|  |  |
| --- | --- |
| **Name:** | **Rendy Shi** |
| **Email:** | **rendy\_shi@mpa.gov.sg** |

**Question 1**

In your own words explain the 4 sub constraints in REST’s Uniform Interface. Give an everyday example to illustrate each of the constraint.

1. Resource can be uniquely identified, for example GET student/profile/1.
2. Manipulation of resources through its representation where for the same piece of data could be in defined form such as json, pdf, html.
3. Message from client to server will contain all the necessary information pertaining to the request and respond.
4. Engine to change the application state such as hyperlink to another resource or actions for changes in state.

**Question 2**

What is the difference between the following HTTP methods?

1. POST, PUT and PATCH

POST is used to submit the data to a specific resource which normally used as creation.

PUT is used to replace the data of specific resource which normally used as update.

PATCH is used to partially modify a specific resource which normally used as update to a certain field.

1. GET and HEAD

GET is to request for a representation/data from the specified resource which normally used as retrieval of information for display

HEAD is similar to GET but it will not return a message body / payload. It only return the METADATA.

**Question 3**

You have a monolithic web application for managing warehouses. The application exposes the following end points

* /warehouses – list of all warehouses
* /warehouse/<warehouse\_id> – returns the warehouse’s details
* /warehouse/<warehouse\_id>/inventories – inventory list for the warehouse
* /inventories – list of all the inventories
* /inventory/<inventory\_id> – inventory detail
* /inventory/<inventory\_id>/report – generate a report

Describe how you can scale this application

1. By duplication

Creating of more instances of the application to serve the request. All the application will be accessing to the same database.

1. By functional decomposition

Creating of microservices for Warehouse and Inventory and having one instance for each microservice and having a dedicated database. Interaction between the two microservices will be via API calls.

1. By data partitioning

Creating different database for Warehouse and Inventory. The application server will be able to know which database to access for resource based on the request.

**Question 4**

Study the top headlines REST API from newsapi.org. Answer the following questions

1. List the different ways you can present the API key when performing an invocation
   * Via the apiKey querystring parameter.
   * Via the X-Api-Key HTTP header.
   * Via the Authorization HTTP header.
2. Construct a URL to get 30 technologies headlines from Japan

* <https://newsapi.org/v2/top-headlines?country=jp&category=technology&pagesize=30&apiKey=API_KEY>

1. What is the status code if an incorrect API key is used?

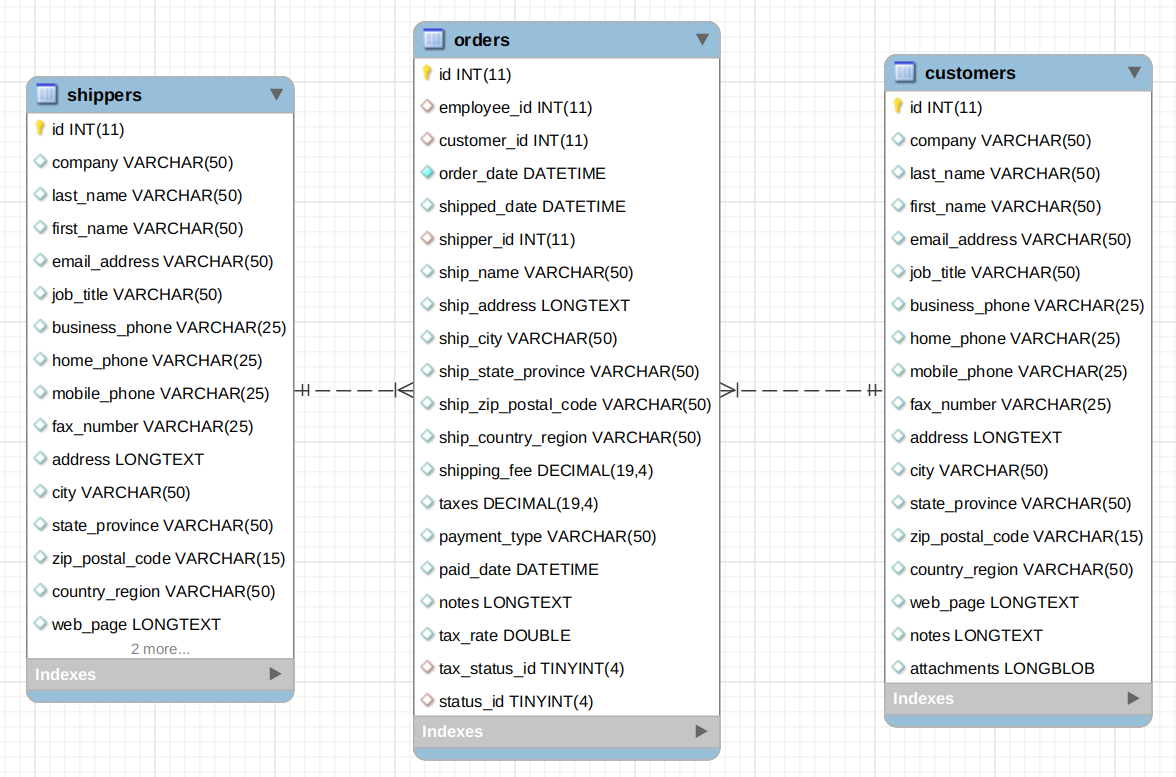
* 401 Unauthorized.

1. How long will the result be cached?

* 3 mins

**Question 5**

Study the following entity-relationship diagram



Both customer and shippers has a one to many relationships with orders.

Answer the following questions.

1. Design one or more API endpoints to return a list of customers and a single customer

* api/customers
* api/customer/1

1. What are some criteria and how might you might include in your endpoint (wrt Q5a)?

* api/customer/1/orders
* api/customer/1/order/1
* api/customer/1/shippers
* api/customer/1/shipper/1

1. Show a sample output of a customer’s list as a result of performing a GET on the resource. (wrt Q5a)

* api/customers

[ “/api/customer/0”, “/api/customer/1”, “/api/customer/2”, ... ]

* api/customer/1

[{ custId: 0, name: ‘ACME’ }]

1. How do you provide flow control or pagination support (wrt Q5a)?

Implementation of paging in response that allow number of returns returned to be specified and where to start from in the request header.

Implementation of rate limiting to control the number of requests processed

**Question 6**

You have deployed a service to encode video viz. convert AVI to mp4, etc. Subscribers of your service uploads their video to the service; after conversion the converted video is returned to the subscriber (assume that the conversion time is short).

You charge the subscribers based on the 2 criteria.

1. Subscription rates based on the cumulative video sizes: 500GB, 1TB, 1.5TB, etc. A subscriber who subscribe to the 500GB package can upload a maximum amount of 500GB videos.
2. Charge the subscribe based on their ingress and egress traffic viz. the upload and downloads of the videos.

Design an API for this encoding service to give your subscribers control over their encoding process.

You can ignore authentication.

* Usage of rate limiting to check the video size uploaded and left for feedback to subscribers to manage their videos
* POST endpoint which use content-type is multipart/form-data
* Usage of chunked transfer-encoding

**Submission**

Copy this Word document to your repository and commit it.

git add .

git commit -m ‘worksheet01’

git push origin master