeSkywalker_T2A2.zip LukeSkywalker_T2A2
```

4. Submit `LukeSkywalker_T2A2.zip` in Canvas

DEADLINE

You are responsible for submitting this assessment before the deadline date/time. Your assessment `submission-time` is set and logged by Canvas once your entire submission is uploaded and submitted in the system. This `submission-time` is used to verify on time submission or determine if you are liable for any late submission penalties.

Slow internet speeds, long submission/upload times, delay in uploading, etc are NOT grounds for special consideration (i.e. for waiving of any applicable late penalty).

> Remember for any valid special consideration request, relevant and full support documentation MUST be provided. The outcome of the special consideration request will be determined by Academic management and not your Educators.

Learning Outcomes and Criteria

Subject	Learning Outcome	Criterias
CMP1001 - Principles of Information Systems	CMP1001-1	CMP1001-1.2, CMP1001-1.3, CMP1001-1.4
	CMP1001-2	CMP1001-2.3, CMP1001-2.4
	CMP1001-6	CMP1001-6.2
	CMP1001-7	CMP1001-7.2
PGM1003 - Database Design and Development	PGM1003-2	PGM1003-2.1, PGM1003-2.2
	PGM1003-4	PGM1003-4.1, PGM1003-4.2
	PGM1003-5	PGM1003-5.2
	PGM1003-6	PGM1003-6.2
	PGM1003-7	PGM1003-7.3

> Refer to the Term Academic overview for further details

Marks

Marks and/or results for this Assessment (as released in Canvas) are only raw marks and may not necessarily reflect final grades on transcripts. Grades are only finalized after review by the Academic Board and applicable processing (moderation, etc).

> The `*` symbol in the *rubric* below indicates only part of the specified requirement is being assessed in the applicable criteria

---

Extraordinary Extensions

Students may apply for an Extraordinary Extension at any time during the course if they have been prevented from completing an assessment by the following:
* Serious Illness/Injury
* Personal trauma
* Pregnancy with medical complications
* Bereavement
* Major technical issues

Students must provide documentation and there is a form to use,

Coder Academy Extraordinary Extension Form (https://forms.gle/41WBQmtNF3vJKWyG6)

View Rubric

T2A2: API Web Server

Criteria	Ratings					Points
CMP1001-6.2: JUSTIFIES the purpose and goal of the developed application.	**6 to >5 pts** **HD** Provides a DETAILED explanation about the problem being solved by the developed application AND about how the app addresses the problem, and DOES use any objective references or statistics to support their answer.	**5 to >4.5 pts** **D** Provides a DETAILED explanation about the problem being solved by the developed application AND about how the app addresses the problem, but DOES NOT use any objective references or statistics to support their answer.	**4.5 to >4 pts** **C** Provides a DETAILED explanation about the problem being solved by the developed application OR about how the app addresses the problem, but not both - one is explained with brief or low detail.	**4 to >2.99 pts** **P** Provides a BRIEF OR LOW-DETAILED explanation about the problem being solved by the developed application, with a BRIEF OR LOW-DETAILED explanation as to how the app addresses the problem.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
CMP1001-2.3: DESCRIBES the way tasks are planned and tracked in the project.	**6 to >5 pts** **HD** Meets D, and includes proof of THOROUGH usage of specific task management tools THROUGH THE LENGTH OF THE PROJECT.	**5 to >4.5 pts** **D** Meets P, and includes reference to and proof of THOROUGH usage of specific task management tools.	**4.5 to >4 pts** **C** Meets P, and includes reference to and proof of BASIC usage of specific task management tools.	**4 to >2.99 pts** **P** Provides a BRIEF description about how tasks are planned and tracked in the project.	**2.99 to >0 pts** **F** Provides an INADEQUATE description of how tasks will be planned and tracked.	/ 6 pts
CMP1001-1.2: DESCRIBES the third party services, packages or dependencies that are used in the developed application.	**6 to >5 pts** **HD** The description provided is DETAILED, and the description details ALL of the services, packages or dependencies that are used in the developed application.	**5 to >4.5 pts** **D** The description provided is DETAILED, and the description details MOST of the services, packages or dependencies that are used in the developed application.	**4.5 to >4 pts** **C** The description provided is BRIEF OR LOW-DETAILED, and the description details ALL of the services, packages or dependencies that are used in the developed application.	**4 to >2.99 pts** **P** The description provided is BRIEF OR LOW-DETAILED, and the description details MOST of the services, packages or dependencies that are used in the developed application.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
CMP1001-2.4: IDENTIFY AND DESCRIBE the benefits and drawbacks of a chosen database system.	**6 to >5 pts** **HD** Meets D, and describes benefits AND drawbacks to a thorough level of detail.	**5 to >4.5 pts** **D** Identifies an appropriate database system and DESCRIBES SOME benefits and/or drawbacks to a THOROUGH level of detail.	**4.5 to >4 pts** **C** Identifies an appropriate database system and DESCRIBES SOME benefits and/or drawbacks to a BASIC level of detail.	**4 to >2.99 pts** **P** Identifies an appropriate database system and LISTS SOME benefits and/or drawbacks.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
CMP1001-1.3: EXPLAINS the features and functionalities of an object-relational mapping (ORM) system	**6 to >5 pts** **HD** Explains MULTIPLE features or functionalities of an ORM to a THOROUGH level of detail, supporting the explanation with AT LEAST ONE code example.	**5 to >4.5 pts** **D** Explains MULTIPLE features or functionalities of an ORM to a THOROUGH level of detail.	**4.5 to >4 pts** **C** Explains MULTIPLE features or functionalities of an ORM to a BASIC level of detail.	**4 to >2.99 pts** **P** Explains ONE feature or functionality of an ORM to a BASIC level of detail.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
PMG1003-2.1, PMG1003-7.3: EXPLAINS a plan for normalised database relations.	**12 to >10 pts** **HD** Meets D, and the explanation includes comparisons to how AT LEAST ONE model or relations would look in other levels of normalisation than the one shown in the ERD.	**10 to >9 pts** **D** Meets CR, and the ERD includes a legend/key of the notation and styles matching a notation or style identified in the accompanying explanation.	**9 to >8 pts** **C** Provides a BASIC entity relationship diagram (ERD), with a BRIEF explanation of how ALL of the relations depicted in the ERD are normalised.	**8 to >5.99 pts** **P** Provides a BASIC entity relationship diagram (ERD), with a BRIEF explanation of how SOME of the relations depicted in the ERD are normalised.	**5.99 to >0 pts** **No marks** Does not provide an ERD, or provides an ERD with no relevant explanation or context.	/ 12 pts

T2A2: API Web Server

Criteria	Ratings					Points
CMP1001-7.2: DESCRIBES the project's models in terms of the relationships they have with each other.	**6 to >5 pts** **HD** Meets D, and includes appropriate code examples supporting the descriptions.	**5 to >4.5 pts** **D** Meets CR, and includes information about the queries that could be used to access data using the models' relationships.	**4.5 to >4 pts** **C** Meets P, and includes BRIEF information about how the relationships of the models interact with other models.	**4 to >2.99 pts** **P** Provides a BRIEF description about the project's models and their relationships, with MINIMAL information about the types of relationships that the models have.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
CMP1001-1.4: IDENTIFY AND DESCRIBE the application's API endpoints.	**6 to >5 pts** **HD** Meets D, applied to ALL of the application's API endpoints.	**5 to >4.5 pts** **D** Meets P, and includes examples of what each identified endpoint will return on success AND failure of that endpoint operation.	**4.5 to >4 pts** **C** Meets P, and includes examples of what each identified endpoint will return on success OR failure of that endpoint operation.	**4 to >2.99 pts** **P** Identifies MOST of the application's API endpoints, including (for each identified endpoint) the HTTP verb, route path, and any required body or header data.	**2.99 to >0 pts** **F** Provides a LIMITED RESPONSE which is INSUFFICIENT in DETAIL, LENGTH or NOT RELEVANT to the question asked.	/ 6 pts
PGM1003-2.2: IMPLEMENTS a normalised database design.	**6 to >5 pts** **HD** Meets D with no duplication and ideal model implementation.	**5 to >4.5 pts** **D** Implemented models each serve a single purpose, contain appropriate fields and relationships. There may be a little duplication.	**4.5 to >4 pts** **C** Implemented models are sufficient for the application but there is some duplication and unnecessary/ poorly defined entities.	**4 to >2.99 pts** **P** Models implemented, but with significant duplication and unnecessary or poorly defined entities.	**2.99 to >0 pts** **F** Models or fields crucial to the application are missing or do not work.	/ 6 pts
PGM1003-6.2: IMPLEMENTS a database design that appropriately addresses the requirements of the planned scenario.	**6 to >5 pts** **HD** Meets D and represents a highly optimised or normalised solution.	**5 to >4.5 pts** **D** All tables, fields, and relationships adequately represent an appropriate solution.	**4.5 to >4 pts** **C** Most tables, fields, and relationships adequately represent an appropriate solution.	**4 to >2.99 pts** **P** Some tables, fields, or relationships are missing or incorrect, but do not significantly impact the solution.	**2.99 to >0 pts** **F** Significant tables, fields, or relationships are missing or incorrect, preventing a working solution.	/ 6 pts
PGM1003-4.1: IMPLEMENTS database queries that provide correct data for the given scenario.	**6 to >5 pts** **HD** Implements queries that provide ALL data needed for a working solution, and the queries are suitably complex and optimised.	**5 to >4.5 pts** **D** Implements queries that provide ALL data needed for a working solution, but the queries are suboptimal or could be refined using complex queries.	**4.5 to >4 pts** **C** Implements queries that provide MOST data needed for a working solution, but the queries are suboptimal or could be refined using complex queries.	**4 to >2.99 pts** **P** Implements queries that provide SOME data needed for a working solution.	**2.99 to >0 pts** **F** No data needed for a working solution is available.	/ 6 pts
PGM1003-4.2: WRITES code comments that demonstrate how the queries implemented correctly represent the database structure.	**6 to >5 pts** **HD** ALL queries or model methods are commented to a THOROUGH level of detail, with reference to a style guide or comment style guide in the project documentation.	**5 to >4.5 pts** **D** ALL queries or model methods are commented to a BASIC level of detail.	**4.5 to >4 pts** **C** MOST queries or model methods are commented to a BASIC level of detail.	**4 to >2.99 pts** **P** SOME queries or model methods are commented to a BASIC level of detail.	**2.99 to >0 pts** **F** No code comments to describe implemented queries or model methods.	/ 6 pts
PGM1003-5.2: IMPLEMENTS sanitization and validation techniques on user input to maintain data integrity	**6 to >5 pts** **HD** Validates ALL user input AND sanitises user input where relevant.	**5 to >4.5 pts** **D** Validates MAJORITY OF user input AND sanitises user input where relevant.	**4.5 to >4 pts** **C** Validates MAJORITY OF user input BUT DOES NOT sanitise.	**4 to >2.99 pts** **P** Validates SOME user input BUT DOES NOT sanitise.	**2.99 to >0 pts** **F** Does not sanitise or validate user input within the application.	/ 6 pts

Total points: 0

Choose a submission type

Upload Canvas Studio More

Choose a file to upload

File permitted: ZIP

or

Canvas Files

Submit assignment