**SCHOOL OF COMPUTING (SoC)**

**ST0503 Back-end Web Development**

**2021/2022 SEMESTER 2**

**ASSIGNMENT 1**

**Instructions and Guidelines:**

1. The assignment **source code** must be submitted before 3rd Jan 2022, 8am. You are required to submit your source codes to the BlackBoard. Remember to provide your Class, Group, Admission Number(s) and Name(s) on the softcopy.
2. A one page word document showing the tables you have created and their linkage (including foreign keys) should be included in the submission.
3. You are required to **clearly** provide instructions on how to setup the project on the lecturer's laptop in a text file that is to be included in the softcopy submission.
4. Students are to work in a group of 1-2 members.
5. **Students of 2 member group must complete one of the 2 additional features (without extra marks) stated in the document or be penalized 10 marks**.
6. Marks will be given separately for each student in the group, depending on his contribution to the assignment. The assignment will account for **30%** of your final grade*.*
7. The assignment should be implemented using Node JS, Express and MySQL.
8. The interview will be conducted during the lessons in the week of **3rd Jan 2022**. You are expected to explain the program logic and modify the program during the interview. **If you are absent, you will be awarded zero mark for the assignment.**
9. Your application will be tested with POSTMAN.
10. **No marks will be awarded**, if the work is copied or you have allowed others to copy your work. Warning: Plagiarism means passing off as one's own the ideas, works, writings, etc., which belong to another person. In accordance with this definition, you are committing plagiarism if you copy the work of another person and turning it in as your own, even if you would have the permission of that person. Plagiarism is a serious offence and disciplinary action will be taken against you. If you are guilty of plagiarism, you may fail all modules in the semester, or even be liable for expulsion.
11. 50% of the marks will be deducted for assignments that are received within ONE (1) calendar day after the submission deadline. No marks will be given thereafter.

Exceptions to this policy will be given to students with valid LOA on medical or

compassionate grounds. Students will need to inform the lecturer as soon as reasonably possible. Students are not to assume on their own that their deadline has been extended.

# Assignment 1: SP IT!

## Background

**SP IT!** is considering setting up an online e-store to allow the public to purchase IT products it is selling. Before it launches its online e-store, it requires a web application to computerize its inventory management module and also allow the public to view the product details online.

As such, SP IT! has tasked you to design the backend API Specs the website. The API specs would support functionalities such as user registration, publication of product info, insertion of products and user reviews.

# Assignment Requirements

You are required to fulfil the following basic requirements:

* Create a new MySQL database with the tables needed for this project.
* Proper database design with correct use of primary and foreign key constraints.
* Create an Express server that comply with the specs provided.
* Consume data from MySQL using the mysql library.

Bonus Requirements:

* Create endpoint for image uploading/storage and retrieval of product listing from the server. Server should only accept **jpg** or **png** images below **1 MB**.
* Create endpoints related to the offering of discounts for promotional periods. Your endpoints should include GET, POST, and DELETE. GET endpoints should provide info of promotion period and discount offered for particular product. Note that the appropriate table(s) should be created in the database and that a product listing can have multiple promotional periods.

**(Students of 2 member group must successfully complete one of the above 2 advanced features (without extra marks awarded) or face a penalty of 10 marks. 2 member teams must also provide a listing detailing the contribution of each member. This is used to determine the individual contribution, and members who do not contribute will be PENALIZED).**

**Grading Guidelines**

The assignment will be assessed based on the following criteria:

* Demonstrate and satisfy the web api endpoint functionalities listed below to access/update and return relevant data from the database upon success or failure(75%)

Note: There are 12 APIs, with each api taking 4-10 marks.

Components of grading per api is based primarily on correctness(returning the right json data with proper Model, Controller layer and database calls) and returning the right failure message. Meaningful comments and proper code practices would also be assessed.

* Proper database design and creation of tables(User, product, category, reviews and userInterest) with sample data in MySQL(5%)
* Advanced Features(10%)
* Amount of individual contribution to the project
* Proper Documentation - screenshots with proper testing of webservices using postman and database schema design(10%)
* Question & Answer during the interview
* **Please note any advanced features would need to be attempted by yourself, as a showcase of your independent learning in order for you to score the advanced feature marks.**

# SP IT! API Specs

For this assignment, you are required to create the following endpoints. The response message body for failed operations can be determined by you in the API.

## 1)/

## Endpoint: POST /users/

Used to add a new user to the database.

### Success Response

**Code**: 201 Created

**Content**: ID of the newly created user:

{   
 "userid": 1  
}

**Request Body**:

Ensure that the id and created timestamp are autogenerated and not provided by the user.

{  
   
 "username": "Terry Tan",  
 “email”: “[terry@gmail.com](mailto:terry@gmail.com)”,  
 "contact": "91234567",  
 “password”: “abc123456”  
 "type": "Customer",  
 "profile\_pic\_url": "https://www.abc.com/terry.jpg"  
}

### Error Response(s)

**i)Condition: The new username OR new email provided already exists.**

**Code: 422 Unprocessable Entity**

**ii)Condition**: Unknown error

**Code**: 500 Internal Server Error

## 2)/

## Endpoint: GET /users/

**Request body schema**: N/A

### Success Response

**Code**: 200 OK

**Content**: Array of all the users (subset of users’ data) in the database, who may be admin or customer type:

[  
 {  
 "userid": 1,  
 "username": "Terry Tan",  
 “email”: “[terry@gmail.com](mailto:terry@gmail.com)”,  
 "contact": "91234567",  
 "type": "Customer",  
 "profile\_pic\_url": "https://www.abc.com/terry.jpg",  
 "created\_at": "2021-11-02 17:54:57"  
 },  
 ...  
]

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 3)/

## Endpoint: GET /users/:id/

Retrieve a single user by their id.

**Request body schema**: N/A

### Success Response

**Code**: 200 OK

**Content**: Subset of the data of the matching single user:

{  
 "userid": 1,  
 "username": "Terry Tan",  
 “email”: “[terry@gmail.com](mailto:terry@gmail.com)”,  
 "contact": "91234567",  
 "type": "Customer",  
 "profile\_pic\_url": "https://www.abc.com/terry.jpg",  
 "created\_at": "2021-11-02 17:54:57"  
}

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 4)/

## Endpoint: PUT /users/:id/

Update a single user. ID and created timestamp should not be updatable.

**Request body schema**:

Refer to the request body schema for the POST /users endpoint.

### Success Response

**Code**: 204 No Content

**Content**: N/A

### Error Response(s)

**i)Condition: The new username OR new email provided already exists.**

**Code: 422 Unprocessable Entity**

**ii)Condition**: Unknown error

**Code**: 500 Internal Server Error

5)/

## Endpoint: POST /category

Inserts a new category. Category ID should be auto generated.

**Request body schema**:

{  
 “category ": "Laptops",

“description”: “Light and high performance laptops for school”  
   
  
}

### Success Response

**Code**: 204 No Content

**Content**: N/A

### Error Response(s)

**i)Condition: The category name provided already exists.**

**Code: 422 Unprocessable Entity**

**ii)Condition**: Unknow error

**Code**: 500 Internal Server Error

## 6)/

## Endpoint: GET /category

Gets all category.

**Request body schema**:

N/A

### Success Response

**Code**: 200

**Content**:

[{  
 “categoryid": "1",

“category": "Laptops",

“description”: “Light and high performance laptops for school”  
   
},

…

]

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 7\*)/

## Endpoint: POST /product/

Used to add a new product to the database.

**Request body schema**:

Ensure that the id and created timestamp are autogenerated and not provided by the user.

{  
 "name": "SP AMD Ryzen 3600 Laptop",  
 "description": "Best Laptop",  
 “categoryid”: 1,  
 “brand”: “SP IT!”  
 "price”:”1855.50”  
}

### Success Response

**Code**: 201 Created

**Content**: ID of the newly created listing:

{  
 "productid": 1  
}

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 8)/

## Endpoint: GET /product/:id

Retrieves product info with matching product id.

**Request body schema**: N/A

### Success Response

**Code**: 200 OK

**Content**: Info of the matching product (including category name):

{  
 "name": " SP AMD Ryzen 3600 Laptop",  
 "description": "Best Laptop",  
 “categoryid”: 1,

“categoryname”,“Laptop”,  
 “brand”: “SP IT!”  
 "price”:”1855.50”  
},

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 9)/

## Endpoint: DELETE /product/:id/

Deletes a product given its id. **The associated reviews related to the product would also be deleted**. Idempotent.

**Request body schema**: N/A

### Success Response

**Code**: 204 No Content

**Content**: N/A

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 10)

## Endpoint: POST /product/:id/review/

Adds a review for a product listing. A product can have many reviews.

**Request body schema**:

Ensure that the review id and created timestamp are autogenerated and not provided by the user.

{

“userid”:1,  
 “rating”:5  
 "review": “Good Laptop, super fast and can play game in class!”  
  
}

### Success Response

**Code**: 201 Created

**Content**: ID of the newly created listing:

{  
 "reviewid": 1  
}

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 11)

## Endpoint: GET /product/:id/reviews

Retrieves all the reviews of a particular product, including the username of the reviewer (tables join required).

### Success Response

**Code**: 200 OK

**Content**:

[  
 {  
 "productid":1,  
 “userid”:1  
 “username”:”Terry Tan”,  
 “rating”:5  
 "review": “Good Laptop, love gaming in school!”,  
 "created\_at": "2021-11-15 18:54:57"  
 },  
 ...  
]

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error

## 12\*)

Endpoint: POST /interest/:userid

User can indicate the category of products they are interested in in their profile preferences. This will be used by the system to provide customized product recommendations to the users.

Note that each user can indicate **multiple categories** they are interested in. Each category can also have **multiple users** interested in it. The userid and categoryid should be stored in a separate table as per Relational database principles (many to many).

(Reference: https://fmhelp.filemaker.com/help/18/fmp/en/index.html#page/FMP\_Help/many-to-many-relationships.html)

**Request body schema**:

One or more category ids separated by “,”

{

categoryids:1,2,4

}

### Success Response

**Code**: 201 Created

**Content**:

Nil

### Error Response(s)

**Condition**: Unknown error

**Code**: 500 Internal Server Error