



RETO 2

EQUIPO B

Alejandro Rebollo, Jon Zorzano, Mario Hernandez, Aaron Oliva,
Julen Lopez

ÍNDICE

Introducción	3
FOL Formación Y Orientación Laboral	4
English	5
Lenguaje de Marcas	6
Bases de datos	8
Sistemas informáticos	11
Programación	13
Página 1:	14
Página 2:	14
Página 2	16
Página 3	17
Entornos de desarrollo:	19
Introducción:	19
Lo que se nos pedía era realizar el trabajo en conjunto mediante Git bash y añadir a los programas Java comentarios (JAVADOC).	19
Imágenes:	19
Javadoc:	20
Mejoras en un futuro	20
Introducción:	21
Problemas:	21

Introducción

Para el reto de la segunda evaluación hemos decidido hacer nuestro proyecto sobre un restaurante cuyo nombre es “El Rincón de Oliva”. En este documento hablaremos sobre los pasos que hemos realizado para llevar a cabo el proyecto mencionando las partes que corresponden a cada asignatura en base a lo que hemos hecho cada día.

El servicio de la página web consiste en una gestión de reservas de un restaurante en forma de formulario que se tendrá que rellenar en dicha web y dicha información se guardará en un servidor de AWS que estará conectada a la base de datos.

FOL Formación Y Orientación Laboral

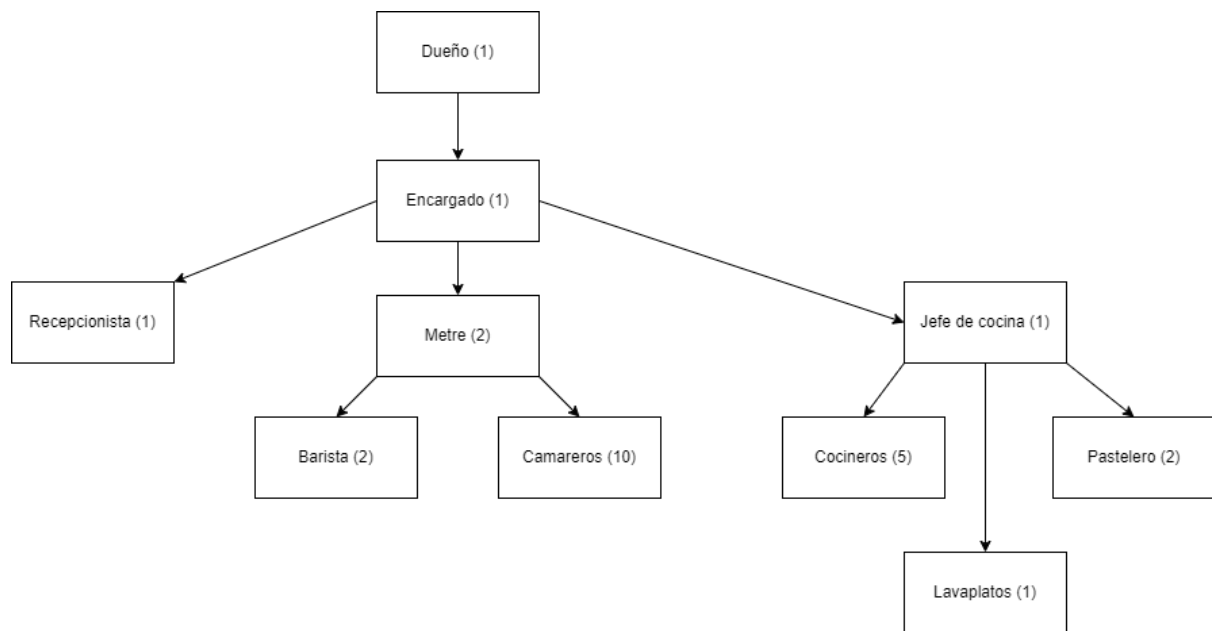
Introducción:

En la asignatura de Formación y Orientación Laboral nos pidieron que llevásemos a cabo una tabla de plan de prevención para cada puesto de trabajo de nuestra empresa y por otro lado teníamos que realizar una nómina de un trabajador.

Proceso llevado a cabo:

Esta parte del trabajo fue el inicio del proyecto, una vez que pensamos cuál sería nuestra empresa (Restaurante), empezamos a investigar en varias páginas web cuántos empleados serían necesarios para una capacidad de unas 150 personas, en dichas páginas indican que serían necesarios alrededor de unas 21 personas (2 metes , 2 recepcionistas , 10 camareros y 5 chefs) sin contar al dueño y al encargado.

Posteriormente a los empleados les asignamos un organigrama profesional, es decir, asignamos una jerarquía del restaurante para tener un orden(en modo gráfica).



Después de asignar los empleos del restaurante iniciamos el plan de prevención de riesgos laborales del restaurante gracias a la plantilla que teníamos en el classroom.

Para finalizar, realizamos una nómina a uno de los empleados del restaurante, gracias a una nueva plantilla que teníamos en classroom, a su vez, miramos en internet los convenios laborales que tiene en base al sector de la hostelería.

English

Introducción:

En la parte de Inglés, debíamos realizar un video explicando los servicios de nuestra página web, es decir, de cara al cliente debemos hacer una presentación en inglés.

Proceso llevado a cabo:

Alejandro Rebollo realizó la parte de la portada de la página web donde se ve la idea general y lo que ofrece nuestra empresa, Jon Zorzano presento la manera de realizar una reserva a través del formulario, Julen Lopez nos enseñó los chefs que conforman nuestro restaurante, Aaron Oliva presento la sección del menu, bebidas, comidas, precio,... Y por último Mario Hernandez presentó los platos y el aviso legal y la política de privacidad donde se pueden ver las reglas.

Lenguaje de Marcas

Introducción:

En lenguaje de marcas nos pedían crear una página web donde tendría que haber un formulario de reservas, con lo que luego deberíamos crear un XML Schema teniendo en cuenta los datos de las reservas. Y que el código creado tendrá que estar en un repositorio de GitHub.

Proceso llevado a cabo:

HTML:

Para este apartado del trabajo comenzamos realizando en VSCodium las páginas web del restaurante (en total 6: Portada, Reservas, Menú, Chefs, Aviso legal y Política de Privacidad). Para el tema de enlazar las páginas unas con otras usamos en html: href y button, el primero hay que indicar a qué página debe llevar (para eso debemos tener los distintos html's en la misma carpeta guardados) y luego usamos button para indicar qué mensaje queremos que salga en la página.

Para tener organizado el HTML y el CSS hemos usado los div's con el objetivo de dividir los objetos diferentes que aparecen en la página y de esta manera tenerlo más ordenado. Más tarde, hemos creado un Main con el objetivo de poder usar en el CSS los grid-row y grid-column (que ayudan mucho para poder colocar los objetos de la página).

Para las imágenes debemos de guardar en el mismo sitio que tenemos guardado los HTML y CSS para que el ordenador pueda localizar las imágenes a su vez en el html debes usar "img src" y la dirección donde está la imagen para que cargue dicha imagen.

También añadimos para las reservas del restaurante un formulario para dichas reservas, la estructura para hacer el formulario es sencilla. Por cada dato que pusimos añadimos un input, indicando el nombre del input, de que tipo es y indicamos que era obligatorio “required”. Al final del formulario añadimos el “button” enviar.

```
<div class="formulario">
  <form name="formulario" method="get" action="reservas.php" onsubmit="return validarFormulario()">

    <b>Tu Nombre:</b>
    <input type="text" name="Nombre" required><br><br>
    <b>Tu Apellido:</b>
    <input type="text" name="Apellido" required><br><br>
    <b>Correo Electrónico:</b>
    <input type="text" name="CorreoElectronico" required><br><br>
    <b>Número de teléfono:</b>
    <input type="int" name="Telefono"><br><br>
    <b>Número de personas</b>
    <input type="int" name="Personas" required><br><br>
    <b>Hora de Reserva</b>
    <input type="time" name="hora" required><br><br>
    <b>Fecha de reserva </b>
    <input type="date" name="fecha" required><br><br>
```

XSD:

A parte del html, otra parte del trabajo era realizar un xsd y un xml en base a lo puesto en el formulario.

El xsd no nos llevó mucho tiempo, lo más importante, indicamos que era una secuencia, es decir, se tiene que respetar el orden de los elementos. A su vez, abrimos dos elementos, el elemento “restaurante” que sería el primer elemento y un segundo elemento llamado reserva con el objetivo de poder añadir un “maxoccurs:unbounded”, de esta manera podemos añadir las reservas que quieramos.

Dentro del elemento restaurante pusimos que era complexType porque hay varios elementos dentro al igual que el elemento reserva. En el elemento nombre pusimos un “whitespace” por la posibilidad de un nombre compuesto. Más adelante, para el elemento de correo electrónico nos vimos en la obligación de poner un pattern, gracias a la profesora nos proporcionó dicho código del correo electrónico. El teléfono la única restricción que pusimos fue que como máximo solo acepta 9 números. Para los elementos fecha y hora indicamos como tipo date y time. Por último, añadimos el atributo de id ya que estos datos son sacados de la información insertada en el formulario.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" version="0.1" xml:lang="es">

  <xs:element name="Restaurante">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="reservas" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="nombre" minOccurs="1" maxOccurs="1">
                <xs:simpleType>
                  <xs:restriction base="xs:string">
                    <xs:whiteSpace value="preserve"/>
                  </xs:restriction>
                </xs:simpleType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

XML:

Esta parte fue la más sencilla ya que al estar bien estructurado el xsd solo teníamos que indicar los datos que se habían hecho por el formulario.

Lo única posible complicación era conectar bien el xsd y el xml. Para ello, guardamos ambos archivos en la misma carpeta e indicamos en las dos primeras filas del xml la conexión con el xsd.

```
Welcome  reservas.html  XSDReto.xsd  RetoXML.xml X  reservas.php
> Users > daw > Desktop > Reto2 > lenguajedemarcas > RetoXML.xml > {} Grammars > file:///c%3A/Users/daw/Desktop/Reto2/lenguajedemarcas/XSD
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Restaurante xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="XSDReto.xsd">
3  <reservas idReserva="1">
```

Bases de datos

Introducción:

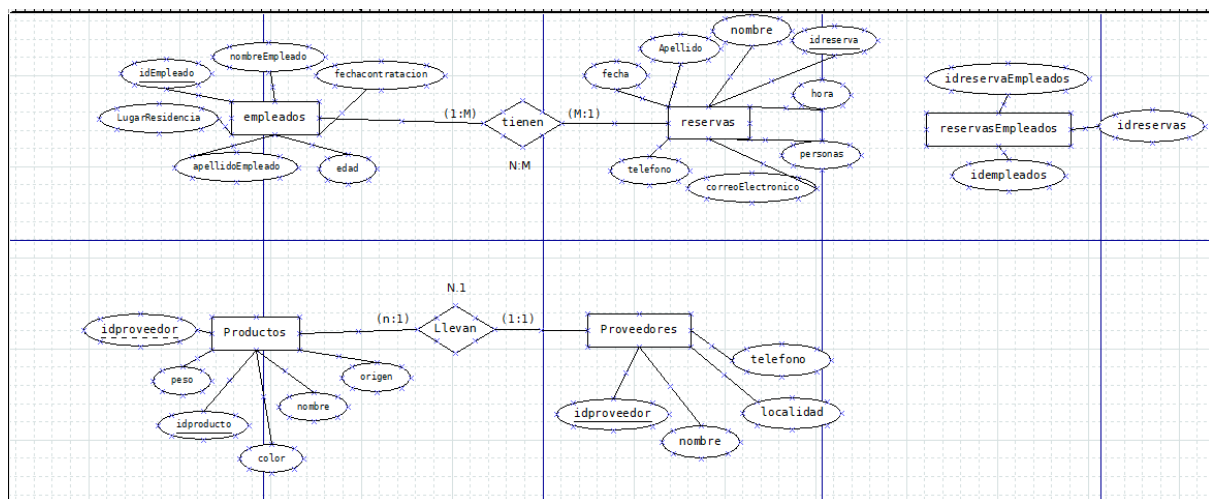
La parte de bases de datos ha sido una de las partes más complicadas del reto ya que debíamos realizar la conexión con el formulario de la página web y ambas cosas deben estar subidas en una máquina virtual (algo que no habíamos dado).

Proceso llevado a cabo:

Nuestro grupo decidió crear la máquina virtual en AWS ya que en las últimas semanas habíamos trabajado en ello y la base de datos decidimos realizarla con PHP porque vimos que la conexión con PHP era más sencilla.

Pero antes de hacer la conexión debíamos crear la base de datos. Empezamos creando el diagrama entidad-relación de nuestra base de datos para asignar las tablas que queríamos y sus relaciones, respetando las cardinalidades para que luego en el PHP no nos de error.

Al diagrama le acabamos asignando 4 tablas (Reservas, Empleados, Productos y Proveedores) de dichas tablas sacamos las relaciones como: N:M (es decir, debíamos crear una tabla más añadiendo los 2 id de dichas tablas) y 1:N. Y a cada tabla le asignamos sus atributos.



Más tarde, iniciamos XAMPP para poder acceder al PHPmyAdmin y desde ahí insertamos las tablas, atributos y por último las relaciones. Comprobamos que todo iba bien cuando en el formulario de la página indicamos la ip de la base de datos y que cuando la enviábamos vemos que la información se había guardado bien.

Tu Nombre:

Tu Apellido:




























Correo Electrónico:

Número de teléfono:

Número de personas

Hora de Reserva --:--

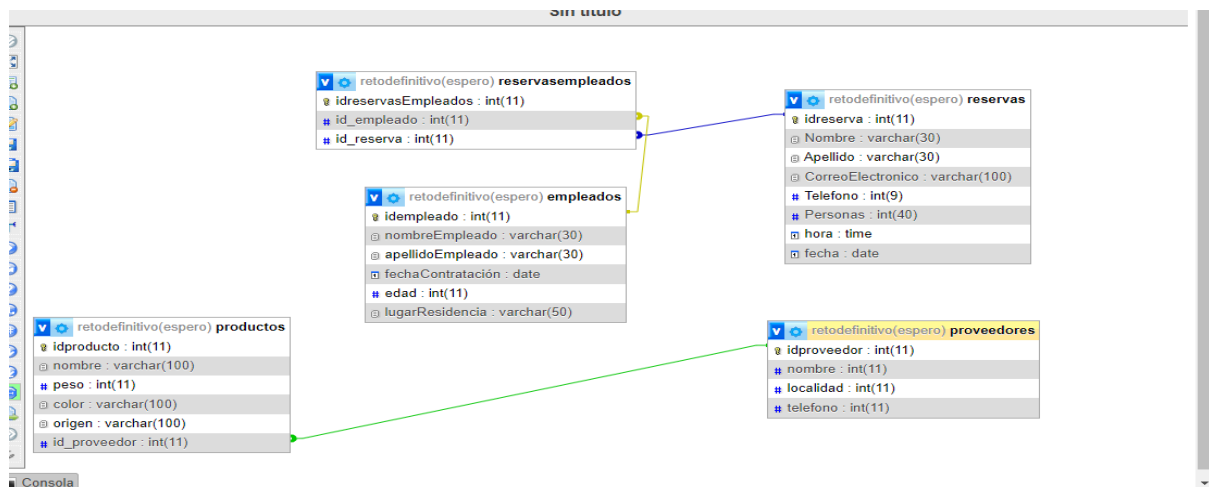
Fecha de reserva dd/mm/aaaa

<input type="checkbox"/>	 Editar	 Copiar	 Borrar	1	Alejandro	Rebollo	ar.alejandro01@gmail.com	987654321	13	16:43:00	2024-02-02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	2	Alejandro	Rebollo	ar.alejandro01@gmail.com	768768768	33	15:49:00	2024-01-08
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	3	Jon	Zorzano	jo.zorzano@aulanz.net	654982783	2	16:00:00	2024-02-14
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	4	jack	jhonson	jkaa@gmail.com	666666661	1	11:31:00	2024-01-31
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	5	Juan	Hernandez	juan@gmail.com	677777777	33	20:33:00	2024-02-14
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	6	Alejandross	Rebolloss	ar.alejandro01@gmail.com	987654321	13	16:43:00	2024-02-02
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	7	Alejandro	Rebollo	ale.rebollo@gmail.com	638255599	10	14:15:00	2024-05-22
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	8	Jon	Zorzano	jon.zorzano@gmail.com	638255599	20	21:00:00	2024-02-20
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	9	Aaron	Oliva	aar.oliva@gmail.com	638255599	30	14:00:00	2024-01-30
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	10	Julen	Lopez	jul.lopez@gmail.com	638255599	18	15:00:00	2024-03-31
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	11	Mario	Hernandez	mar.hernandez@gmail.com	638255599	12	20:00:00	2024-02-29
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	12	Maialen	Berasategui	mai.berasategui@gmail.com	638255599	2	21:30:00	2024-08-25
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	13	Julen	Salinas	jul.salinas@gmail.com	638255599	5	22:00:00	2024-05-05
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	14	Ivan	Boiev	iva.boiev@gmail.com	638255599	1	13:30:00	2024-09-09
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	15	Mireia	EliceGUI	mir.elicegui@gmail.com	638255599	14	14:45:00	2024-01-24
<input type="checkbox"/>	 Editar	 Copiar	 Borrar	16	Markel	Irastorza	mar.irastorza@gmail.com	638255599	13	20:30:00	2024-03-02

Registro insertado correctamente

Por último, en la siguiente imagen se mostrará las tablas y sus relaciones en el PHPmyadmin.Los datos que no pertenecen a la tabla de reservas han sido insertados

manualmente desde el PHP, sin embargo, los datos de la reserva han sido insertados desde el formulario.



Sin embargo, hubo complicaciones a la hora de conectar la base de datos con el programa Java y nos vimos obligados a realizar la base de datos en workbench. Para este proceso tuvimos que crear un servidor de base de datos en AWS (RDS), una vez creada la instancia se nos abre el programa de Workbench que tendrá un usuario, contraseña y la ip pública en programa. Por último, teníamos que exportar la base de datos de PHPmyAdmin al workbench para no tener que volver a insertar todos los datos.

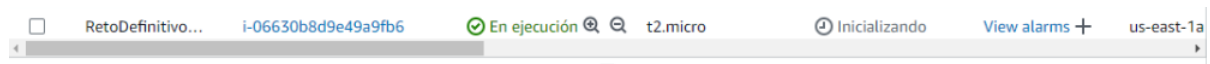
Sistemas informáticos

Introducción:

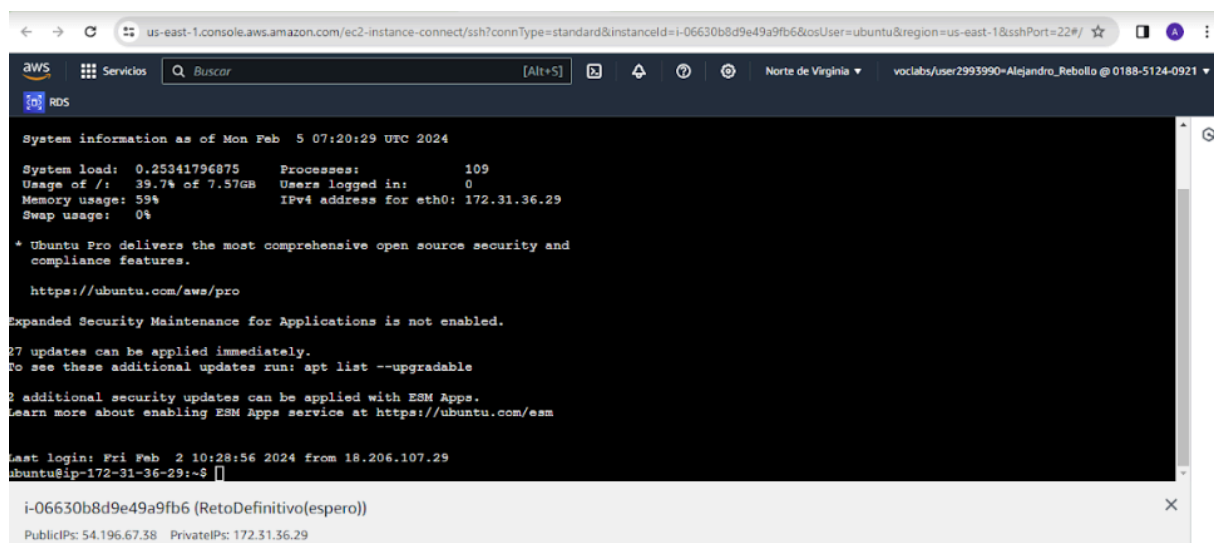
La parte que corresponde a sistemas es la manera que hemos conectado la página web y la base de datos al servidor. Pero primero explicaré cómo creamos la máquina virtual en AWS Academy ya que veníamos trabajando en ello las últimas semanas.

Proceso llevado a cabo:

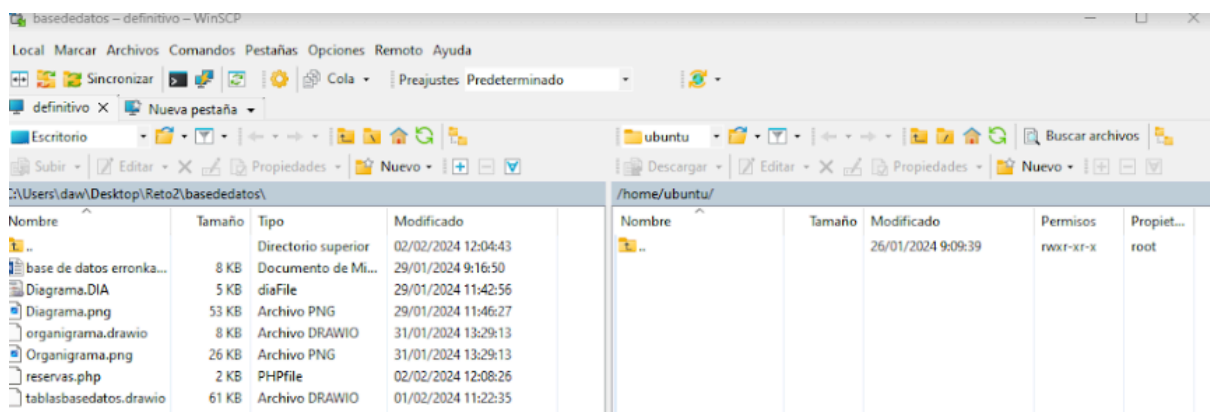
En AWS lanzamos una nueva instancia cuya aplicación de inicio sea UBUNTU, a su vez, pedimos una clave “PERM” para tener algo de seguridad y no pueda acceder fácilmente en el servidor. Posteriormente indicamos que tuviese acceso de tráfico de HTTPS para que se pueda acceder desde internet y no solo desde Localhost.



A partir de aquí ya tenemos creado el servidor y tenemos que realizar ahora los pasos para subir la página web al servidor. Para comenzar, debemos instalar en el servidor APACHE2 con el comando “sudo apt install apache2” con el objetivo de poder asignar las páginas de la página web. (al instalarlo se nos crea en /VAR las carpetas /WWW y /HTML en esta última debemos meter las páginas web cuya página principal se debe llamar index.html.



Sin embargo, no es tan fácil, debemos descargarnos WinSCP para poder subir los archivos al servidor para ello debemos indicar en dicho programa la ip publica del servidor e indicar el usuario y la clave. Una vez que hemos accedido, podemos subir los archivos que queramos al servidor y hacer los pasos dichos anteriormente.



Por otra parte, falta el proceso de instalación del PHP para poder subir la base de datos (ya creada) al servidor, nuestro grupo realizó dicha base de datos en PHP lo cual explicare de la manera para bases de PHP, la otra manera es con workbench.

Para subir la base de datos debemos instalar el PHP en el servidor de AWS siguiendo uno por uno los siguientes comandos: (importante hacer **SUDO -SU** para ser root)

Instalar el lenguaje de programación PHP

- **apt install php**

Instalar el servidor de bases de datos MySQL

- **apt install mysql-server**

Instalar el conector entre PHP y MySQL

- **apt install php-mysql**

Instalar phpMyAdmin

- **apt install phpmyadmin** (pide escoger servidor web => Apache, y establecer contraseña de acceso)

Una vez realizado estos pasos ya tenemos instalado lo necesario para subir la base de datos. Lo primero, debemos añadir el siguiente comando que nos proporcionará acceder a una base de datos PHP creado por el propio servidor, donde tendremos que indicar un usuario y una contraseña.

mysql -u user1 -p (Este comando nos abre el php)

- **create user 'user1'@'localhost' identified by 'pwuser1';**
- **create database db1;**
- **grant all on db1.* to 'user1'@'localhost';**

Cuando accedemos al PHP ya simplemente debemos exportar desde localhost de la página PHP al servidor.

Programación

Introducción:

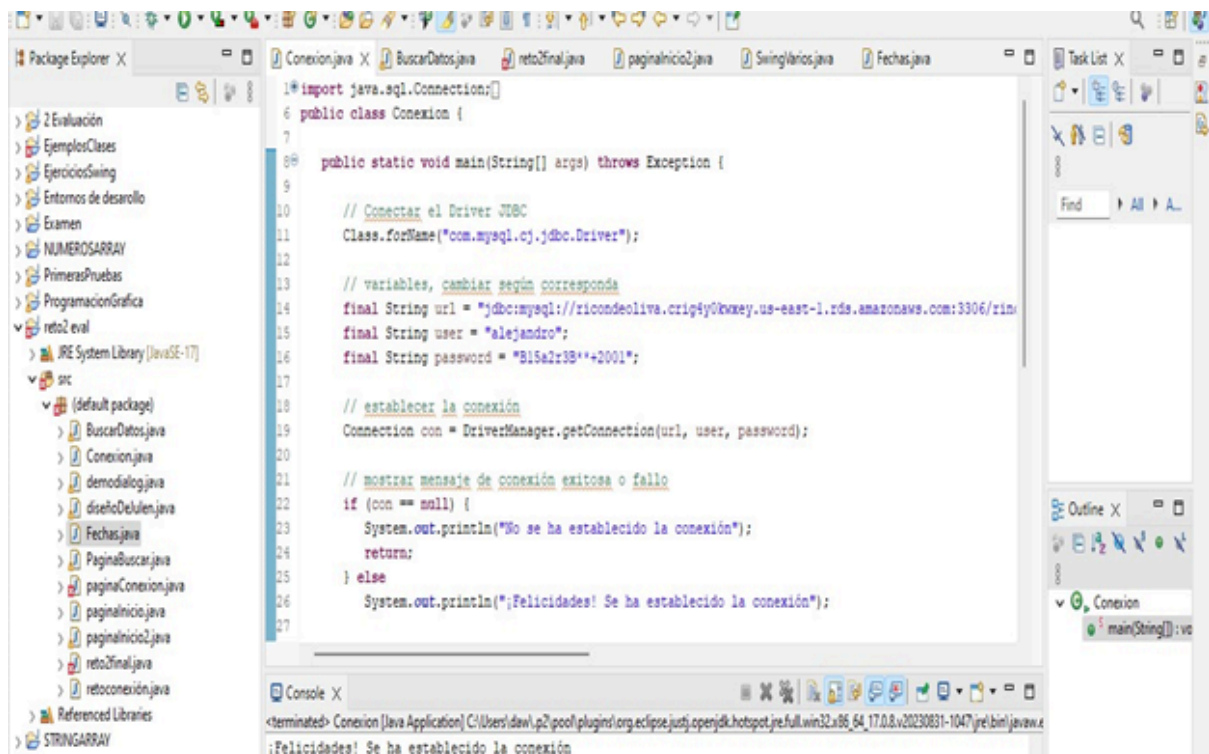
Para la parte de programación se nos pedía crear un proyecto Java donde debíamos crear un programa gráfico donde se muestre las reservas de la base de datos del restaurante para que de cara al empleado puedan gestionar el tema de las reservas. A su vez, debíamos indicar el programa con comentarios(JavaDoc).

Proceso llevado a cabo:

Página 1:

Nuestro trabajo se basa en la creación de 4 páginas Java, dichas páginas empezamos a desarrollar una vez que conseguimos subir la base de datos al servidor, en este punto, con el código que se nos proporcionó en el classroom creamos la primera página en la que el objetivo es que la conexión del programa Java con la base de datos es exitosa, donde en caso de que la conexión es buena se nos mostraría el siguiente mensaje:

“! Felicidades ¡Se ha establecido conexión”



```
1 import java.sql.Connection;
2
3 public class Conexion {
4
5     public static void main(String[] args) throws Exception {
6
7         // Conectar el Driver JDBC
8         Class.forName("com.mysql.cj.jdbc.Driver");
9
10        // variables, cambiar según corresponda
11        final String url = "jdbc:mysql://ricondeoliva.cripty0kxey.us-east-1.rds.amazonaws.com:3306/rin";
12        final String user = "alejandro";
13        final String password = "B15a2x3B**+2001";
14
15        // establecer la conexión
16        Connection con = DriverManager.getConnection(url, user, password);
17
18        // mostrar mensaje de conexión exitosa o fallo
19        if (con == null) {
20            System.out.println("No se ha establecido la conexión");
21            return;
22        } else {
23            System.out.println("¡Felicidades! Se ha establecido la conexión");
24        }
25    }
26 }
27
```

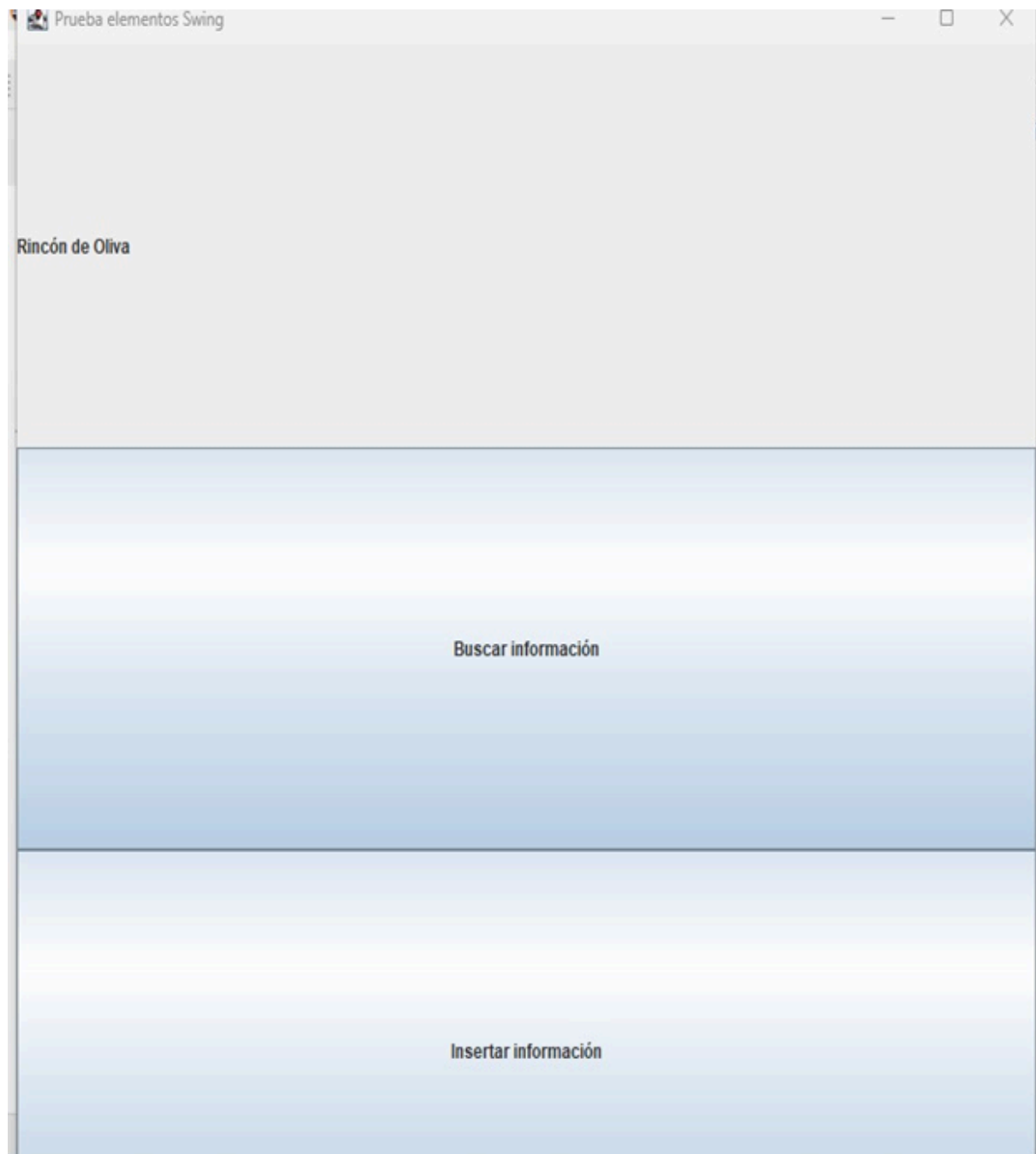
Console X

```
<terminated> Conexion [Java Application] C:\Users\daw1.p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.8.v20230831-1047\jre\bin\java.exe
¡Felicidades! Se ha establecido la conexión
```

Página 2:

Una vez comprobada que la conexión es exitosa nos reunimos para saber cómo sería nuestro programa Java. La idea que decidimos fue añadir 3 páginas más donde la página 1 sería la siguiente:

Una página donde le mostraría al empleado, aparte del nombre del restaurante, dos botones donde saldrían una frase en cada botón (“Buscar información” y “Insertar información”).



Esta página tiene una estructura sencilla, se basa en la creación de un JLabel donde se muestra el nombre del restaurante y de dos JButton para que nos de la opción de pulsar y que nos lleve a las otras páginas.

En el constructor hemos indicado que sea GridLayout, es decir, que nosotros indicamos donde queremos que vaya el JLabel y los dos JButton. Después indicamos

como queremos que sea la gráfica (Tamaño, sea visible...). Por último, indicamos con ADD en el orden que queremos que salga la información (en este caso no importa)

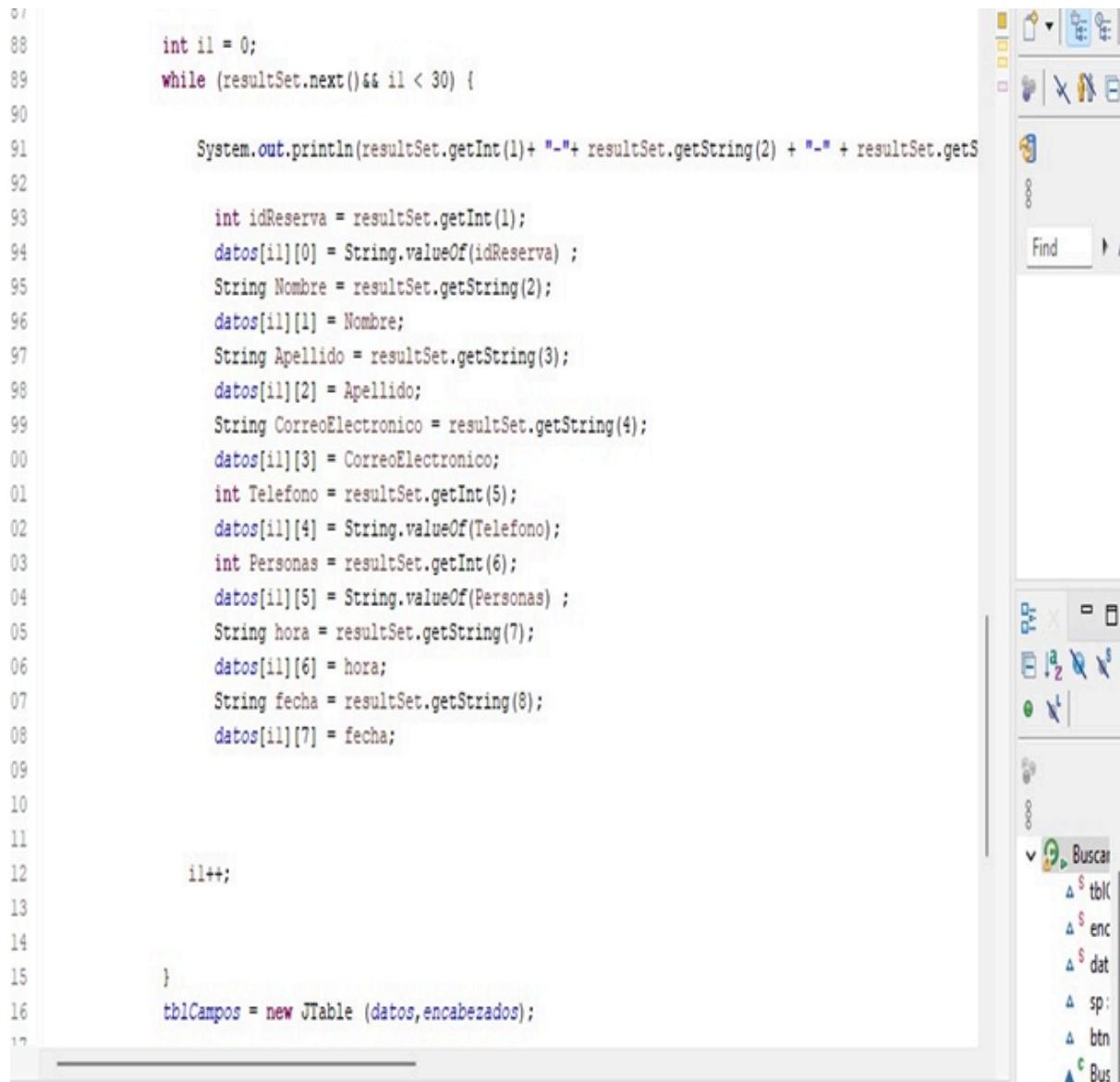
```
8      setLayout(new GridLayout(3,1));
9
10
11
12      btnbuscar = new JButton ("Buscar información");
13      btnenviar = new JButton ("Insertar información");
14
15      lblTitulo = new JLabel ("Rincón de Oliva") ;
16
17      btnenviar.addActionListener(this);
18      btnbuscar.addActionListener(this);
19
20
21      add(lblTitulo);
22      add(btnbuscar);
23      add(btnenviar);
24
25
26      setTitle("Prueba elementos Swing");
27      setSize(800,800);
28      setVisible(true);
29      setDefaultCloseOperation(EXIT_ON_CLOSE);
```

Página 2

La segunda página de Java se basa en que muestre en forma gráfica los datos de la base de datos del servidor.

El diseño es sencillo, hemos necesitado un JButton, JTable para indicar los campos, String[] para indicar la dimensión de los datos que se muestren. En el constructor hemos indicado que es FlowLayout y los demás elementos.

Sin embargo, la parte clave de la página está a la hora de la conexión ya que indicamos en el main el código de la conexión y tenemos que indicar los campos de que tipo son y qué campo tiene que coger de la base de datos.



```
87
88     int il = 0;
89     while (resultSet.next() && il < 30) {
90
91         System.out.println(resultSet.getInt(1) + "-" + resultSet.getString(2) + "-" + resultSet.getS
92
93         int idReserva = resultSet.getInt(1);
94         datos[il][0] = String.valueOf(idReserva) ;
95         String Nombre = resultSet.getString(2);
96         datos[il][1] = Nombre;
97         String Apellido = resultSet.getString(3);
98         datos[il][2] = Apellido;
99         String CorreoElectronico = resultSet.getString(4);
100        datos[il][3] = CorreoElectronico;
101        int Telefono = resultSet.getInt(5);
102        datos[il][4] = String.valueOf(Telefono);
103        int Personas = resultSet.getInt(6);
104        datos[il][5] = String.valueOf(Personas) ;
105        String hora = resultSet.getString(7);
106        datos[il][6] = hora;
107        String fecha = resultSet.getString(8);
108        datos[il][7] = fecha;
109
110
111
112        il++;
113
114
115    }
116    tblCampos = new JTable (datos, encabezados);
117
```

Página 3

En esta página hemos intentado insertar los datos, pero a falta de tiempo e información no hemos podido completar el código. Aunque, dejamos una captura del diseño donde se ve la estructura de la gráfica.

```

6
7 public class reto2final extends JFrame implements ActionListener {
8
9     JLabel lblnombre, lblapellido, lblcorreoelectronico, lbltelefono, lblpersonas, lblhora, lblfecha;
10    JTextField txtnombre, txtapellido, txtcorreoelectronico, txttelefono, txtpersonas, txthora, txtfecha;
11    JButton btnenviar;
12    static String datos [][]= new String [30][8];
13
14
15
16 reto2final () {
17
18     setLayout(new GridLayout(8,7));
19
20
21     lblnombre = new JLabel("Nombre");
22     lblapellido = new JLabel("Apellido");
23     lblcorreoelectronico = new JLabel("Correo Electronico");
24     lbltelefono = new JLabel("Telefono");
25     lblpersonas = new JLabel("Personas");
26     lblhora = new JLabel("Hora");
27     lblfecha = new JLabel("Fecha");
28
29
30     txtnombre = new JTextField();
31     txtapellido = new JTextField();
32     txtcorreoelectronico = new JTextField();
33     txttelefono = new JTextField();
34     txtpersonas = new JTextField();

```

Console X

Prueba elementos Swing

Nombre	
Apellido	
Correo Electronico	
Telefono	
Personas	
Hora	
Fecha	
<div>Enviar</div>	

Entornos de desarrollo:

Introducción:

Lo que se nos pedía era realizar el trabajo en conjunto mediante Git bash y añadir a los programas Java comentarios (JAVADOC).

Imágenes:

```
commit 079a2b4005c8e19474acc71f7d3cbb812db7c52a
Author: JonZorz <jz.zorzano@aulanz.net>
Date:   Wed Feb 7 08:55:04 2024 +0100

    Hemos añadido comentarios al xsd

commit 079a2b4005c8e19474acc71f7d3cbb812db7c52a
Author: JonZorz <jz.zorzano@aulanz.net>
Date:   Tue Feb 6 13:45:26 2024 +0100

    Inicio de los comentarios xsd

commit ffcfc863171f46c84d6004507631b621c243d1d8
Merge: feab481 1981f3c
Author: alejandrosalegui01 <al.rebollo@aulanz.net>
Date:   Tue Feb 6 11:53:49 2024 +0100

    Merge branch 'main' of https://github.com/JonZorz/Reto2

commit feab4810a270bb196078994700c507cc1d76f1a5
Author: alejandrosalegui01 <al.rebollo@aulanz.net>
Date:   Tue Feb 6 11:52:16 2024 +0100

    cambios en xml

commit 1981f3c3266bb4e5325dbd0df80b1ba68740bc3
Author: JonZorz <jz.zorzano@aulanz.net>
Date:   Tue Feb 6 09:25:48 2024 +0100

    hemos cambiado el organigrama

commit d7800703971ac7ed7359b202065161eef6f99fa0
Merge: 6460483 4e53eac
Author: julenlbi <ju.lopez@aulanz.net>
Date:   Mon Feb 5 11:44:31 2024 +0100

    dgh

commit 6460483f7eb7684bfff780110c157464b8a79f3ce
Author: julenlbi <ju.lopez@aulanz.net>
Date:   Mon Feb 5 11:37:21 2024 +0100

    relacionesDiagrama

commit 6247e5de55cb199135293e2357a155c3ee15f777
Author: julenlbi <ju.lopez@aulanz.net>
Date:   Mon Feb 5 11:34:51 2024 +0100
```

Javadoc:

También hemos añadido a los programas java comentarios javadoc para explicar lo que hemos hecho a lo largo de cada programa java.

Y después hemos comprobado mediante eclipse que genere un javadoc he aquí una prueba del resultado.

Class PaginaInicio2

```
java.lang.Object®
  java.awt.Component®
    java.awt.Container®
      java.awt.Window®
        java.awt.Frame®
          javax.swing.JFrame®
            PaginaInicio2
```

All Implemented Interfaces:

ActionListener[®], ImageObserver[®], MenuContainer[®], Serializable[®], EventListener[®], Accessible[®], RootPaneContainer[®], WindowConstants[®]

```
public class PaginaInicio2
  extends JFrame®
  implements ActionListener®
```

Aquí declaramos lo botones, los labels y los importamos

See Also:

Serialized Form

Mejoras en un futuro

Hemos tomado la decisión de crear un apartado dedicado a la formulación de ideas y objetivos futuros, fundamental para un posible avance en este proyecto. Esta adición no solo servirá como documento archivado, sino también como una guía organizada para facilitar el progreso coherente en caso de retomar el proyecto en el futuro. La idea es tener a mano un documento que facilite el proceso y nos sirva de ayuda en nuestro desarrollo.

Creemos que actualmente está bastante bien realizado el trabajo pero por el tiempo, se nos han quedado algunas ideas en el aire y que sí podríamos añadir en un futuro tanto como en mejorar la página web, mejorar la base de datos,...

Estas son algunas ideas que hemos tenido:

- ☐ Optimizar el programa Java (que la fecha actual salga automáticamente)

Este objetivo se refiere a que cuando los empleados quieran ver las reservas de ese mismo día no tengan que escribir dicha fecha, es decir, que la fecha salga de forma automática.

- ☐ Separar las base de datos (comidas y cenas)

Con este objetivo intentamos mantener un mayor orden en la base de datos al separar las comidas con las cenas.

- ☐ Ordenar las reservas, añadir en qué mesa se sitúa cada cliente

Con este objetivo queremos asignar un número a cada mesa con lo cual los empleados sepan a qué mesas atienden.

- ☐ Diseñar mejor el programa html

Simplemente queremos mejorar el diseño HTML de nuestra página ya que creemos que nuestra página resulta en cierta manera vulgar.

- ☐ Hacer un apartado especial para que la gente pueda poner sus fotos y reseñas

Añadir una nueva página donde los clientes puedan subir fotos dentro del establecimiento comiendo o cenando y que además puedan añadir una pequeña reseña explicando su experiencia.

- ☐ Mejorar la accesibilidad

Intentar optimizar la página lo máximo posible para que al cliente y empleado les resulte sencillo navegar tanto por la página como por el programa Java.

Fallos y complicaciones

Introducción:

En este apartado vamos a explicar todos los problemas que hemos tenido a lo largo de estas 3 semanas y de qué manera los hemos solucionado (no hemos podido resolver todos los problemas).

Problemas:

La conexión de phpmyadmin al servidor AWS, al insertar las páginas en el servidor solo nos quedaba añadir la base de datos, sin embargo, el servidor requería la instalación de varios paquetes de mysql y php, comandos que desconocíamos por completo.

Una vez consultado con el profesor nuestro problema nos proporcionó dichos comandos para la instalación de phpmyadmin y de esta manera conseguimos solucionar dicho retraso.

Otro problema que hemos tenido ha sido a la hora de intentar conectar el programa java con la base de datos situada en phpmyadmin. EL profesor nos proporcionó un código java para comprobar que la conexión era exitosa, al realizar todos los pasos que nos había indicado el profesor, sin embargo, no se efectuaba la conexión. Nos dimos cuenta gracias a una página de internet al añadir nuestra ip pública nos indicaba que nuestro puerto 3306 estaba cerrado, sin embargo, habíamos indicado en la instancia del AWS que permitiese todo el tráfico. Este problema se lo comentamos al profesor y no encontró donde estaba el error y que la única solución era hacerla en un servidor RDS.

Los pasos de la instalación del RDS ya han sido comentados en el apartado de bases de datos.

Otra complicación que tuvimos fue añadir el código de la conexión con la base de datos al programa gráfico de java, ya que no sabíamos donde teníamos que insertar dicho código en el programa.

Intentamos insertarla como una función, pero no lo conseguimos después lo intentamos añadir en el actionPerformed, sin embargo, el programa nos indicaba errores que no podíamos solucionarlos. Finalmente, gracias a un compañero de clase nos ayudó diciéndonos que lo debíamos poner en el main, a parte nos indicó que debíamos poner el actionPerformed.

Por último, el profesor nos indicó que era un buen detalle añadir un programa java para insertar datos desde java en la base de datos. Tras muchos intentos no conseguimos conectar con la base de datos ya que la estructura pensábamos que era similar a la de extraer datos de la base de datos, ni con las explicaciones del profesor, no conseguimos que el programa funcionase a pesar de entender cómo sería la estructura del programa.

Y a su vez, por la falta de tiempo y sabiendo que este programa no era obligatorio crearlo decidimos abandonarlo por ahora.