**COMP1835**

**Lab 5**

**Working with JSON and JSON Schema**

**Overview**

In this lab you will be creating and validating JSON objects. You can use any JSON Editor to create JSON files, or regular Text Editor. You do not need to use any of your VMs for this lab. Not all editors have capability to validate JSON document data against JSON schema. For validating your JSON data you can use free on-line tool - JSON Validator:

<https://www.liquid-technologies.com/online-json-schema-validator>

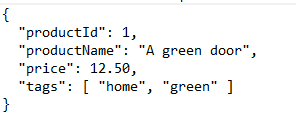
**Lab 5.1 Creating JSON documents**

You can use any free JSON Editor of your choice or Notepad to create a JSON document.

You are asked to create a JSON based product catalog. This catalog defines each product which has:

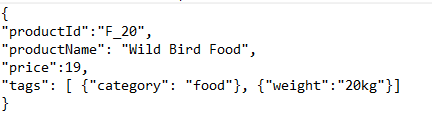
* An identifier: **productId**
* A product name: **productName**
* A selling cost for the consumer: **price**
* An optional set of tags: **tags**.

Use any JSON Editor or Notepad to create JSON document:



save as **prod1.json** into new folder called JSON somewhere on your network drive.

You can create a second JSON document for the second product, for example:



**Lab 5.2 Creating JSON Schema.**

We can store the above documents in the same collection, but it will not be a good practice, as the data types of the values are not the same in both and therefore, we might have problems during the manipulation of these documents in the future.

In order to prevent ‘garbage-in, garbage-out’ situation we would need to ensure that both documents (and any future JSON documents describing products in our catalog) do adhere to certain validation rules.

As there are several questions about data types of the values in the above documents that we cannot answer just based on the documents themselves:

1. What data type is productid?
2. Is productName required?
3. Can the price be zero (0)?
4. Are all the tags string values? Is there minimum number of tags, or maximum number of tags?

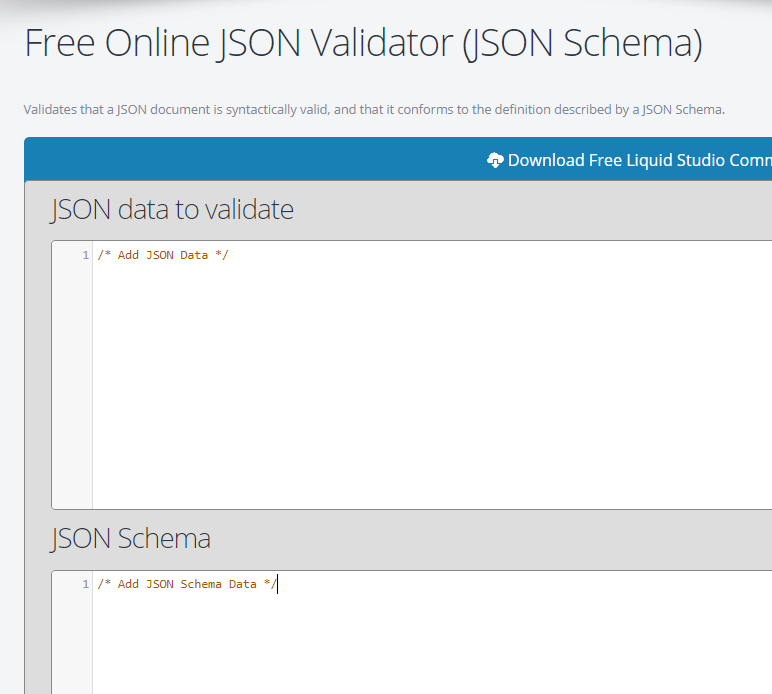
We need to create a JSON schema to answer those questions and then validate all our JSON documents against that schema to ensure the accuracy of the data.

**5.2.1**

Create JSON schema: start by defining four keywords: **$schema, $id, title, description and type:**

You can use online JSON Validator. Open Browser outside of your VM and navigate to:

<https://www.liquid-technologies.com/online-json-schema-validator>



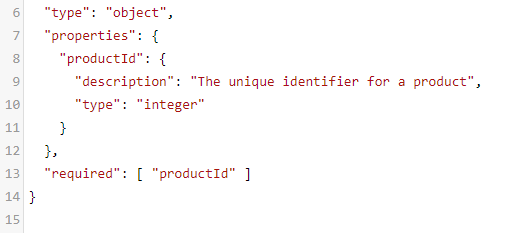
In the bottom part of the window start defining JSON schema:



**5.2.2**

Now you need to identify productId as a numeric value that uniquely identifies a product and which is required for each product, by adding key properties:

So, put comma after “object” in line 6 and add code to define properties:



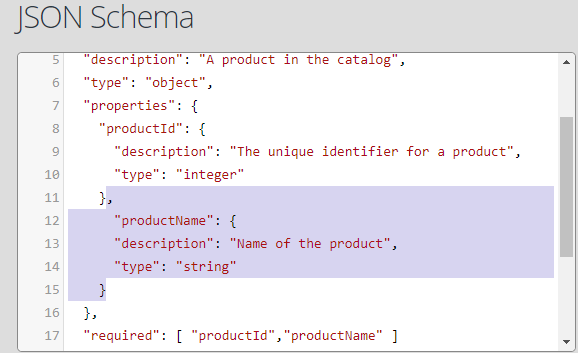
**5.2.3**

Now you need to define **productName**

**1. productName** is a string value that describes a product. As each product must have a product name, we need to ensure that **productName** is mandatory (required) as well.

Since the required validation keyword is an array of strings, we can note multiple keys as required; so, we just include **productName** into the array after “required”.

2. Insert code that defines **productName** and add **productName** into array **required:**

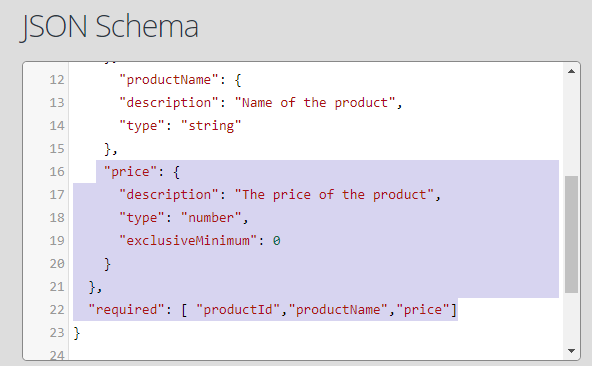


**5.2.4.**

Now you need to describe **price** key

1. Add  **price** key with the usual **description** schema annotation and **type**  validation keywords as previously. It also needs to be included in the array of keys defined by the **required** validation keyword.

2. Since the price cannot be 0, we must use the **exclusiveMinimum**validation keyword. (If we wanted to include zero as a valid price, we would have specified the **minimum** validation keyword.



**5.2.5.**

Next, we need to describe the **tags**key.

The requirements are:

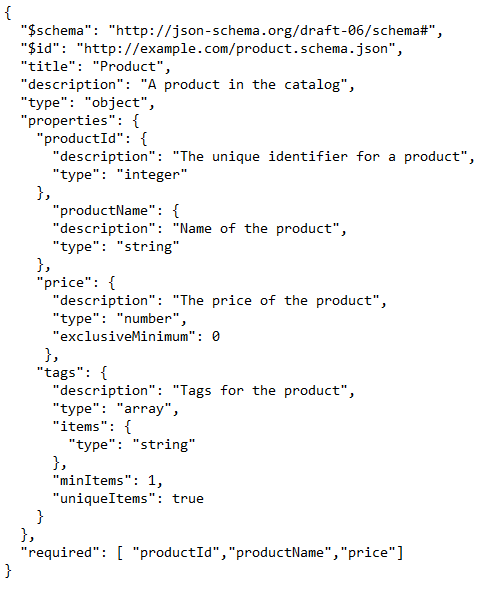
1. The document should be able to store several tags.
2. All tags must be text.
3. If there are tags there must be at least one tag,
4. All tags must be unique, no duplication within a single product.
5. Tags are useful, but they are not required to be present.

We implement those as following:

1. The **tags** key is added with the usual annotations and keywords, the type should be array.
2. We need to use **items** validation keyword so we can define what appears in the array. In this case the items of the array are strings, so we use "type": "string" validation keyword.
3. The **minItems** validation keyword is used to make sure there is at least one item in the array.
4. The  **uniqueItems** validation keyword will ensure the uniqueness.
5. Since this key is optional, we do not need to add this key to the **required** validation keyword array.



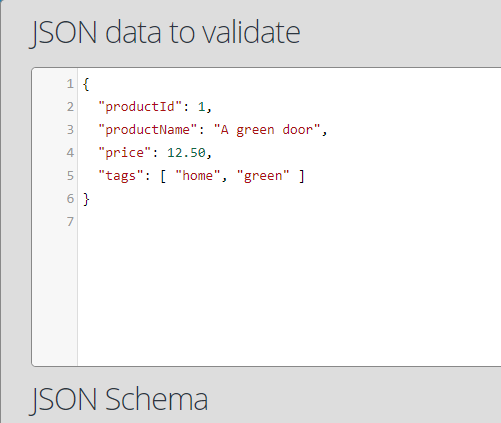
JSON schema is complete. Check the syntax, it should look like this:

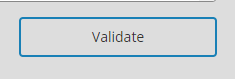


**Lab 5.3 Validating JSON document against JSON schema**

We defined JSON schema for product catalog. Now we need to validate different JSON documents against that schema. You can start with JSON document we had above.

**5.3.1** Type it in the top part of the online JSON Validator:



And press button Validate at the bottom of the screen 

What is the result?

**5.3.2**

Try to validate this document:



What is the result? Explain.

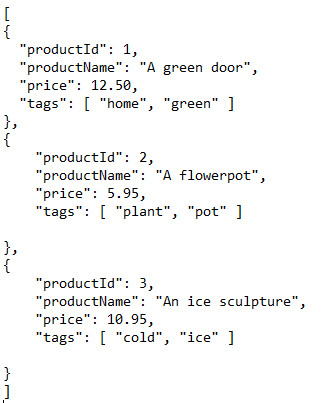
**5.3.3**

Validate the following:



**5.3.4**

Now you need to add at least 3 more products to the above JSON document taking into consideration all validation requirements set in our schema.



Validate this document against our schema. What is the result?

How to fix it?

Fix it and validate again.

**Lab 5.4. Using other elements of JSON schema.**

**5.4.1**

Update the above schema an incorporate other validation keywords, for example, (maxLength, minLength or pattern) for strings, (maxItems, maxContains, minContains) for arrays, (maxproperties, minProperties, dependentRequired) for objects.

**5.4.2**

Validate several JSON documents against new schema.

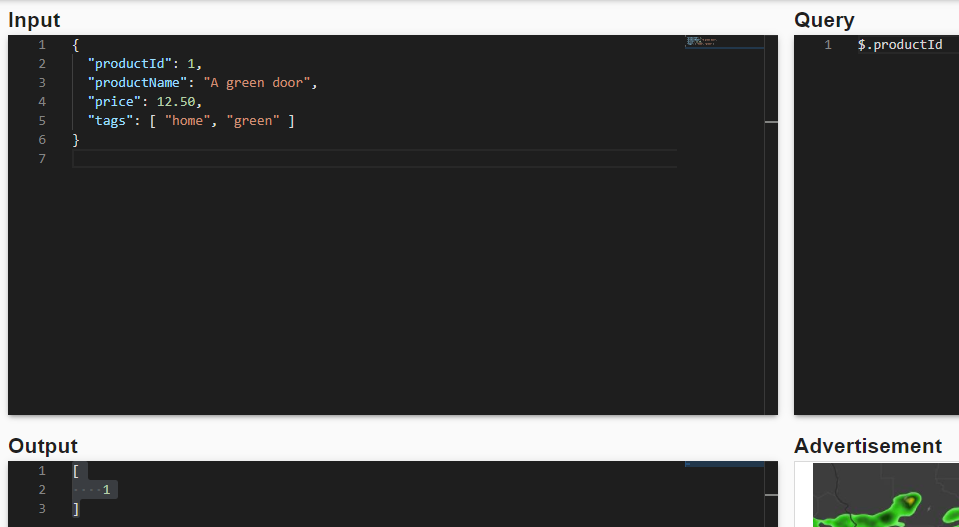
**Optional Lab 5.5. Querying JSON document.**

There are many online tools available to help you query JSON document. For example: [**https://www.jsonquerytool.com/**](https://www.jsonquerytool.com/)

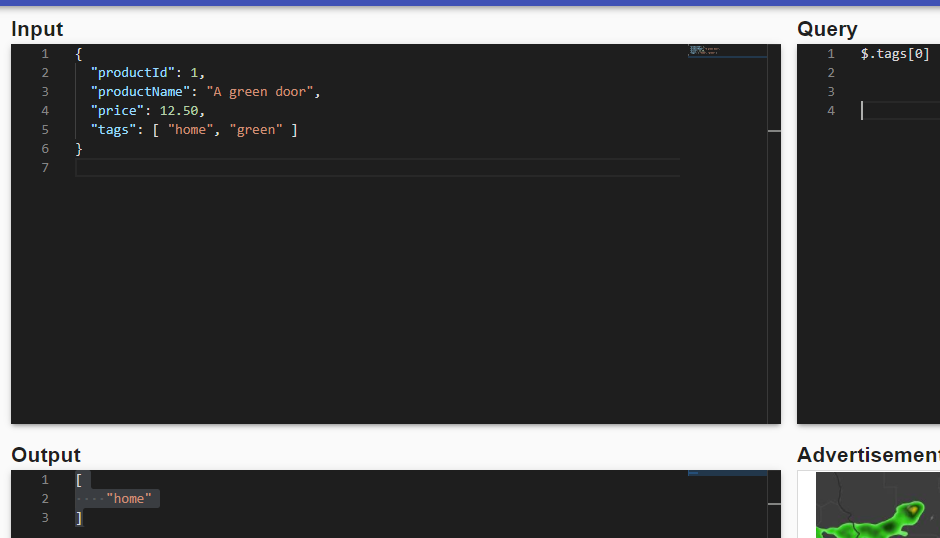
You can use this tool to query JSON documents loaded into Input area on the right-hand side and query string into Query area on the left. The result is displayed in the area Output.

For example,

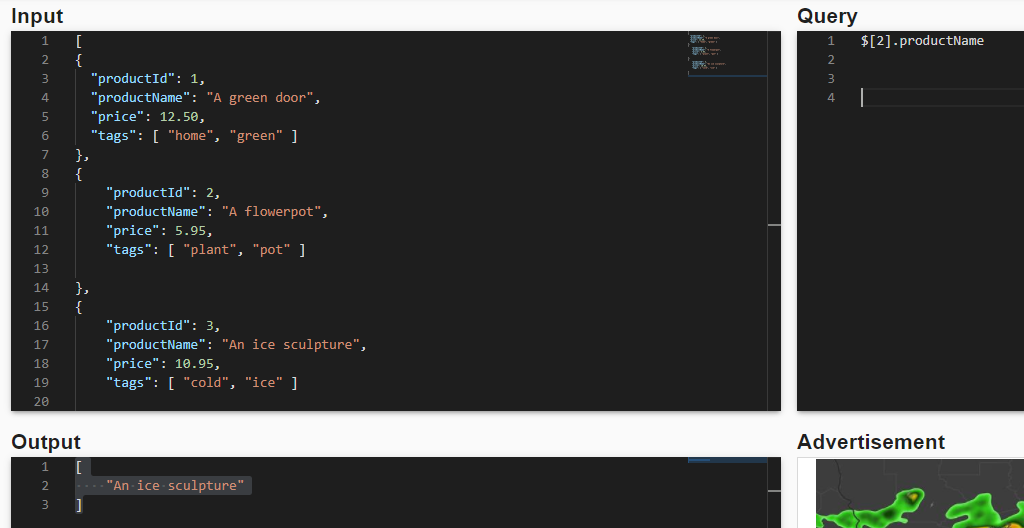
1. What is the value of the productid?



2. What is the value of the first tag?



3. What is the product name of the third object in this array?



You can experiment with different JSON documents.