University of Concepción
Faculty of Engineering

PRACTICE VI:
THE FIRST MICROSERVICE

Professor : Zheng Li
Student : Juan Fecci

Index.

Objectives

The objective of this work is implement a microservice only focus on the order management (all the purchase process) of a Pet Store and its domain logic using cloud systems with Gcloud.

To achieve this, the software will :

- Keep a record of orders
- Keep a record of the payment and items related to an specific order
- List all the orders stored in the system
- List all the orders made by an specific user
- List all the items of an order
- Update a payment of a order

The software is based on the MVC model and presents an API based on REST

Installation

To perform the installation correctly, it is necessary that the user have an account in Google and the system have python installed with the module virtualenv.

1. Create a GCP project and create an App Engine app in this page https://console.cloud.google.com/projectselector/appengine/create?lang=flex_python&st=true&_ga=2.55764200.-2103237472.1556472397 following the instructions

2. Download, install and initialize with your project the Google's Cloud SDK following the nexts instructions: https://cloud.google.com/sdk/docs/

3. Acquire local credentials for authenticating with GCP services using this command:

```
gcloud auth application-default login
```

4. Clone the repository of the order management with this command:

```
git clone https://github.com/juanfecci/OrderPetStore.git
```

5. Open the Datastore page in the GCP Console (https://console.cloud.google.com/datastore/?_ga=2.169600226.-2103237472.1556472397) and select Datastore Mode to initialize the service.

6. Install the dependencies of the system using the requirements.txt. If you want to create a virtual environment, use the following commands:

```
virtualenv -p python3 env
source env/bin/activate
pip install -r requirements.txt
```

Running the app

There are 3 ways for run the app:

Running on your local machine

1. With the virtual environment activated, start the local server with this command:

```
python main.py
```

2. In your browser, enter the following address for use the system: http://localhost:8080

Deploying the app

1. Deploy with the next command

```
gcloud app deploy
```

2. In your browser, enter the following address for use the system: https://[projectName].appspot.com *replace [projectName] with the name of your project

Use the deployed app

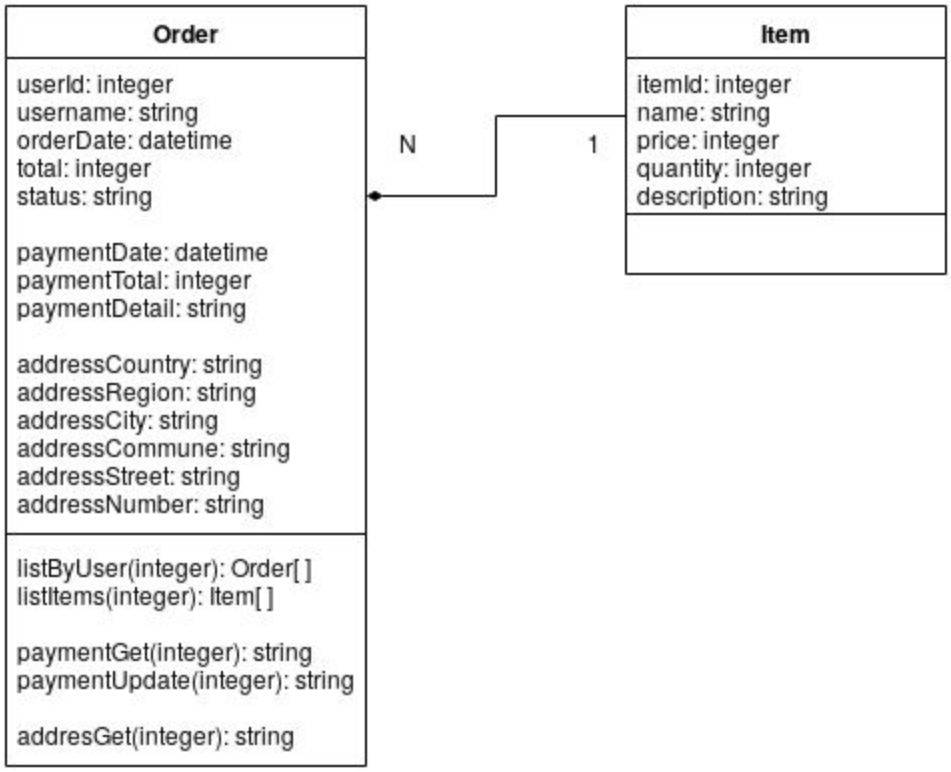
You can access to the deployed app with this address:

```
https://order-fecci.appspot.com
```

In this app an example page is available using the system and it is possible to interact through the api explained in the API reference of this report.

Class Diagram

It was decided to use MVC model for the realization of the modeling of the system. The data is stored in Google Cloud Datastore and the class diagram of the implemented models are as follow:



Description

The model was made focusing on the design of the order and prioritizing a faster performance versus space, where the following points must be highlighted.

- The order class is divided into three themes:
    a. Order details
    b. Payment details
    c. Address details

This implementation was planned to reduce the number of queries in the administration of orders

- The order's class have all the necessary elements to make their methods without the use of other api's

- The payment and addresses methods implemented in the order's class are for the management of this properties.

- The class Item was created to have a list of items relationated to the order's class. This have more information than the id to save the data in case that the item was deleted. For the same reason, the order's class save the username

- There is a more detailed description of every attribute in the api reference

Api Reference

The api reference is based in REST principles, requests are done via HTTP and all data received from API and submitted to API is JSON, the content type should be: `application/json.`

This API reference it's about one resource type, that has data representations and methods.

Order

Resource representations

Representation of a order.

```
{
      "id": integer,
      "user": integer,
      "userName": string,
      "orderDate": datetime,
      "total": integer,
      "status": string,
      "items": [
      {
            "id": integer,
            "name": string,
            "price": integer,
            "quantity": integer,
            "description": string
      }
      ],
      "paymentDate" datetime,
      "paymentTotal" integer,
      "paymentDetail" string,

      "addressCountry": string,
      "addressRegion": string,
      "addressCity": string,
      "addressCommune": string,
      "addressStreet": string,
      "addressNumber": string,
}
```

Property name	Value	Description
id	integer	The ID of the order
userId	integer	The ID of the user of the order
userName	string	The name of the user of the order
orderDate	datetime	The date of the order
total	integer	The total amount of the order
status	string	The status of the order
Items[ ]	list	The list of the items of the order
Items[ ].id	integer	The ID of the item
Items[ ].name	string	The name of the item
Items[ ].price	integer	The price of the item
Items[ ].quantity	integer	The quantity of the item
Items[ ].description	string	The description of the item
paymentDate	datetime	The date when the payment was created or modified
paymentTotal	integer	The total amount of payment
paymentDetail	string	The detail of the payment
addressCountry	string	The country of the address
addressRegion	string	The region of the address
addressCity	string	The cuty of the address
addressCommune	string	The commune of the address
addressStreet	string	The street of the address
addressNumber	string	The number of the address

Methods

Summary

Method	HTTP request	Description
get	GET /orderId	Gets a order by id
create	POST /create	Create a order
listAll	GET /list/all	Gets a list of all the orders in the system
listByUser	GET /list/user/userId	Get a list of all the orders associated with a user.
listItems	GET /items/orderId	Get a list of all the items associated with a order.
paymentGet	GET /payment/orderId	Gets a payment by the order id
paymentUpdate	PATCH /payment/update/orderId	Update a details of a payment using his associated order
addressGet	GET /address/orderId	Gets an address by the order id

Orders: get

Gets a order by id

Request

HTTP request

```
GET  http://127.0.0.1:5000/api/orders/orderId
```

Parameters

Parameter name	Value	Description
orderId	integer	The ID of the order

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a Order resource in the response body.

Orders: create

Create a order

Request

HTTP request

```
POST http://127.0.0.1:5000/api/orders/create
```

Parameters

Do not supply a http request parameter with this method

Request body

Property name	Value	Description
userId	integer	The ID of the user of the order
userName	string	The name of the user of the order
status	string	The status of the order
Items[ ]	list	The list of the items of the order
Items[ ].id	integer	The ID of the item
Items[ ].name	string	The name of the item
Items[ ].price	integer	The price of the item
Items[ ].quantity	integer	The quantity of the item
Items[ ].description	string	The description of the item
addressCountry	string	The country of the address
addressRegion	string	The region of the address
addressCity	string	The cuty of the address
addressCommune	string	The commune of the address
addressStreet	string	The street of the address

addressNumber	string	The number of the address

Response

If successful, this method returns a Order resource in the response body.

Orders: listAll

Gets a list of all the orders in the system

Request

HTTP request

```
GET  http://127.0.0.1:5000/api/orders/list/all
```

Parameters

Do not supply a http request parameter with this method

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a list of Order resource in the response body with all the orders in the system.

Orders: listByUser

Get a list of all the orders associated with a user.

Request

HTTP request

```
GET  http://127.0.0.1:5000/api/orders/list/user/userId
```

Parameters

Parameter name	Value	Description
userId	integer	The ID of the user

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a list of Order resource in the response body with all the orders associated with the user id.

Orders: listItems

Get a list of all the items associated with a order.

Request

HTTP request

```
GET  http://127.0.0.1:5000/api/orders/list/items/orderId
```

Parameters

Parameter name	Value	Description
orderId	integer	The ID of the order

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a response body with the following structure:

```
{
    "items": [
      {
            "id": integer,
            "price": integer,
            "quantity": integer,
            "description": string
      }
    ],
}
```

Property name	Value	Description
Items[ ]	list	The list of the items of the order
Items[ ].id	integer	The ID of the item
Items[ ].name	string	The name of the item
Items[ ].price	integer	The price of the item
Items[ ].quantity	integer	The quantity of the item
Items[ ].description	string	The description of the item

Orders: paymentGet

Gets a payment by the order id

Request

HTTP request

`GET  http://127.0.0.1:5000/api/payment/orderId`

Parameters

Parameter name	Value	Description
orderId	integer	The ID of the order associated to the payment

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a response body with the following structure:

```
{
      "paymentDate" datetime,
      "paymentTotal" integer,
      "paymentDetail" string,
}
```

Property name	Value	Description
paymentDate	datetime	The date when the payment was created or modified
paymentTotal	integer	The total amount of payment
paymentDetail	string	The detail of the payment

Orders: paymentUpdate

Update a details of a payment using his associated order

Request

HTTP request

`PATCH  http://127.0.0.1:5000/api/payment/update/orderId`

Parameters

Parameter name	Value	Description
orderId	integer	The ID of the associated order

Request body

Property name	Value	Description
detail	string	The new detail of the payment

Response

If successful, this method returns a response body with the following structure:

```
{
      "paymentDate" datetime,
      "paymentTotal" integer,
      "paymentDetail" string,
}
```

Property name	Value	Description
paymentDate	datetime	The date when the payment was created or modified
paymentTotal	integer	The total amount of payment
paymentDetail	string	The detail of the payment

Orders: addressGet

Gets an address by the orderid

Request

HTTP request

```
GET  http://127.0.0.1:5000/api/address/orderId
```

Parameters

Parameter name	Value	Description
orderId	integer	The ID of the associated order

Request body

Do not supply a request body with this method,

Response

If successful, this method returns a response body with the following structure:

Property name	Value	Description
addressCountry	string	The country of the address
addressRegion	string	The region of the address
addressCity	string	The cuty of the address
addressCommune	string	The commune of the address
addressStreet	string	The street of the address
addressNumber	string	The number of the address