

Hands-on Lab

Experimento – Mantenimiento de código

Duración:

120 min

Contactos:

- Wladymir Brborich
- Bryan Oscullo

Contenido del documento

- Objetivo General.
- Definición y requerimientos iniciales del sistema.
- Requisitos para el desarrollo del experimento.
- Modo de calificación de las tareas a realizar.
- Instrucciones para uso del repositorio donde se subirán los cambios solicitados.
- Descripción de las tareas de mantenimiento a realizar.

Objetivo

El objetivo de este documento es especificar las tareas que debe efectuar el desarrollador para realizar mantenimiento de una aplicación de servicios web.

Actualmente el sistema tiene su interfaz web completa y funcional. El subsistema backend cuenta con errores y requerimientos faltantes, los cuales el desarrollador deberá corregir e implementar.

Definición del Sistema

- El sistema web tiene como objetivo la simulación de gestión de una pizzería, en la cual es posible realizar órdenes de pizza especificando tamaño e ingredientes, también siendo posible la gestión individual de ingredientes y tamaños.

- El sistema se encontrará compuesto por dos subsistemas: una interfaz web para el usuario y un Backend de servicios Rest para interacción con información con la base de datos.

Requerimientos funcionales del sistema

1. Ingredientes

- a. El sistema permitirá la creación de nuevos ingredientes, con los siguientes parámetros: Nombre, Precio e ID, el cual será un entero autogenerado
- b. El sistema permitirá la actualización de los campos Nombre y Precio para cada ingrediente
- c. El sistema permitirá listar los ingredientes almacenados en la base de datos dentro de una tabla con todos sus parámetros.

2. Tamaños

- a. El sistema permitirá agregar nuevos tamaños de pizza con los siguientes parámetros: Nombre, Precio e ID, el cual será un entero autogenerado.
- b. El sistema permitirá listar los tamaños de pizza almacenados en la base de datos dentro de una tabla con todos sus parámetros.

3. Órdenes

- a. El sistema permitirá la creación de órdenes de pizza, las cuales tendrán los siguientes parámetros: Nombre del solicitante, Cédula del solicitante, Dirección del solicitante, número telefónico del solicitante, un único tamaño de pizza de los existentes en la base de datos y de 0 a n ingredientes disponibles en la base de datos.
- b. El sistema realizará el cálculo del precio total de la orden de manera automática, usando la siguiente fórmula: **suma**(precio de cada ingrediente) + precio del tamaño elegido.

- c. El sistema registrará automáticamente la fecha de la orden en el momento de su creación.
- d. El sistema permitirá listar las órdenes que hayan sido creadas, en una tabla que permita revisar todos sus parámetros.

Prerrequisitos

Este laboratorio asume que el desarrollador ha completado y comprendido:

- Entrenamiento en paradigma de programación Orientado a Objetos, lenguaje programación Java y manejo del framework de desarrollo Spring.
- Entrenamiento en paradigma de programación Procedimental, lenguaje de programación Python y manejo del framework de desarrollo Flask.

Requerimientos de software y sistema

- Windows 10 home
- Python 3.7.4
- Java 8 (JDK 8)
- Visual Studio Code (VSC)
- Paquete de extensiones de Java y Python para VSC.
- Conexión estable a internet

Rúbrica de evaluación del ejercicio

Para cada tarea de mantenimiento que debe efectuar el desarrollador se tiene un grupo de pruebas unitarias para verificar su cumplimiento y correcto funcionamiento.

- Si todas las pruebas unitarias son correctas se obtendrá una calificación por tarea de manteniendo de: 10/10 puntos.
- Si las pruebas unitarias son correctas parcialmente se obtendrá una calificación por tarea de mantenimiento de: 8/10 puntos.
- Si ninguna prueba unitaria es correcta se obtendrá una calificación por tarea de mantenimiento de: 7/10 puntos.

Repositorio

A cada desarrollador se le ha asignado una rama (branch) para subir sus cambios en un repositorio Git.

Por cada mantenimiento completado se deberá hacer un commit a su rama (total de por lo menos 5 commits).

Para el nombre de los commit se manejará el siguiente estándar de versiones:

Código (Versión), donde el código es el código único de la tarea (detallado más adelante en el presente documento), y la versión:

- 1.0 si los cambios corrigen por completo el error, la tarea es satisfactoria.
- 0.5 si los cambios corrigen parcialmente el error.
- 0.1 si los cambios no son significativos y la funcionalidad errónea persiste.

Mantenimiento 1: Obtener lista de tamaños de pizza no implementada

- **Código tarea de mantenimiento:** MA_S_01
- **Tipo de mantenimiento:** Adaptativo
- **Requerimiento:**

Obtener todos los tamaños de pizza: El sistema deberá presentar todos los registros de tamaños de pizza como una lista, cada elemento de la lista contendrá la información: ID, nombre y precio.

- **Descripción del error o fallo:** La consulta de tamaños response nulo, sin registros.
- **¿Cómo reproducir el error (interfaz web)?**

Ruta del servicio: <http://127.0.0.1:5500/app/size/sizes.html>

Salida:

- **Correcta:**

Sizes			
ID	Name	Price	Actions
1	Familiar	\$15.99	<button>Update</button>
2	Small	\$5.5	<button>Update</button>
3	Medium	\$10.5	<button>Update</button>
4	Extra Large	\$29.99	<button>Update</button>
<button>Add New Size</button>			

- **Incorrecta:** los tamaños de pizza no se obtienen

Sizes			
ID	Name	Price	Actions
<button>Add New Size</button>			

Build your own pizza!

Name:

DNI:

Address:

Phone #:

Choose a size

Choose your ingredients

- ☐ Onions1 - \$0.89
- ☐ Ham - \$0.6
- ☐ Anchovies - \$3.5
- ☐ Peperony - \$0.86
- ☐ Extra Cheese - \$1.15

Order

- **¿Cómo reproducir el error (servicio web)?**

Entrada: ninguna, solo llamada al servicio

Ruta del servicio: <http://localhost:5000/api/ingredient>

Salida:

- **Correcta:**

```
[
  { "_id": 1, "name": "Salami", "price": 0.5 },
  { "_id": 2, "name": "Peperoni", "price": 0.25 },
  { "_id": 3, "name": "Queso", "price": 0.4 }
]
```

- **Incorrecta:**

Error Code 404
null
[]

Duración estimada: 15 minutos

Mantenimiento 2: Orden Vacía o Nula

- **Código tarea de mantenimiento:** MC_O_01
- **Tipo de mantenimiento:** Correctivo
- **Requerimiento:**

Obtener orden por ID: El sistema deberá presentar la información de una orden mediante su identificador, mostrando: Id, nombre y precio en caso de existir el ingrediente. Se entrega como parámetro al servicio el identificador único del ingrediente (entero positivo).

- **Descripción del error o fallo:** La consulta de orden presenta una orden vacía, es decir con sus parámetros nulos.
- **¿Cómo reproducir el error (interfaz web)?**

Ruta del servicio: <http://127.0.0.1:5500/app/order/order-detail.html? id=5>

Orders				
Name	DNI	Address	Total Sale	Actions
Wladimir	1724561921	Conocoto	\$	View
Alexander	1724561921	Conocoto	\$	View
Juan	1724561921	Quito	\$	View
Pepe	1724561921	Cuenca	\$	View
Kevin	1724561921	Cuenca	\$17.44	View
Marcelo	1724561921	Cuenca	\$17.44	View

Salida:

- **Correcta:**

Order Detail - 2019-08-26T04:46:13.306052	
Name:	Kevin
DNI:	1724561921
Address:	Cuenca
Phone #:	0983147655
Size	Ingredients
Familiar - \$15.99	Onions1 - \$0.85 Ham - \$0.6
Total: \$17.44	

- **Incorrecta:** los campos están vacíos

Order Detail -

Name:
DNI:
Address:
Phone #:

Size	Ingredients
- \$	

Total: \$

- ¿Cómo reproducir el error (servicio web)?

Entrada: Entero positivo (identificador de la orden). Ejemplo: 1

Ruta del servicio: <http://localhost:5000/api/order/id/{1}>

Salida:

- **Correcta:**

```
{
  "_id": 1,
  "client_address": "Sangolqui",
  "client_dni": "1717196625",
  "client_name": "Pepe Pecas",
  "client_phone": "0983803458",
  "date": "2019-10-20",
  "detail": [
    {
      "ingredient": {
        "_id": 1,
        "name": "Salami",
        "price": 0.5
      },
      "ingredient_price": 0.5
    }
  ],
  "size": {
    "_id": 1,
    "name": "Mediana",
    "price": 15.0
  },
  "total_price": 15.50
}
```


○ **Incorrecta:**

```
{
  "_id": 1,
  "client_address": null,
  "client_dni": null,
  "client_name": null,
  "client_phone": null,
  "date": null,
  "detail": null,
  "size": null,
  "total_price": 0
}
```

Null

Error 404

Duración estimada: 25 minutos






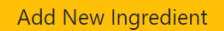
Mantenimiento 3: Obtener Ingrediente Vacío o Nulo

- **Código tarea de mantenimiento:** MC_I_01
- **Tipo de mantenimiento:** Correctivo
- **Requerimiento:**

Obtener ingrediente por ID: El sistema deberá presentar la información de un ingrediente mediante su identificador, mostrando: Id, nombre y precio en caso de existir el ingrediente. Se entrega como parámetro al servicio el identificador único del ingrediente (entero positivo).

- **Descripción del error o fallo:** La consulta del ingrediente presenta un ingrediente vacío, es decir con sus parámetros nulos.
- **¿Cómo reproducir el error (interfaz web)?**

Ruta del servicio: <http://127.0.0.1:5500/app/ingredient/ingredients.html>

Ingredients			
ID	Name	Price	Actions
1	Onions1	\$0.89	
2	Ham	\$0.6	
3	Anchiovies	\$3.5	
4	Peperony	\$0.86	
5	Extra Cheese	\$1.15	
			

Salida:


- **Correcta:**

Update Ingredient!

ID:

Name:

Price:



- **Incorrecta:** los campos están vacíos

Update Ingredient!

ID:

ID

Name:

Name

Price:

Price

Submit

- **¿Cómo reproducir el error (servicios web)?**

Entrada: entero positivo (identificador del ingrediente). Ejemplo: 1, 2 o 3.

Ruta del servicio: <http://localhost:5000/api/ingredient/id/{id}>

Salida:

- **Correcta:**

<code>{"_id": 1, "name": "Salami", "price": 0.5}</code>
<code>{"_id": 2, "name": "Peperoni", "price": 0.25}</code>
<code>{"_id": 3, "name": "Queso", "price": 0.4}</code>

- **Incorrecta:**

<code>{"_id": 0, "name": null, "price": 0}</code>
<code>null</code>
<code>Error 404</code>

Duración estimada: 25 minutos

Mantenimiento 4: Precio de Orden Erróneo

- **Código tarea de mantenimiento:** MC_O_03
- **Tipo de mantenimiento:** Correctivo
- **Requerimiento:**

Calcular precio orden: El sistema realizará el cálculo del precio total de la orden de manera automática, usando la fórmula: suma (precio de cada ingrediente seleccionado) + precio del tamaño de pizza elegido.

- **Descripción del error o fallo:** La orden se guarda satisfactoriamente, pero el cálculo del precio será incorrecto, \$0.
- **¿Cómo reproducir el error (interfaz web)?**

Ruta del servicio: <http://127.0.0.1:5500/app/order/orders.html>

Build your own pizza!

Name: Alexander

DNI: 1724561921

Address: Conocoto

Phone #: 2346352

Choose a size

- ☒ Familiar - \$15.99
- ☐ Small - \$5.5
- ☐ Medium - \$10.5
- ☐ Extra Large - \$29.99

Choose your ingredients

- ☒ Onions1 - \$0.89
- ☒ Ham - \$0.6
- ☐ Anchovies - \$3.5
- ☐ Pepperony - \$0.86
- ☐ Extra Cheese - \$1.15

Order

Salida:

- **Correcta:**

Orders				
Name	DNI	Address	Total Sale	Actions
Alexander	1724561921	Conocoto	\$17.48	View

- **Incorrecta:** precio incorrecto de la orden. Explicación: \$15.99 del tamaño de la pizza Familiar + \$0.89 del ingrediente Onion + \$0.6 del ingrediente Ham = \$17.48

Orders				
Name	DNI	Address	Total Sale	Actions
			\$15.99	View

- ¿Cómo reproducir el error (servicios web)?

Entrada: Detalle de la orden, entero positivo (identificador de la orden).

Ejemplo: 1

Ruta del servicio: <http://localhost:5000/api/order/id/{1}>

Salida:

- **Correcta:**

```
{
  "_id": 1,
  "client_address": "Sangolqui",
  "client_dni": "1717196625",
  "client_name": "Pepe Pecas",
  "client_phone": "0983803458",
  "date": "2019-10-20",
  "detail": [
    {
      "ingredient": {
        "_id": 1,
        "name": "Salami",
        "price": 0.5
      },
      "ingredient_price": 0.5
    }
  ],
  "size": {
    "_id": 1,
    "name": "Mediana",
    "price": 15.0
  },
  "total_price": 15.50
}
```

- **Incorrecta:**

```
{
  "_id": 1,
  "client_address": "Sangolqui",
  "client_dni": "1717196625",
  "client_name": "Pepe Pecas",
```

```
"client_phone": "0983803458",
"date": "2019-10-20",
"detail": [
  {
    "ingredient": {
      "_id": 1,
      "name": "Salami",
      "price": 0.5
    },
    "ingredient_price": 0.5
  }
],
"size": {
  "_id": 1,
  "name": "Mediana",
  "price": 15.0
},
"total_price": 0
}
```

Duración estimada: 30 minutos

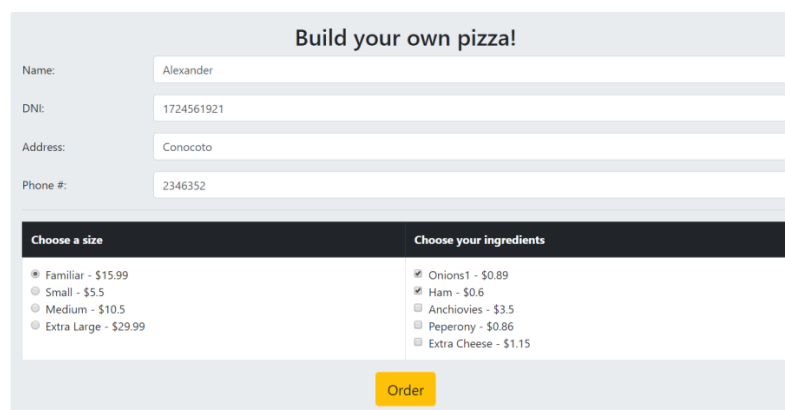
Mantenimiento 5: Orden sin Datos del Cliente

- **Código tarea de mantenimiento:** MC_O_02
- **Tipo de mantenimiento:** Correctivo
- **Requerimiento:**

Crear Orden: El sistema deberá crear órdenes de pizza, las cuales tendrán los siguientes parámetros: nombre del cliente, cédula, dirección, número, único tamaño de piza y 0 a n ingredientes disponibles.

- **Descripción del error o fallo:** La orden se guarda satisfactoriamente, pero los datos del cliente (nombre, cédula, dirección, número) están vacíos.
- **¿Cómo reproducir el error (interfaz web)?**

Ruta del servicio: <http://127.0.0.1:5500/app/order/orders.html>



Salida:

- **Correcta:**

Orders				
Name	DNI	Address	Total Sale	Actions
Alexander	1724561921	Conocoto	\$17.48	<button>View</button>

- **Incorrecta:** los campos de cliente están vacíos

Orders				
Name	DNI	Address	Total Sale	Actions
			\$17.48	<button>View</button>

- ¿Cómo reproducir el error (servicio web)?

Entrada: Detalle de la orden, entero positivo (identificador de la orden).

Ejemplo: 1

Ruta del servicio: <http://localhost:5000/api/order/id/{1}>

Salida:

- **Correcta:**

```
{
  "_id": 1,
  "client_address": "Sangolqui",
  "client_dni": "1717196625",
  "client_name": "Pepe Pecas",
  "client_phone": "0983803458",
  "date": "2019-10-20",
  "detail": [
    {
      "ingredient": {
        "_id": 1,
        "name": "Salami",
        "price": 0.5
      },
      "ingredient_price": 0.5
    }
  ],
  "size": {
    "_id": 1,
    "name": "Mediana",
    "price": 15.0
  },
  "total_price": 15.50
}
```

- **Incorrecta:**

```
{
  "_id": 1,
  "client_address": null,
  "client_dni": null,
  "client_name": null,
  "client_phone": null,
  "date": "2019-10-20",
  "detail": [
    {
```



```
        "ingredient": {
            "_id": 1,
            "name": "Salami",
            "price": 0.5
        },
        "ingredient_price": 0.5
    },
    "size": {
        "_id": 1,
        "name": "Mediana",
        "price": 15.0
    },
    "total_price": 15.50
}
```

Duración estimada: 30 minutos