**Lenguaje Programación Java**

**Ingeniería en Sistemas de Información**

**Documentación Trabajo FINAL**

****

**Aplicación: “La Hora del Bondi”**

**Pagina web: [www.lahoradelbondi.com.ar](http://www.lahoradelbondi.com.ar)**



**Integrantes:** 35914 -Broin, Maximiliano

36099 - Sabino, Franco

Indice

[Indice 2](#_Toc324858983)

[Introducción 7](#_Toc324858984)

[Descripción del negocio 7](#_Toc324858985)

[Funcionalidades Principales de la aplicación 8](#_Toc324858986)

[Descripción de Casos de Uso Resumen con Alcance de Sistema (Solo casos de uso principales) 8](#_Toc324858987)

[Lenguajes utilizados en la aplicación 15](#_Toc324859093)

[Arquitectura de la aplicación 17](#_Toc324859094)

[Base de datos de la Aplicación 23](#_Toc324859095)

[Fragmentos más importantes de código de la aplicación 29](#_Toc324859106)

[Capturas de pantallas 37](#_Toc324859107)

[Conclusión 42](#_Toc324859108)

[Información utilizada para realizar aplicación 43](#_Toc324859109)

# 

# Introducción

Como finalización de la asignatura LENGUAJE DE PROGRAMACION JAVA, se ha desarrollado una aplicación Web que nos servirá como búsqueda de horarios de colectivos entre distintas ciudades. La aplicación contendrá información de varias empresas de corta, mediana y larga distancia y a partir de ello podrán utilizarlo muchos usuarios que estén interesados en consultar horarios de algún recorrido en particular.

# Descripción del negocio

Partiendo de la necesidad nuestra de saber los horarios de los colectivos hacia “nuestras localidades”, decidimos construir una aplicación Web llamada la “Hora del Bondi” en la cual estarán cargados los diferentes horarios de partida y de llegada entre las localidades de las empresas que están cargadas en nuestra aplicación. Por el momento existen dos empresas pero esto puede expandirse a mayores números para así cubrir necesidades de usuarios que estén interesados en consultar sus horarios.

La función principal que la aplicación brinda a los usuarios es que podrán seleccionar su empresa de colectivos (corta, mediana o larga distancia), luego seleccionar la ciudad origen y destino a la que quieren viajar, una fecha determinada de viaje y un horario aproximado en la cual la persona quiere realizar el viaje. La búsqueda de “La hora del Bondi” mostrara aquellos horarios (si es que existen) entre esas dos localidades, teniendo en cuenta la fecha seleccionada y filtrando los horarios en un rango de dos horas antes de la hora elegida y dos horas después de la hora elegida.

También los usuarios podrán ver todos los horarios que realiza la empresa elegida entre la cuidad origen y la cuidad destino, ese día seleccionado sin elegir una determinada hora deseada del viaje.

Cabe recordar que al tener que elegir una fecha del viaje, los horarios entre las ciudades orígenes y destinos serán diferentes de acuerdo a los distintos días, por ejemplo: Los recorridos y horarios que realiza la empresa “Güemes” son diferentes de acuerdo a si el día es sábado, domingo, feriado, etc.

Es una aplicación en la cual su funcionalidad principal será la consulta de horarios, aunque presenta otras funcionalidades.

# Funcionalidades Principales de la aplicación

* Mostrar los horarios de colectivos como resultado de la búsqueda de un usuario que selecciona la empresa, ciudad origen, ciudad destino, fecha viaje y hora deseada del viaje.
* Mostrar todos los horarios de colectivos “del día” como resultado de la búsqueda de un usuario que selecciona la empresa, ciudad origen, ciudad destino y fecha viaje, sin elegir un horario deseado del viaje.
* Filtra, de acuerdo a una fecha elegida por el usuario, los horarios de los recorridos de las ciudades elegidas, teniendo en cuenta que los mismos pueden variar de acuerdo a si el día es por ejemplo “Domingo, Feriado, Día de semana, etc”. La aplicación realiza la validación necesaria para cumplir con estos requerimientos.
* Mostrar los precios “Normales” y de “Descuento” que tiene el viaje de acuerdo a la ciudad origen y destino seleccionados.
* Posibilidad de registrarse como nuevo usuario a la aplicación.
* Siendo usuario de la aplicación, además de poder modificar algunos de tus datos personales, podes enviar un comentario, queja, problema, a través de un formulario de contacto que van a ser respondidos por los administradores de la aplicación. (Con el correr del tiempo se agregaran mas funcionalidades al usuario)
* Existen los llamados “Administradores de la aplicación” que tienen las siguientes funcionalidades:
  + Modificar los precios de los recorridos de las ciudades de acuerdo a la empresa elegida. Los precios varían muy rápidamente por lo cual con esta funcionalidad los administradores pueden actualizar los precios.
  + Modificar los feriados trasladables (Ejemplo: “feriados puentes”) ya que si la fecha elegida es un feriado cambiaran los recorridos y los horarios de las empresas de viajes.
  + Mostar, a los administradores, la cantidad de mensajes no leídos de aquellos usuarios que mandaron completando el formulario de contacto.
  + Modificar el/los datos de los administradores.

# Descripción de Casos de Uso Resumen con Alcance de Sistema (Solo casos de uso principales)

1. R1 - “Buscar Horarios con restricción de hora”.
2. R2 - “Ver todos los horarios del día”.
3. R3 - “Registrase como nuevo usuario”.
4. R4 – “Enviar formulario de contacto”.
5. R5 – “Modificar precios de recorridos”.
6. CODIGO Y NOMBRE DE CASO DE USO: R1 - “Buscar Horarios con restricción de hora”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Sin-Estructurar | Sistema | Negra | Real | Semántica |

Meta del CASO DE USO: Obtener los horarios entre ciudad origen y ciudad destino seleccionando una hora deseada del viaje.

**ACTORES**

**Primario:** Usuario **Otros:-**

**PRECONDICIONES (de negocio):-**

**PRECONDICIONES (de sistema)**

**Primarias:** Usuario ingresa a dirección web “www.lahoradelbondi.com.ar”

**Complementarias:-**

**DISPARADOR: -**

**FLUJO DE SUCESOS**

**CAMINO BÁSICO:**

1. El Usuario selecciona la “empresa” de colectivos en la cual desea saber los horarios.
2. El Usuario selecciona “ciudad origen” del recorrido.
3. El Usuario selecciona “ciudad destino” del recorrido.
4. El usuario presiona campo de “Fecha del Viaje”. El sistema muestra calendario. Usuario selecciona la fecha en el mismo.
5. Usuario selecciona “hora” y “minutos” en los cual desea viajar. Presiona botón “Buscar”.
6. Sistema muestra horarios según datos seleccionados anteriormente por el usuario. Muestra además el precio el viaje entre esas dos localidades.

**CAMINOS ALTERNATIVOS:**

**[1..5]. a.: Usuario deja sin seleccionar alguno de los datos entre el punto 1 y 5.**

[1..5] .a.1. Sistema muestra error informando “Complete todos los campos antes de realizar la búsqueda”.

**6. a.: No existen horarios según los requerimientos seleccionados por el usuario.**

6. a.1. Sistema informa situación indicando “No existen horarios que cumplan con las condiciones establecidas”.

**POSTCONDICIONES (de sistema)**

**Éxito:** Sistema muestra al usuario horarios según requerimientos establecidos por el usuario.

**Fracaso:** Sistema informa que se deben completar todos los campos para realizar la búsqueda.

**Éxito alternativo:** Sistema no encuentra horarios entre los requerimientos establecidos por el usuario e informa situación.

**POSTCONDICIONES (de negocio)**

**Éxito: -**

**Fracaso:-**

**Éxito alternativo:-**

1. CODIGO Y NOMBRE DE CASO DE USO: R2 - “Ver todos los horarios del día”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Sin-Estructurar | Sistema | Negra | Real | Semántica |

Meta del CASO DE USO: Obtener todos los horarios de una fecha entre ciudad origen y ciudad destino.

**ACTORES**

**Primario:** Usuario **Otros:-**

**PRECONDICIONES (de negocio):** Usuario realizo una búsqueda seleccionando todos los requerimientos y además estableció hora deseada del viaje. Sistema mostro todos los horarios según lo seleccionado por el usuario.

**PRECONDICIONES (de sistema)**

**Primarias:** Sistema muestra resultado de CU R1.

**Complementarias:-**

**DISPARADOR: -**

**FLUJO DE SUCESOS**

**CAMINO BÁSICO:**

1. El Usuario presiona botón “Ver todos los horarios del día”.
2. Sistema muestra todos los horarios de la fecha entre esa localidad origen y destino.

**CAMINOS ALTERNATIVOS:**

**2. a.: No existen horarios según los requerimientos seleccionados por el usuario.**

2. a.1. Sistema informa situación indicando “No existen horarios que cumplan con las condiciones establecidas”.

**POSTCONDICIONES (de sistema)**

**Éxito:** Sistema muestra al usuario todos los horarios de una fecha.

**Fracaso:** -.

**Éxito alternativo:** Sistema no encuentra horarios entre los requerimientos establecidos por el usuario e informa situación.

**POSTCONDICIONES (de negocio)**

**Éxito: -**

**Fracaso:-**

**Éxito alternativo:-**

1. CODIGO Y NOMBRE DE CASO DE USO: R3 - “Registrase como nuevo usuario”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Sin-Estructurar | Sistema | Negra | Real | Semántica |

Meta del CASO DE USO: Registrase exitosamente como nuevo usuario.

**ACTORES**

**Primario:** Usuario **Otros:-**

**PRECONDICIONES (de negocio):-.**

**PRECONDICIONES (de sistema)**

**Primarias**:**-.**

**Complementarias:-**

**DISPARADOR: -**

**FLUJO DE SUCESOS**

**CAMINO BÁSICO:**

1. El Usuario presiona botón “Registrarse”.
2. Sistema muestra formulario a completar para finalizar la operación de registración a la aplicación.
3. Usuario completa el campo “Nombre de usuario”.
4. Usuario completa el campo “Apellido”.
5. Usuario completa el campo “Nombre”.
6. Usuario completa el campo “Teléfono”.
7. Usuario completa el campo “Mail”.
8. Usuario completa el campo “Contraseña”.
9. Usuario completa el campo “Repetir Contraseña”.
10. Usuario presiona botón “Crear Cuenta”. Sistema informa que se completo la operación exitosamente y dirige al usuario a la pantalla principal.

**CAMINOS ALTERNATIVOS:**

**3. a.: Nombre de usuario existente en la aplicación.**

3. a.1. Sistema informa situación: “Nombre de usuario existente”. Vuelve al paso 3.

**3. b.: Caracteres incorrectos escritos en campo Nombre de usuario.**

3. b.1. Sistema informa situación: “Solo letras y números”. Vuelve al paso 3.

**3. c.: Longitud incorrecta de Nombre de Usuario.**

3. c.1. Sistema informa situación: “Entre 2 - 25 caracteres”. Vuelve al paso 3.

**4. a.: Caracteres incorrectos escritos en campo Apellido.**

4. a.1. Sistema informa situación: “Primer letra mayúscula - solo letras”. Vuelve al paso 4.

**4. b.: Longitud incorrecta de Apellido.**

4. b.1. Sistema informa situación: “Entre 2 - 25 caracteres”. Vuelve al paso 4.

**5. a.: Caracteres incorrectos escritos en campo Nombre.**

5. a.1. Sistema informa situación: “Primer letra mayúscula - solo letras”. Vuelve al paso 5.

**5. b.: Longitud incorrecta de Nombre.**

5. b.1. Sistema informa situación: “Entre 2 - 25 caracteres”. Vuelve al paso 5.

**6. a.: Caracteres incorrectos escritos en campo Teléfono.**

6. a.1. Sistema informa situación: “Formato: Carac - Número”. Vuelve al paso 6.

**6. b.: Longitud incorrecta de Teléfono.**

6. b.1. Sistema informa situación: “Entre 2 - 25 caracteres”. Vuelve al paso 6.

**7. a.: Caracteres incorrectos escritos en campo Mail.**

7. a.1. Sistema informa situación: “Formato: tumail@empresa.com”. Vuelve al paso 7.

**7. b.: Longitud incorrecta de Mail.**

7. b.1. Sistema informa situación: “Entre 2 - 35 caracteres”. Vuelve al paso 7.

**8. a.: Caracteres incorrectos escritos en campo Contraseña.**

8. a.1. Sistema informa situación: “Solo letras y números”. Vuelve al paso 8.

**8. b.: Longitud incorrecta de Contraseña.**

8. b.1. Sistema informa situación: “Entre 7 - 20 caracteres”. Vuelve al paso 8.

**9. a.: Campo Contraseña y campo Repetir Contraseña no coinciden.**

9. a.1. Sistema informa situación: “Las contraseñas deben coincidir”. Vuelve al paso 9.

**POSTCONDICIONES (de sistema)**

**Éxito:** Sistema dirige al usuario a la página principal de la aplicación y registra en base de datos el nuevo usuario. Muestra confirmación de la registración al Usuario.

**Fracaso: -**.

**Éxito alternativo: -.**

**POSTCONDICIONES (de negocio)**

**Éxito:** Usuario se registra en la aplicación web y recibe mensaje de confirmación por parte del sistema**.**

**Fracaso:** Usuario no logra registrase como usuario nuevo.

**Éxito alternativo:-**

1. CODIGO Y NOMBRE DE CASO DE USO: R4 – “Enviar formulario de contacto”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Sin-Estructurar | Sistema | Negra | Real | Semántica |

Meta del CASO DE USO: Enviar exitosamente el mensaje establecido en el formulario de contacto.

**ACTORES**

**Primario:** Usuario **Otros:-**

**PRECONDICIONES (de negocio):** Usuario logeado e iniciado sesión en la aplicación.

**PRECONDICIONES (de sistema)**

**Primarias**:Sesión iniciada por usuario**.**

**Complementarias:-**

**DISPARADOR:** Usuario presiona botón “Contáctenos” del panel de “Usuarios”.

**FLUJO DE SUCESOS**

**CAMINO BÁSICO:**

1. El Usuario completa campo “Mensaje”.
2. Usuario presiona botón “Enviar”. Sistema informa que el mensaje se envió exitosamente.

**CAMINOS ALTERNATIVOS:**

**1. a.: Campo Mensaje vacio.**

1. a.1. Sistema informa situación: “Campo Mensaje esta vacio”. Vuelve al paso 1.

**1. b.: Longitud de campo Mensaje mayor a 400 caracteres.**

1. b.1. Sistema informa situación: “Solo se pueden escribir 400 caracteres”. Vuelve al paso 1.

**POSTCONDICIONES (de sistema)**

**Éxito:** Sistema dirige al usuario a la página principal de la aplicación y registra mensaje enviado por el Usuario.

**Fracaso: -**.

**Éxito alternativo: -.**

**POSTCONDICIONES (de negocio)**

**Éxito:** Usuario envía mensaje exitosamente y recibe informe de confirmación del sistema.

**Fracaso:** Usuario no logra enviar mensaje.

Éxito alternativo:-.

1. CODIGO Y NOMBRE DE CASO DE USO: R5 – “Modificar precios de recorridos”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Sin-Estructurar | Sistema | Negra | Real | Semántica |

Meta del CASO DE USO: Modificar exitosamente el precio del recorrido entre 2 ciudades.

**ACTORES**

**Primario:** Administrador **Otros:-**

**PRECONDICIONES (de negocio):** Administrador logeado e iniciado sesión en la aplicación.

**PRECONDICIONES (de sistema)**

**Primarias**:Sesión iniciada por administrador**.**

**Complementarias:-**

**DISPARADOR:** Usuario presiona botón “Precios” del panel de “Administrador”.

**FLUJO DE SUCESOS**

**CAMINO BÁSICO:**

1. El Administrador selecciona la “empresa” de colectivos en la cual desea saber los horarios.
2. El Administrador selecciona “ciudad origen” del recorrido.
3. El Administrador selecciona “ciudad destino” del recorrido.
4. Sistema muestra “Precio Normal” y “Precio Descuento” anterior según lo seleccionado por el administrador.
5. El Administrador completa la “Precio Actual” para el recorrido establecido.
6. El Administrador presiona botón “Modificar Precio”. Sistema informa que la operación se completo exitosamente.

**CAMINOS ALTERNATIVOS:**

**[1..3]. a.: Administrador deja sin seleccionar alguno de los datos entre el punto 1 y 3.**

[1..3] .a.1. Sistema muestra error informando “Complete todos los campos antes de realizar la modificación”.

**4. a.: Sistema muestra “Precio Normal” y “Precio Descuento” sin cargar**.

4. a.1. Sistema no encuentra precios cargados en base de datos e informa situación indicando “Sin cargar”.

**5. a.: Caracteres incorrectos en campo “Precio Actual”.**

6. a.1. Sistema informa situación indicando “Solo acepta números con o sin punto”. Vuelve al paso 5.

**5. b.: Campo “Precio Actual” en blanco.**

5. b.1. Sistema informa situación indicando “Precio Normal sin modificar”. Vuelve al paso 5.

**5. c. <posterior><opcional>: Administrador completa campo “Precio Descuento Actual”.**

5. c.1. Sistema valida campo “Precio Descuento Actual”.

5. c.2. **Caracteres incorrectos en campo “Precio Descuento Actual”.**

5. c.2.a: Sistema informa situación indicando “Solo acepta números con o sin punto”. Vuelve al paso 5c.

**POSTCONDICIONES (de sistema)**

**Éxito:** Sistema registra modificación en el precio del recorrido establecido.

**Fracaso: -**.

**Éxito alternativo: -.**

**POSTCONDICIONES (de negocio)**

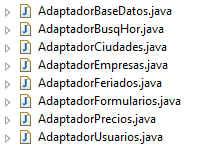
**Éxito:** Administrador recibe informa de modificación por parte del sistema**.**

**Fracaso:** Administrador no logra modificar el precio del recorrido establecido.

**Éxito alternativo:-.**

# Lenguajes utilizados en la aplicación

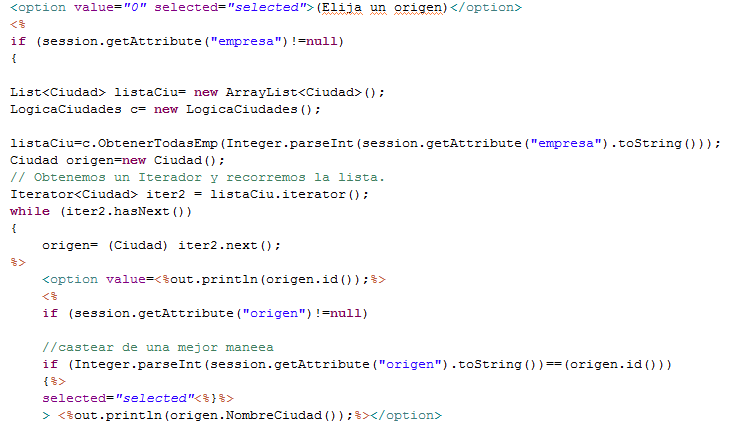
* **Clases Java (.java)**



El proyecto está compuesto por 34 clases java dentro de las cuales posteriormente explicacaremos su estructura y veremos un poco su contenido-

* **Jsp (Java Server Pages .jsp)**

En este fragmento de código mostramos una parte de un JSP donde utilizamos lenguaje HTML incluyendo dentro de él lenguaje JAVA. El proyecto está compuesto de 14 JSP.



* **Lenguaje HTML.**

Para realizar la parte visual de nuestra aplicación utilizamos lenguaje HTML como el fragmento de código mostrado en el siguiente ítem.



* **Scripts de Javascript.**

El script que utilizamos fue para mostrar un calendario del cual la persona interesado en consultar los horarios selecciona la fecha en la que desea viajar.



Script de Javascript:

Llamada desde la página index.jsp al script “popCalendar.js”:



* **Hojas de Estilo (.css)**

Para mejorar la estética de la aplicación utilizamos hojas de estilo externas a los JSP, en los cuales dentro de los mismos establecemos las llamadas a cada una de las CSS.



Ejemplo de una llamada a una hoja de estilo:



# Arquitectura de la aplicación

La **programación por capas** es una arquitectura [cliente-servidor](http://es.wikipedia.org/wiki/Cliente-servidor) en el que el objetivo primordial es la separación de la lógica de negocios de la lógica de diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.

La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado.

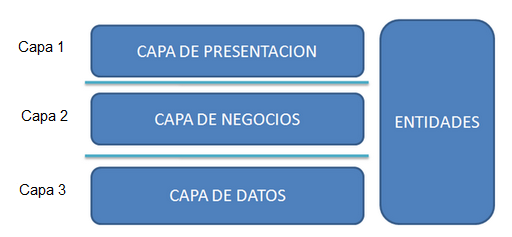
Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente [abstraído](http://es.wikipedia.org/wiki/Abstracci%C3%B3n_(programaci%C3%B3n_orientada_a_objetos)) del resto de niveles.

En esta arquitectura a cada nivel se le confía una misión simple, lo que permite el diseño de una arquitectura escalable (que pueden ampliarse con facilidad en caso de que las necesidades aumenten).

Ventajas de utilizar el modelo en 3 capas:

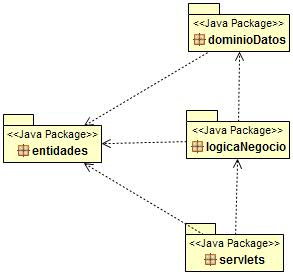
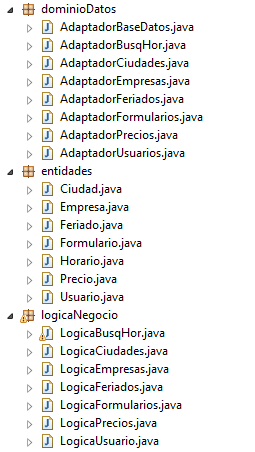
* Reutilización de capas.
* Contención de cambios a una o pocas capas.
* Darle seguridad y versatilidad al sistema.
* Es más ordenado.
* Clara distribución de las responsabilidades.
* Podemos cambiar de repositorio de datos con un bajo impacto en el resto de la aplicación.
* Podemos cambiar la interfaz más fácilmente que con otro modelo.

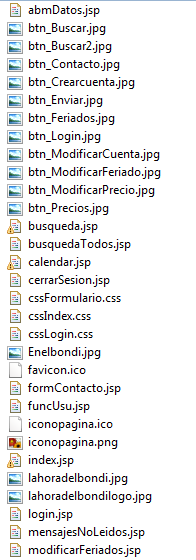
Nuestro proyecto está desarrollado en 3 capas y la forma la presentamos en el siguiente imagen:

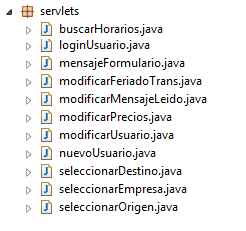


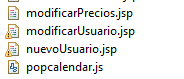
Cada una de estas capas está representada en nuestro proyecto con diferentes paquetes:

* Capa de datos 🡪 Paquete dominioDatos.
* Capa de negocios 🡪 Paquete logicaNegocios.
* Capa de presentación 🡪 Paquete servlets y archivos .jsp
* Luego tendremos un paquete dentro de nuestro proyecto llamado “entidades” que es aquel que contendrá todos los objetos que se irán comunicando de capa en capa.









**Capa de presentación:** es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio.

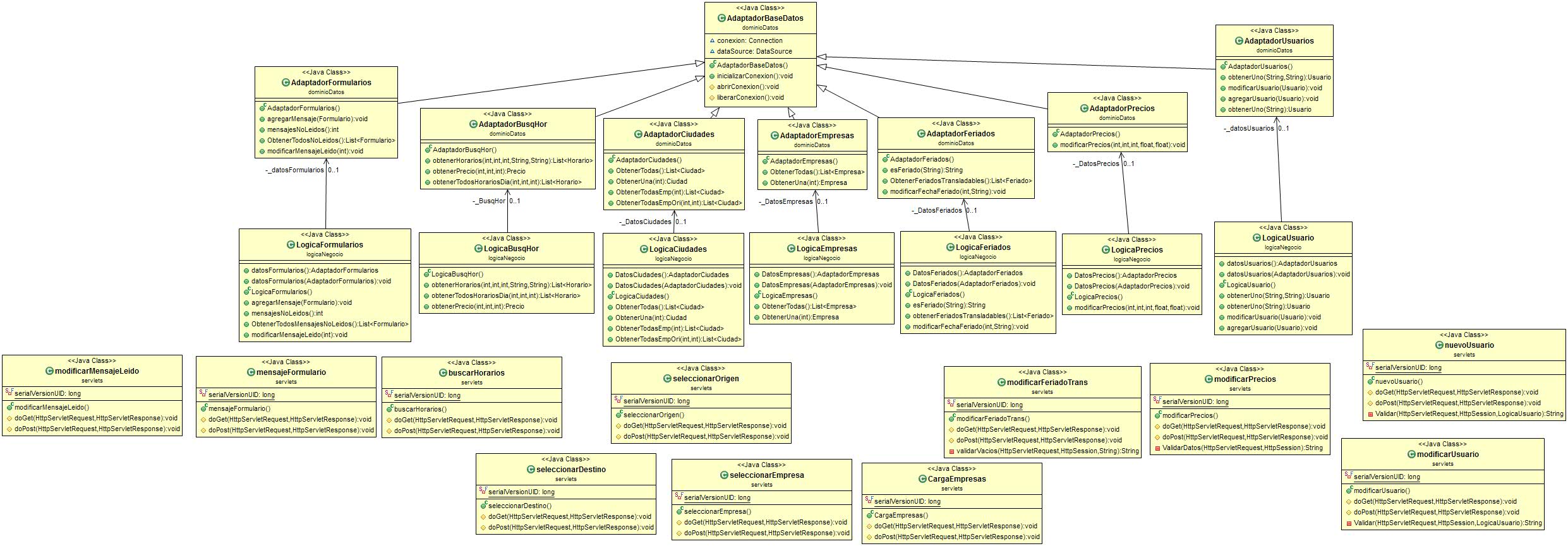
* Funciones
  + Recoger la información del Usuario.
  + Enviar esta información a la Capa de Negocios.
  + Recoger resultados de la Capa de Negocios
  + Presentar los resultados al usuario.
  + Realizar validaciones del lado del cliente.

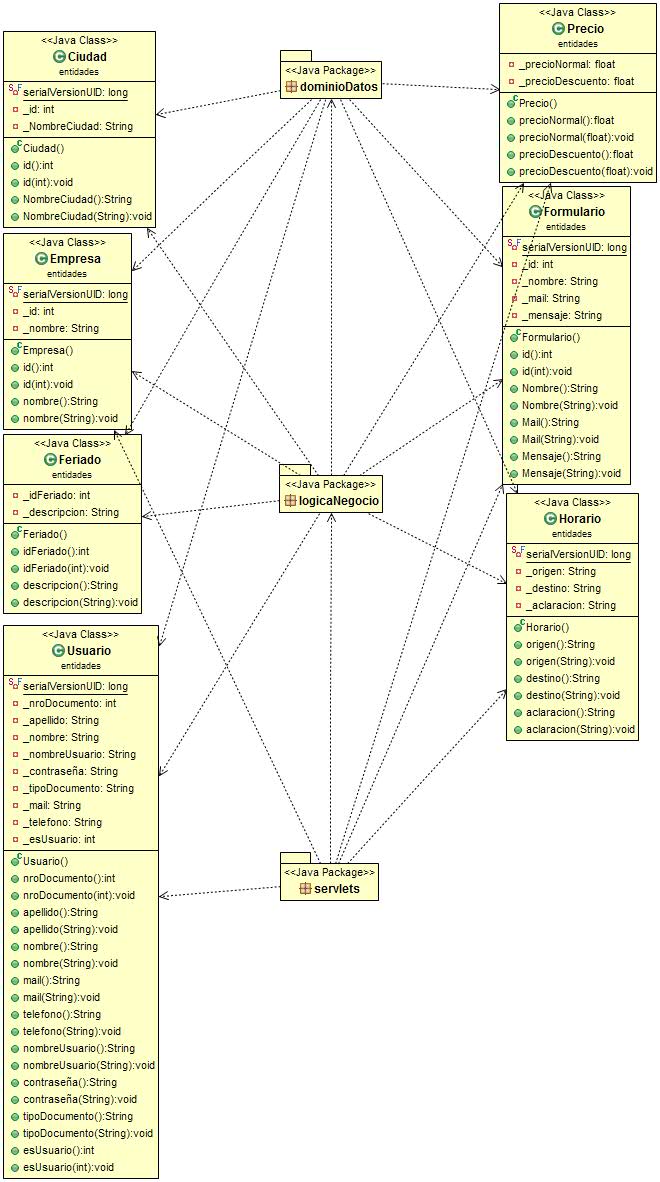
**Capa de negocio:** es donde residen los [programas](http://es.wikipedia.org/wiki/Programa_(computaci%C3%B3n)) que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de [base de datos](http://es.wikipedia.org/wiki/Base_de_datos) almacenar o recuperar datos de él.

* Funciones
  + Recibir la información de la capa de presentación
  + Interactuar con los servicios de datos para realizar la lógica de negocio y de la aplicación.
  + Enviar resultados a la capa de presentación

**Capa de datos:** es donde residen los datos y es la encargada de acceder a los mismos. Está formada por un gestor de bases de datos que realiza todo el almacenamiento de datos (MySql) y reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

* Funciones
  + Almacenar Datos
  + Recibir datos
  + Mantenimientos de los datos
  + Integridad de los datos

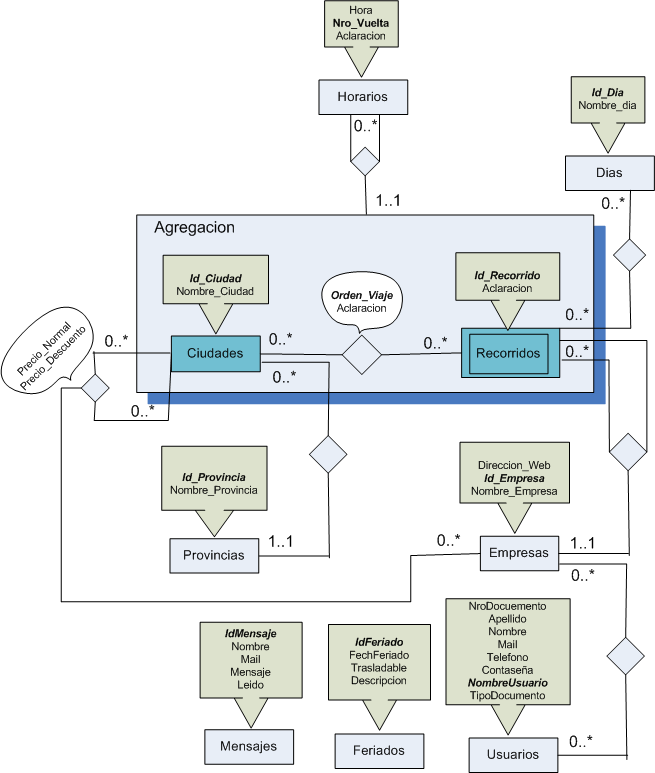




# Base de datos de la Aplicación

La aplicación está conectada a una base de datos MySql donde existen 13 tablas, las cuales guardan toda la información en registros para luego realizar las diferentes operaciones con los mismos.

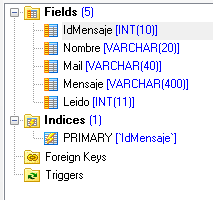
*Mostramos:*

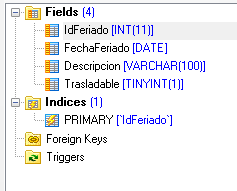
* Diagrama Entidad-Relación de la base de datos
* Pasaje a tablas y estructura de cada una de las tablas.

*Diagrama Entidad-Relación Base de datos HORARIOS\_COLECTIVOS*

*Pasaje a tablas del Modelo Entidad Relación y estructura de cada una de las tablas*

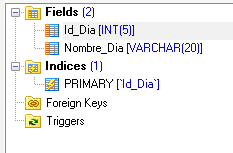
* Mensajes (**IdMensaje**, Nombre, Mail, Mensaje, Leido)



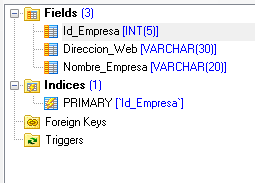
* Feriados (**IdFeriado**, FechaFeriado, Descipcion, Trasladable)
* Provincias (**Id\_Provincia**, Nombre\_Provincia)

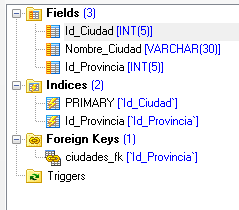


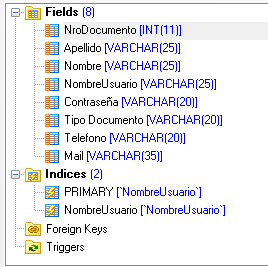
* Dias (**Id\_Dia**, Nombre\_Dia)



* Empresas (**Id\_Empresa**, Direccion\_Web, Nombre\_Empresa)



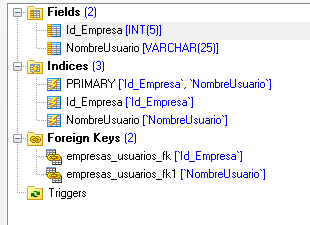
* Ciudades (**Id\_Ciudad**, Nombre\_Ciudad, Id\_Provincia (fk))
* Usuarios (**NombreUsuario**, Apellido, Nombre, Mail, Telefono, NroDocumento, TipoDocumento, Contraseña)



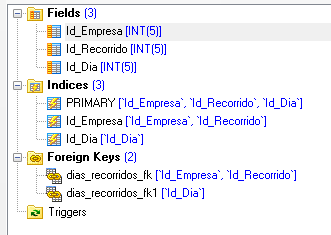
* Recorridos (**Id\_Empresa, Id\_Recorrido**, Aclaracion)



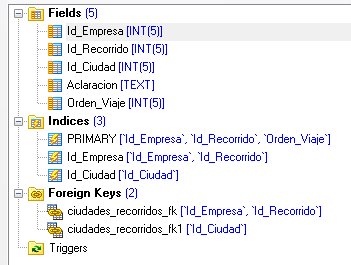
* Empresas\_Usuarios (**Id\_Empresa, NombreUsuario**)



* Dias\_Recorridos (**Id\_Empresa, Id\_Recorrido, Id\_Dia**)

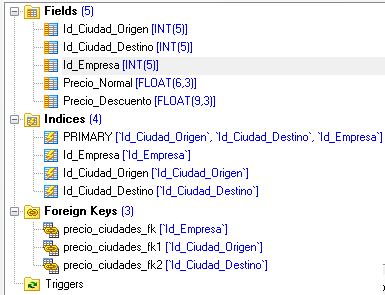


* Ciudades\_Recorridos (**Id\_Empresa, Id\_Recorrido, Orden\_Viaje**, Id\_Ciudad**,** Aclaracion)



* Horarios\_Ciudad\_Recorridos (**Id\_Empresa, Id\_Recorrido, Orden\_Viaje, Nro\_Vuelta**, Hora, Aclaracion)



* Precio\_Ciudades (**Id\_Ciudad\_Origen, Id\_Ciudad\_Destino, Id\_Empresa**, Precio\_Normal, Precio\_Descuento)

# Fragmentos más importantes de código de la aplicación

* *Conexión a la base de datos MySql*

La conexión a la base de datos MySql se realiza como ya hemos dicho a través de la capa de datos y en ella establecimos una clase llamada AdaptadorBaseDatos que tendrá la particularidad de establecer la inicialización de un pool de conexiones, además con esta clase se abre la conexión y luego se libera si la misma no es utilizada. Las demás clases de la capa de datos heredaran de esta clase AdaptadorBaseDatos.

*¿Qué es un pool de conexiones?*

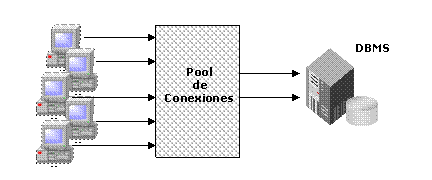
Es el manejo de una colección de [conexiones](http://es.wikipedia.org/wiki/Conexi%C3%B3n) abiertas a una [base de datos](http://es.wikipedia.org/wiki/Base_de_datos) de manera que puedan ser reutilizadas al realizar múltiples consultas o actualizaciones. Este pool es manejado por un servidor de aplicaciones de forma tal, que dichas conexiones pueden ser reutilizadas por los diferentes usuarios. También es administrado por un servidor de aplicaciones que va asignando las conexiones a medida que los clientes van solicitando consultas o actualizaciones de datos.

Cada vez que un programa cliente necesita comunicarse con una base de datos, establece una conexión, generalmente utilizando un [protocolo](http://es.wikipedia.org/wiki/Protocolo) especializado. Esta conexión, generalmente se mantiene abierta el tiempo que dura la ejecución del programa y sólo es cerrada al finalizar el trabajo de la aplicación con la base de datos.

Este caso es típico de las aplicaciones que proveen servicio de páginas web a múltiples usuarios; en ellas, el número de operaciones sobre la base de datos y su cadencia dependen de la actividad de los usuarios de las páginas servidas.

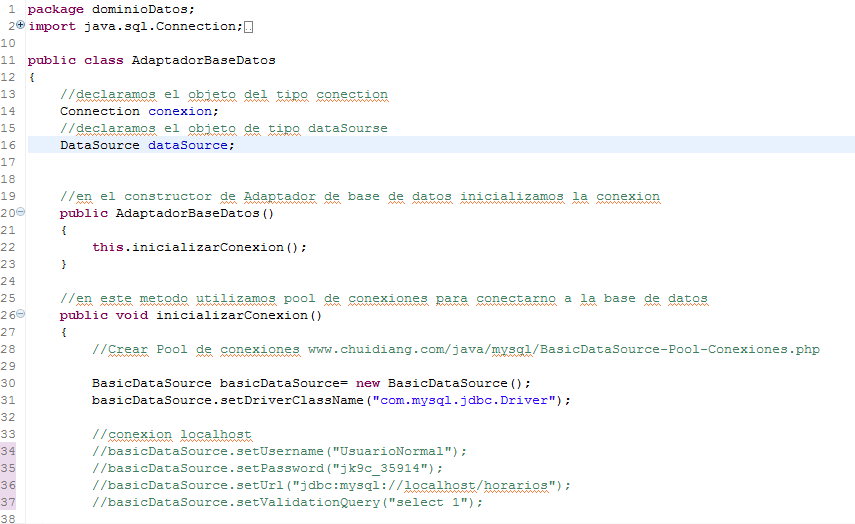
Al mantenerse abierto un grupo de conexiones, éstas son atribuidas a los diferentes hilos de ejecución únicamente el tiempo de una transacción con la base de datos. Al finalizar su utilización, la conexión se pone a disposición de otro hilo de ejecución que necesite de ese recurso, en lugar de cerrarla o de asignarla permanentemente a un único hilo de ejecución.

Según las políticas de agrupamiento de conexiones, cuando todas están en uso se establecen nuevas conexiones, y si ello no es posible, se deja el hilo de ejecución en espera de la liberación de alguna conexión. A la inversa, si pasa mucho tiempo sin que se utilicen las conexiones, algunas de ellas o todas podrían ser cerradas.

 La idea general de un pool de conexiones, es que cada vez que un cliente necesita una conexión se la solicita al pool. Esta conexión pertenece al cliente hasta que de algún modo se determina que no la necesita más, y en ese momento se le puede asignar la conexión a otro cliente.

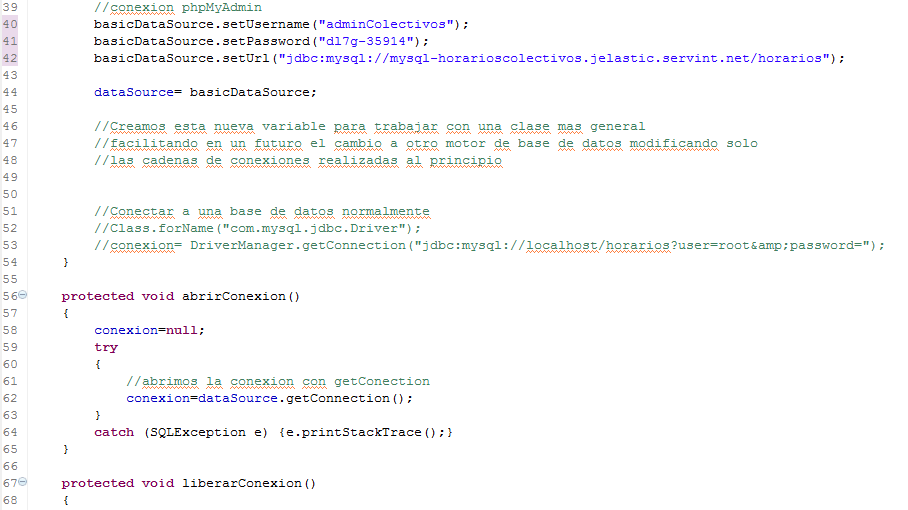
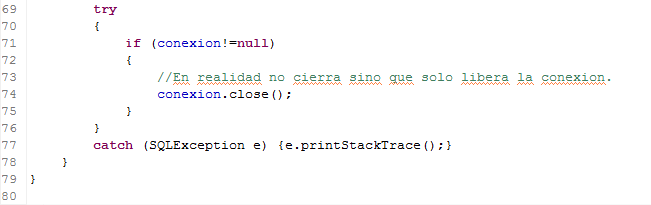
 Si un cliente necesita una conexión y no existe ninguna disponible, se queda esperando hasta que alguna conexión se libere.

 Otra ventaja es que las conexiones se mantienen abiertas, por lo que los clientes se ahorran el tiempo de conexión al DBMS al iniciar la aplicación.

AdaptadorBaseDatos.java

*Cadena para Localhost*

*Constructor de la clase*



*Liberamos la conexión*

*Abrimos la conexión*

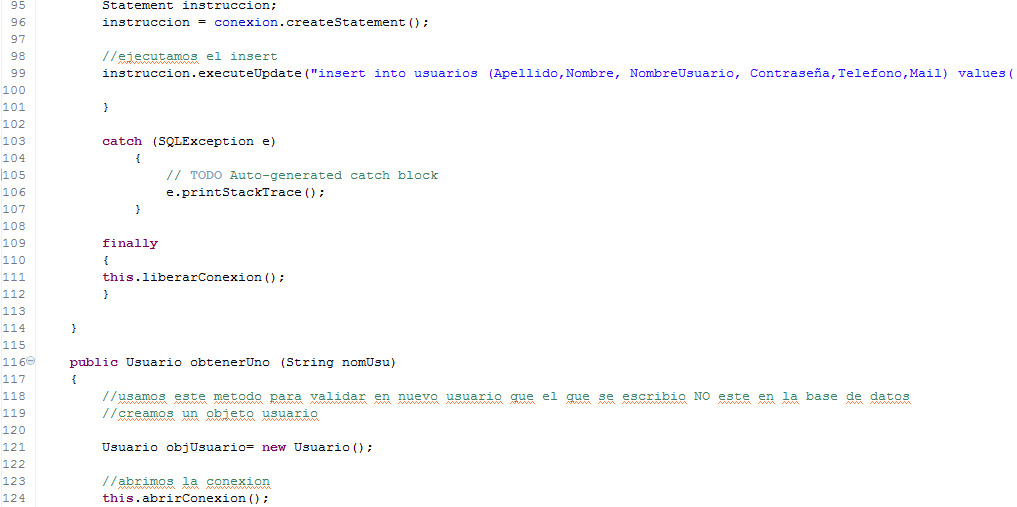
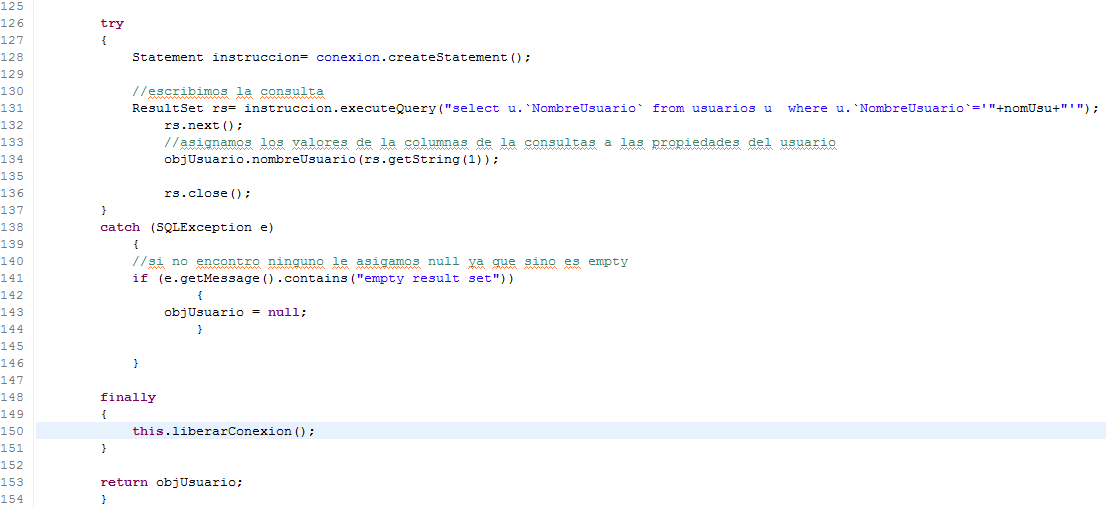
*Cadena para que la base de datos funcione en la web*

* Ejemplo de consulta, alta y modificación de base de datos en AdaptadorUsuarios.java

*La clase hereda de AdaptadorBaseDatos.java*



*Consulta a tabla usuarios*



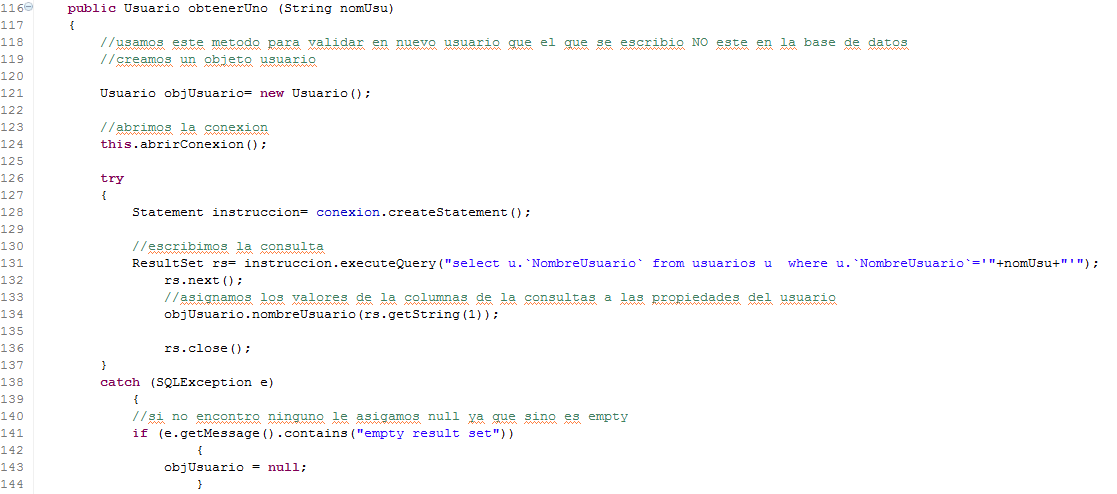
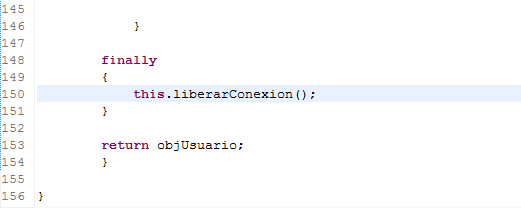
*Consulta a tabla usuarios*

*Inserción de tabla usuarios*

*Modificación de tabla usuarios*

* Ejemplo de Manejo de excepción, en clase AdaptadorUsuarios.java

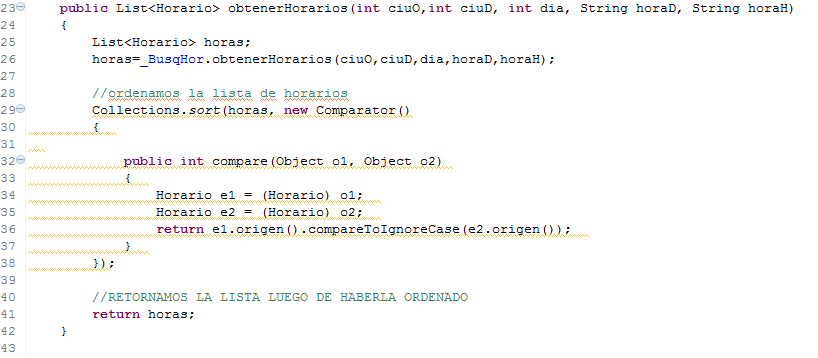
*Pedimos a la tabla usuarios que nos busque si existe el usuario con ese nombre de usuario*



*Retornamos el usuario en caso de encontrarlo o null en caso de no encontrara ningún usuario con ese nombre de usuario*

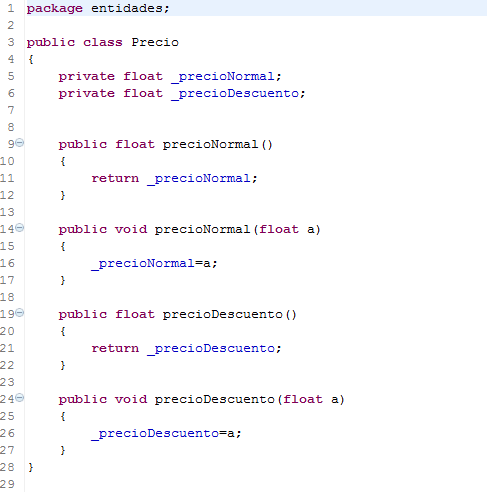
*En caso de no encontrar a ningún usuario se produce un error “empty result set” por el cual , capturamos la excepción y le asignamos a objUsuario = null.*

* Ejemplo de ordenamiento en clase LogicaBusqHor.java



*Horas es una lista de objetos “Horarios” por la cual la ordenamos de acuerdo a la propiedad origen() de cada uno.*

* Ejemplo de propiedades (“get” y “set”) en clase Precio.java



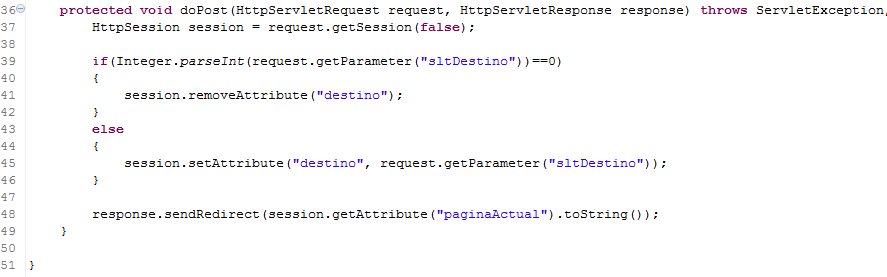
*Similar a propiedad “Set”, setea el valor almacenado en \_precioNormal*

*Similar a propiedad “Get”, retorna el valor almacenado en \_precioNormal*

*Declaramos las variables que tendrá la clase precio.java*

* Ejemplo de utilización de HttpSession en clase seleccionarDestino.java

*Guardamos en el atributo “destino” de la sesión el parámetro seleccionado en el jsp.*



* Ejemplo de comunicación entre capas. Mostramos el proceso de ingreso como usuario a la aplicación. (Mirar primero las flechas rojas y luego las negras)

C

A

P

A

P

R

E

S

E

N

T

A

C

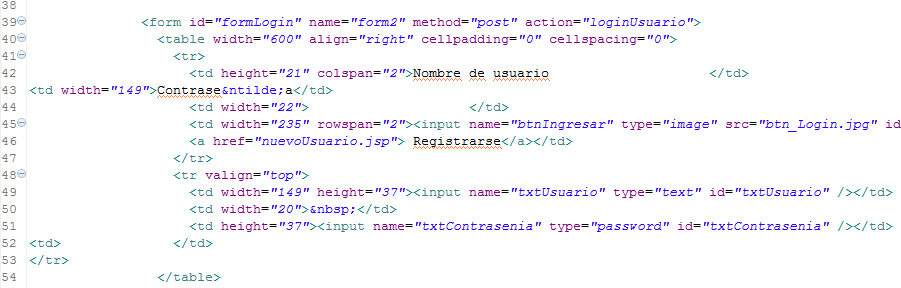
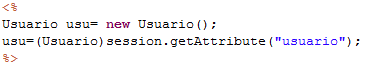
I

O

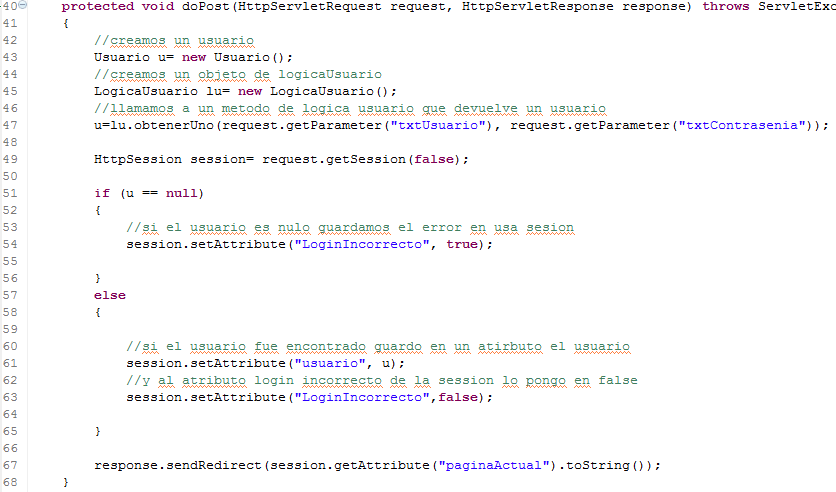
N

*Ejecuta el servlet “loginUsuario” al apretar el botón “btn\_Login.jpg”*

Login.jsp

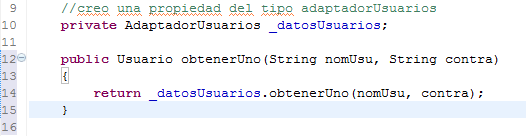


loginUsuario.java



*Creamos un objeto del tipo Usuario del paquete Entidades.*

LogicaUsuario.java



Capa Lógica de Negocio



Capa de datos

AdaptadorUsuario.java

AdaptadorUsuarios.java

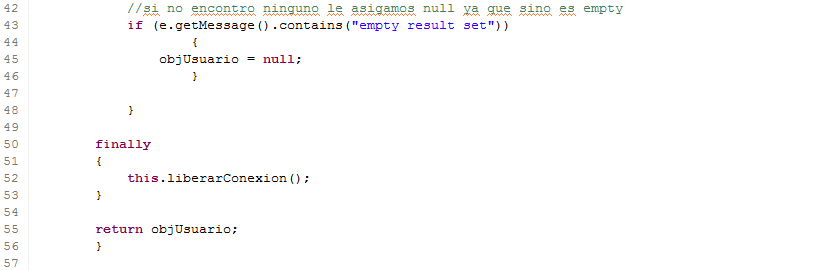
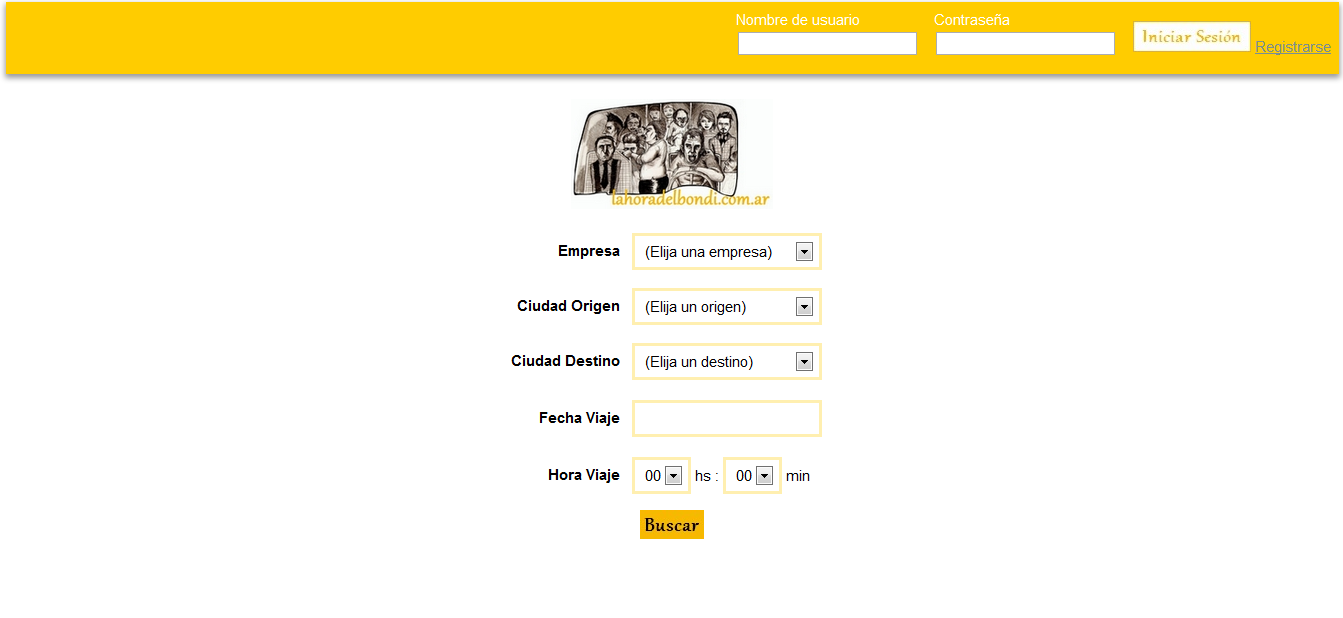


Tabla Usuarios

# Capturas de pantallas

* ***Página de inicio***



* ***Panel Administrador***
* ***Panel Usuario***

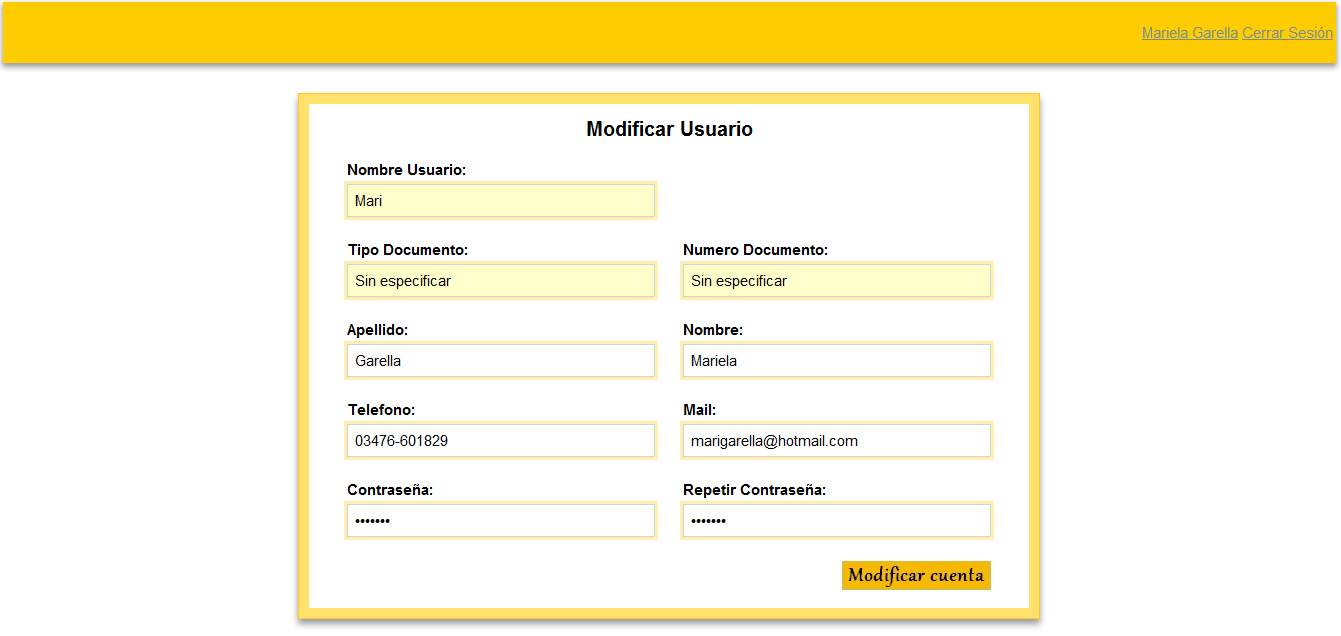


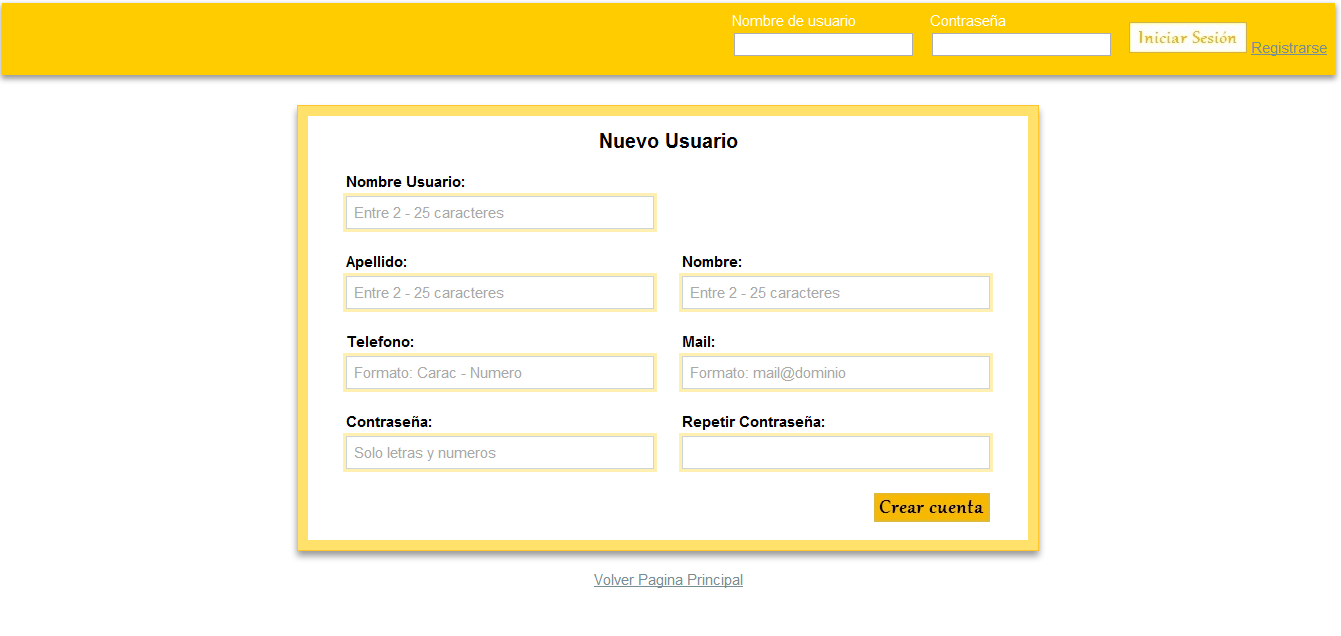
* ***Pantalla de búsqueda***



* Pantalla “Ver todos los horarios del día”



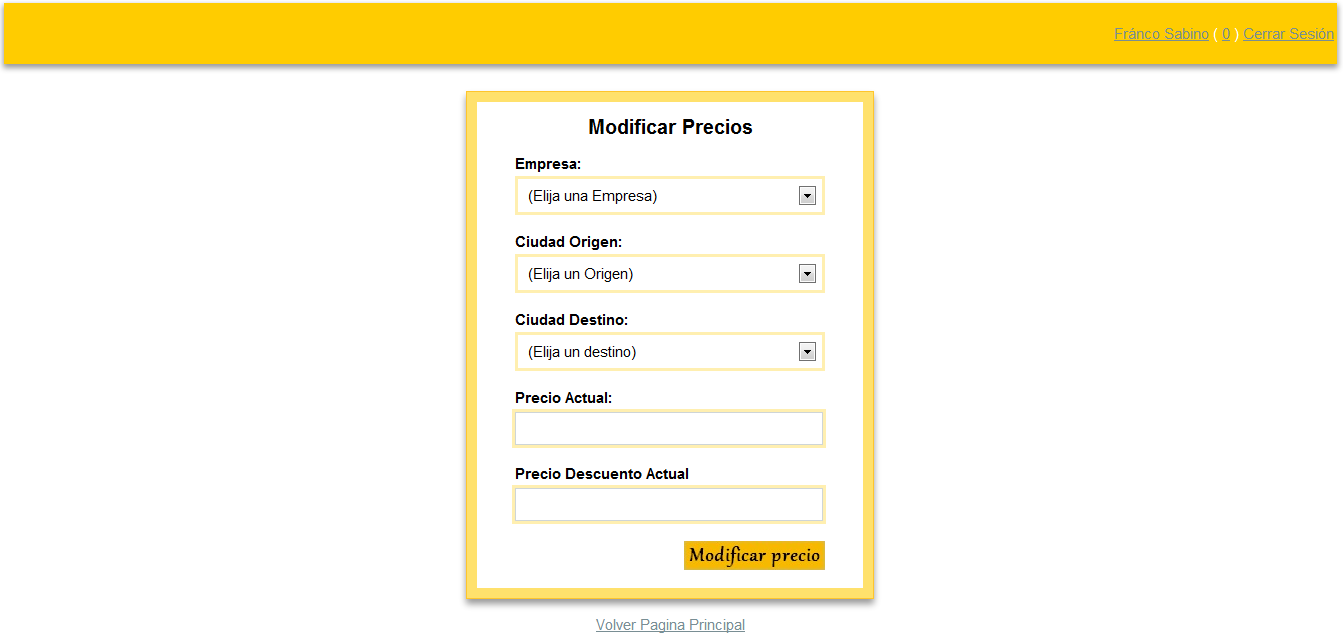
* ***Pantalla “Modificar Usuarios (o Administradores)"***
* ***Pantalla “Nuevo Usuario”***



* ***Pantalla “Contáctenos”***



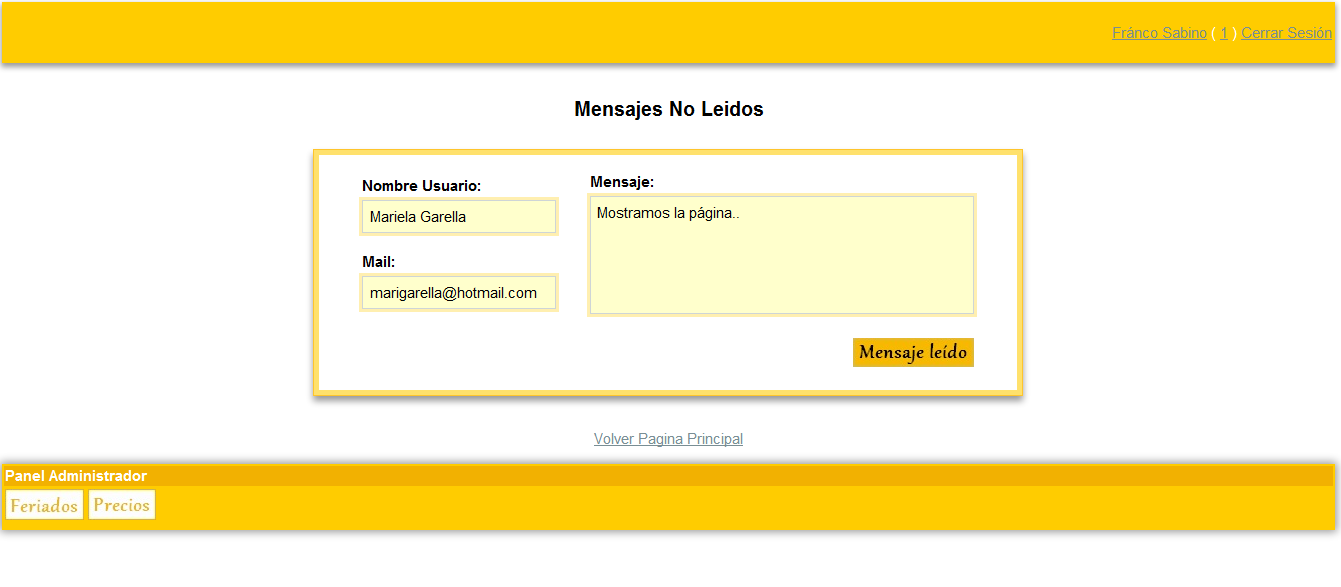
* ***Página “Modificar Precios”***



* ***Página “Modificar Feriados”***



* ***Página “Mensajes no leídos”***



# Conclusión

E

l objetivo de la aplicación “LaHoradelBondi” fue crear una página web que nos permitiera averiguar los horarios de los colectivos de las empresa “Güemes” y “Rosario Bus” ya que son las dos líneas que nos permiten viajar a las ciudades en las cual nos radicamos. Luego, nos dimos cuenta que podíamos realizar una estructura de base de datos que se ampliara a mayores empresas para poder aumentar el beneficio de consulta a mayores usuarios. Próximamente agregaremos mas empresas como “Monticas”, “Serodino”, etc. Además, será de gran ayuda ya que algunas empresas no tienen informatizados sus horarios, los usuarios podrán consultar, a través de internet o dispositivos móviles, sus distintos recorridos.

En cuanto al proyecto en sí, comenzamos a realizarlo a mediados de septiembre del pasado año, entre idas y vueltas y aproximadamente con un mes de duración, pudimos realizar la estructura de la base de datos (en MySql, ya que manejábamos este gestor porque lo aprendimos en Gestión de Datos). Terminado esto, decidimos realizar la aplicación en capas, nunca habíamos realizado un proyecto con esta estructura pero investigando pudimos formar los diferentes paquetes y relacionar la información entre los mismos. A medida que avanzábamos en la construcción de la aplicación fuimos agregando funcionalidades a la misma y así cumpliendo con los requerimientos de la asignatura. En cuanto al lenguaje Java utilizado, nos fue muy ventajoso usar el modelo en 3 capas ya que la información se manejaba con mejor lógica y con la utilización de servlets, sesiones y demás funciones pudimos darle forma al objetivo principal de “LaHoradelBondi”: “realizar una búsqueda para averiguar los horarios de salida y llegada”. Uno de los mayores problemas al cual nos enfrentamos fue darle “estética” a la página. El proyecto estaba finalizado pero no tuvimos en cuenta la parte “visual”, no contábamos con el conociemiento para realizar las modificaciones para hacerlo más “lindo” pero investigando en la web pudimos aplicar hojas de estilo y así mejorar la vista de nuestro proyecto.

Para finalizar podemos decir que este trabajo final, además de aplicar conocimientos aprendidos en “Java” y demás asignaturas de la carrera, pudimos aprender a buscar “buena” información en la web y así aplicarla a problemas que aparecían. Nos basamos, no solo en los requerimientos de la materia, sino en realizar ***“algo útil***” que pueda ser usado por personas para la realización de sus consultas. “LaHoradelBondi” será difundida a amigos, conocidos que usan las empresas que vamos a cargar y así extenderse a todos aquellos que quieran utilizarla.

# Información utilizada para realizar aplicación

[EATJ JSP Web Hosting](http://www.eatj.com/)

[Instalación de un entorno Web: Tomcat + JSP + MySQL « Linux, Java y programación](http://casidiablo.net/instalacion-de-un-entorno-web-tomcat-jsp-mysql/)

[¿Cómo conectar aplicaciones Java a MySQL? « MySQL en Español](http://mysql-espanol.org/2010/06/30/%C2%BFcomo-conectar-aplicaciones-java-a-mysql/)

[Ejemplo básico de programación con JDBC y MySQL « Linux, Java y programación](http://casidiablo.net/ejemplo-basico-de-programacion-con-jdbc-y-mysql/)

[La básico de MySQL con Java](http://www.chuidiang.com/java/mysql/mysql-java-basico.php)

[JReyes Blog!: Tutorial practico de programacion por capas con Java y Mysl](http://jreyes1982.blogspot.com/2011/04/tutorial-practico-de-programacion-por.html)

[BasicDataSource: Pool de conexiones](http://www.chuidiang.com/java/mysql/BasicDataSource-Pool-Conexiones.php)

[Tecnologías Java: Eclipse: User Libraries y Javadoc](http://tecnologiasjava.blogspot.com/2008/09/eclipse-user-libraries-y-javadoc.html)

[EjemploDataSource.java - chuidiang-ejemplos - Ejemplos de www.chuidiang.com - Google Project Hosting](http://code.google.com/p/chuidiang-ejemplos/source/browse/trunk/ejemplos-java-mysql/src/main/java/com/chuidiang/ejemplos/datasource_mysql/EjemploDataSource.java)

[MADEJA - GWT](http://www.juntadeandalucia.es/xwiki/bin/view/MADEJA/GWT)

[GWT – Ejemplo que usa RPC « Anónimos Anónimos](http://aalcuadrado.wordpress.com/2008/03/03/gwt-ejemplo-que-usa-rpc/)

[Operando datos contra el servidor utilizando GWT RPC - YouTube](http://www.youtube.com/watch?v=jMTF2ZzAjHM)

[MADEJA - GWT](http://www.juntadeandalucia.es/xwiki/bin/view/MADEJA/GWT#aplicaciones)

[Aplicaciones web 100% Java con Google Web Toolkit - Asociación javaHispano](http://www.javahispano.org/contenidos/es/aplicaciones_web_100_java_con_google_web_toolkit_/)

[Desarrollo de Aplicaciones Web 2.0 con GWT](http://www.slideshare.net/emiliobg/desarrollo-de-aplicaciones-web-20-con-gwt)

[Estoy haciendo una aplicacion JSP que permita ingresar un usuario y contraseña? - Yahoo! Respuestas](http://es.answers.yahoo.com/question/index?qid=20080419113754AAfGUET)

[www.itpuebla.edu.mx/Eventos/MemoriasyResSemanaInformatica2007/36-PedroBello.pdf](http://www.itpuebla.edu.mx/Eventos/MemoriasyResSemanaInformatica2007/36-PedroBello.pdf)

[Desarrollo de aplicaciones web en Java | Asociación de Usuarios de GNU/Linux de Castilla y León (AUGCyL)](http://www.augcyl.org/?q=glol-desarrollo-aplicaciones-web-java)

[Crear sitios y aplicaciones web en JSP con Java](http://www.cristalab.com/tutoriales/crear-sitios-y-aplicaciones-web-en-jsp-con-java-c74730l/)

[Instalar Eclipse y Tomcat para desarrollo con Java](http://www.cristalab.com/tutoriales/instalar-eclipse-y-tomcat-para-desarrollo-con-java-c64596l/)

[Cómo acceder a MySQL con JSP, Tomcat y JDBC Proyecto AjpdSoft](http://www.ajpdsoft.com/modules.php?name=News&file=article&sid=287)

[[Aporte] Registro y Login de usuarios con JSP,SERVLETS,MySQL - Foros del Web](http://www.forosdelweb.com/f67/aporte-registro-login-usuarios-con-jsp-servlets-mysql-930805/)

[Introducción al Desarrollo Visual de JSP con JDeveloper 10g. Programación en Castellano.](http://www.programacion.com/articulo/introduccion_al_desarrollo_visual_de_jsp_con_jdeveloper_10g_332)

[jsp + formulario - Foros del Web](http://www.forosdelweb.com/f67/jsp-formulario-240097/)

[Calendario emergente en formularios](http://www.sectorweb.net/2007/05/07/calendario-emergente-en-formularios/)

[Time picker](http://www.pit-r.de/timepicker/demo/pickerDemo.htm)

[dis.um.es/~lopezquesada/documentos/IES\_0607/DFSI/curso/UT11/cookiesJSP.pdf](http://dis.um.es/~lopezquesada/documentos/IES_0607/DFSI/curso/UT11/cookiesJSP.pdf)

[Video6 : Acceso a Datos JAVA with MySQL - Store Procedure Complejo - YouTube](http://www.youtube.com/watch?v=_847az-sPW4)

[Stored Procedure en MySQL (parte 1) [inicio] ~ Cassianet](http://cassianinet.blogspot.com/2011/05/stored-procedure-en-mysql-parte-1.html#axzz1b5hZ665P)

[Formulario con varios botones. Implementación usable y accesible](http://olgacarreras.blogspot.com/2007/02/formulario-con-varios-botones.html)

[10 Expresiones Regulares Imprescindibles en Desarrollo Web | Web.Ontuts](http://web.ontuts.com/snippets/10-expresiones-regulares-imprescindibles-en-desarrollo-web/)

[java ejemplos expresiones regulares - Buscar con Google](http://www.google.com.ar/search?rlz=1C1GGGE_esAR458AR458&gcx=c&sourceid=chrome&ie=UTF-8&q=expresion+regular+para+Apellido#sclient=psy-ab&hl=es&rlz=1C1GGGE_esAR458AR458&source=hp&q=java+ejemplos+expresiones+regulares&pbx=1&oq=java+ejemplos+expresiones+regulares&aq=f&aqi=&aql=&gs_sm=e&gs_upl=9181l13844l0l14113l11l8l0l0l0l5l1454l7125l6-1.5l6l0&bav=on.2,or.r_gc.r_pw.,cf.osb&fp=b)

[AppEngine](http://recursostic.educacion.es/observatorio/web/es/internet/recursos-online/796-appengine)

[Como hacer un formulario de contacto en HTML y CSS | WebTursos](http://web.tursos.com/tutoriales/html-css/como-hacer-un-formulario-de-contacto-i-html-y-css)

Maximiliano Broin - Franco Sabino