

Universidad Rafael Landívar
Facultad de ingeniería
Ingeniería en informática y sistemas
Manejo e implementación de archivos
Ing. David Fernando Luna Hernández

SEGUNDO PROYECTO DE PROGAMACIÓN E-MAIL

Javier Andrés Morales Gonzales	1210119
Diego Andrés Véliz Arauz	1230019
José Vinicio De León Jiménez	1072619

Guatemala, 26 de octubre del 2020

REQUERIMIENTOS DE SOFTWARE

Aplicación realizada en Java Maven:

Requerimientos mínimos:

- Java Development Kit (JDK):
Se requiere JDK 1.7 para poder ejecutar la aplicación.
- Sistema Operativo:
Sin requisito mínimo
- IDE:
Apache NetBeans

REQUERIMIENTOS DE HARDWARE

Requerimientos mínimos:

- Memoria RAM:
128 MB
- Memoria en disco:
Aproximadamente 124 MB de memoria en disco
- Procesador:
Mínimo Pentium 2 a 266 MHz

ALCANCES EN LA FUNCIONALIDAD

En esta entrega se hace uso de las funcionalidades implementadas en la versión anterior de este mismo proyecto. Con esto se logra una interacción entre los usuarios ingresados al sistema de archivos con las características nuevas como la creación de contactos y listas.

- Contactos
 - Búsqueda de usuarios
En esta implementación se hizo uso de una búsqueda de todos los usuarios ingresados al sistema de archivos, con esto se logra que el usuario logueado pueda encontrar a los distintos usuarios que forma parte del sistema. Esta función cuenta las opciones de búsqueda por usuario, por nombre y por apellido.
 - Asociación de usuarios
Para asociar el usuario que inicio sesión con otro usuario previamente ingresado es necesario buscarlo por algún parámetro que lo defina, luego

de esto se selecciona el usuario deseado para almacenar el registro y garantizar su persistencia en el sistema de archivos.

- Modificación de contactos

Luego de haber ingresado un contacto con el usuario logueado, este tendrá la posibilidad de modificar el registro que ingreso, para esto hace uso del menú de administración en esta sección del programa.

El usuario al abrir el menú de administración encontrará todos los registros asociados a su nombre, en dicha lista tendrá que seleccionar el registro a modificar. Permitiendo cambiarle únicamente la fecha y usuario de transacción.

- Eliminación de contactos

Para realizar esta operación el usuario logueado tendrá que seleccionar los registros asociados a su nombre y únicamente presionar el botón eliminar para eliminarlo de forma lógica del sistema de archivos.

- Listas

- Búsqueda de listas

En esta implementación se hizo uso de una búsqueda de todas las listas ingresadas al sistema de archivos, con esto se logra que el usuario logueado pueda encontrar a las distintas listas que forma parte del archivo.

- Modificación de listas

Luego de haber ingresado una lista con el usuario logueado, este tendrá la posibilidad de modificar el registro que ingreso, para esto hace uso del menú de administración en esta sección del programa.

El usuario al abrir el menú de administración encontrará todos los registros asociados a su nombre, en dicha lista tendrá que seleccionar el registro a modificar. Permitiendo cambiarle únicamente la fecha de creación y la descripción.









- Eliminación de listas

Para realizar esta operación el usuario logueado tendrá que seleccionar los registros asociados a su nombre y únicamente presionar el botón eliminar para eliminarlo de forma lógica del sistema de archivos.

















- Bloques de lista-usuario
 -

DOCUMENTACIÓN DE LA SOLUCIÓN

Dentro de la carpeta del proyecto podrá encontrar los siguientes archivos y directorios. En el directorio llamado MEIA se encuentran los archivos de texto que permiten la persistencia de datos que se ingresan programa.

 build	25/10/2020 17:38	Carpeta de archivos	
 MEIA	25/10/2020 17:38	Carpeta de archivos	
 nbproject	25/10/2020 17:38	Carpeta de archivos	
 src	25/10/2020 17:38	Carpeta de archivos	
 .gitignore	25/10/2020 17:38	Documento de tex...	1 KB
 build	25/10/2020 17:38	Documento XML	4 KB
 commons-codec-1.15	25/10/2020 17:38	Executable Jar File	346 KB
 manifest.mf	25/10/2020 17:38	Archivo MF	1 KB

La carpeta MEIA contiene los siguientes archivos de texto:

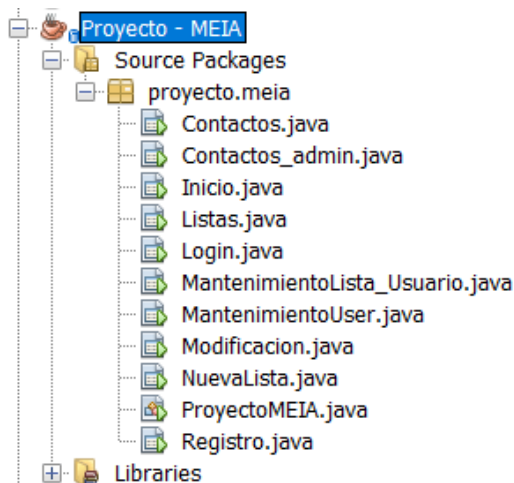
 img	25/10/2020 17:38	Carpeta de archivos	
 bitacora_contactos	25/10/2020 17:38	Documento de tex...	1 KB
 bitacora_lista	25/10/2020 17:38	Documento de tex...	1 KB
 bitacora_usuario	25/10/2020 17:38	Documento de tex...	1 KB
 contactos	25/10/2020 17:38	Documento de tex...	1 KB
 desc_bitacora_contactos	25/10/2020 17:38	Documento de tex...	1 KB
 desc_bitacora_lista	25/10/2020 17:38	Documento de tex...	1 KB
 desc_bitacora_usuario	25/10/2020 17:38	Documento de tex...	1 KB
 desc_contactos	25/10/2020 17:38	Documento de tex...	1 KB
 desc_lista	25/10/2020 17:38	Documento de tex...	1 KB
 desc_Lista_usuario	25/10/2020 17:38	Documento de tex...	0 KB
 desc_usuario	25/10/2020 17:38	Documento de tex...	1 KB
 lista	25/10/2020 17:38	Documento de tex...	1 KB
 Lista_usuario	25/10/2020 17:38	Documento de tex...	0 KB
 logo	25/10/2020 17:38	Archivo JPG	262 KB
 usuario	25/10/2020 17:38	Documento de tex...	4 KB

Estos archivos permiten el correcto funcionamiento del proyecto, por lo que no se recomienda la edición por medio de editores de texto al desconocerse la estructura de los archivos.

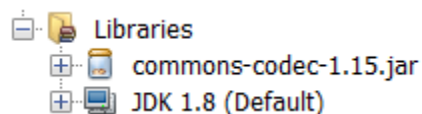
Cabe mencionar que el directorio img es el lugar físico del disco donde se almacenan las imágenes seleccionadas por el usuario en su registro inicial. Mientras que el archivo logo.png corresponde al icono identificador del email.

Para ejecutar la clase principal del proyecto se recomienda abrir la solución del mismo en NetBeans o algún IDE que permita la compilación de archivos java.

La solución del proyecto incluye las siguientes clases tipo JFrame, estas clases asocian una clase para lógica del funcionamiento junto con los métodos para la renderización gráfica del formulario.



El paquete Libraries incluye las dependencias necesarias para ejecutar el programa, específicamente el archivo commons-codec es utilizado para la encriptación de contraseñas en los usuarios ingresados.



Entre las clases nuevas de esta segunda entrega se encuentran las siguientes:

- Contactos.java
- Contactos_admin.java
- Listas.java

- MantenimientoLista_Usuario.java
- NuevaLista.java

CONTACTOS.JAVA

Dentro de esta clase se hizo uso de los siguientes métodos para el correcto funcionamiento del archivo tipo secuencial.

```
public String NormalizarResultado(String[] linea){...8 lines }

public String[] BuscarPorUsuario(String usuario, String path, String strError){...45 lines }

public void BuscarPorNombre(String nombre, String path, String strError, String usuario){...42 lines }

public void BuscarPorApellido(String apellido, String path, String strError, String usuario){...40 lines }

public void BuscarPorNombreApellido(String nombre, String apellido, String path, String strError, String usuario){...40 lines }

public boolean LlenarArchivo(String usuario, String contacto, String path)
{...31 lines }

void Reorganizar(String path_master, String path_bitacora){...67 lines }

boolean BorrarArchivos(String path1, String path2)
{...18 lines }

public String[] ObtenerRegistro(String key, String path, String strError){...43 lines }

String[] ObtenerLlavesActivas(String path, String strError){...45 lines }

int ObtenerDato(String path, String campo, String strError){...45 lines }

public void ActualizarDescriptorBitacora(String usuario)
{...41 lines }

    public void ActualizarDescriptorBitacora_Despues(String usuario)
    {...36 lines }

    public void ActualizarDescriptorMaster_Despues(String usuario)
    {...38 lines }

public boolean ExisteContacto(String key, String path, String strError)
{...46 lines }
```

CONTACTOS_ADMIN.JAVA

Dentro de esta clase se hizo uso de los siguientes métodos para el correcto funcionamiento del archivo tipo secuencial en operaciones de eliminación y actualización.

```
public boolean LlenarNuevo(String usuario, String fecha)
{...25 lines }

public void ActualizarDescriptorBitacora(String usuario)
{...41 lines }
```

```

public void BuscarUsuarios()
{...14 lines }

public void BuscarRegistros()
{...14 lines }

public ArrayList<String> LlenarUsuarios(String path, String strError){...45 lines }

public ArrayList<String> LlenarAdmin(String path, String usuario,String strError){...46 lines }

public int ObtenerPosicionRegistro(String key, String path, String strError)
{...48 lines }

void Eliminar(int posicion, String path, String linea){...14 lines }

public void ActualizarDescriptor_Eliminacion(String usuario, String path)
{...40 lines }

//REORGANIZACION
int ObtenerDato(String path, String campo, String strError){...45 lines }

void Reorganizar(String path_master, String path_bitacora){...67 lines }

String[] ObtenerLlavesActivas(String path, String strError){...45 lines }

boolean BorrarArchivos(String path1, String path2)
{...18 lines }

public String[] ObtenerRegistro(String key, String path, String strError){...43 lines }

public boolean LlenarArchivo(String usuario, String contacto, String usuario_transac, String fecha,String path)
{...26 lines }

public void ActualizarDescriptorBitacora_Despues(String usuario)
{...36 lines }

```

NUEVA_LISTA.JAVA

Dentro de esta clase se hizo uso de los siguientes métodos para el correcto funcionamiento del archivo tipo secuencial en operaciones de eliminación y actualización.

```

void Reorganizar(String path_master, String path_bitacora){...67 lines }

boolean BorrarArchivos(String path1, String path2)
{...18 lines }

public String[] ObtenerRegistro(String key, String path, String strError){...43 lines }

public void ActualizarDescriptorBitacora(String usuario)
{...41 lines }

public void ActualizarDescriptorBitacora_Despues(String usuario)
{...36 lines }

public void ActualizarDescriptorMaster_Despues(String usuario)
{...38 lines }

String[] ObtenerLlavesActivas(String path, String strError){...45 lines }

public boolean ExisteLlave(String key, String path, String strError)
{...46 lines }

public boolean LlenarArchivo(String[] registro, String path)
{...27 lines }

int ObtenerDato(String path, String campo, String strError){...45 lines }

```

LISTAS.JAVA

```
public void BuscarListas(boolean all)
{...32 lines }

public ArrayList<String> LlenarListasPorNombre(String path, String usuario, String nombreLista, String strError){...46 lines }

public ArrayList<String> LlenarListas(String path, String usuario, String strError){...45 lines }

public int ObtenerPosicionRegistro(String key, String path, String strError)
{...48 lines }

void Eliminar(int posicion, String path, String linea){...14 lines }

public void ActualizarDescriptor_Eliminacion(String usuario, String path)
{...40 lines }

public boolean LlenarNuevo(String descripcion, String fecha_creacion, String usuarios)
{...27 lines }

public void ActualizarDescriptorBitacora(String usuario)
{...41 lines }

void Reorganizar(String path_master, String path_bitacora){...67 lines }

boolean BorrarArchivos(String path1, String path2)
{...18 lines }

public String[] ObtenerRegistro(String key, String path, String strError){...43 lines }

String[] ObtenerLlavesActivas(String path, String strError){...45 lines }

public boolean LlenarArchivo(String[] registro, String path)
{...27 lines }

public void ActualizarDescriptorBitacora_Despues(String usuario)
{...36 lines }

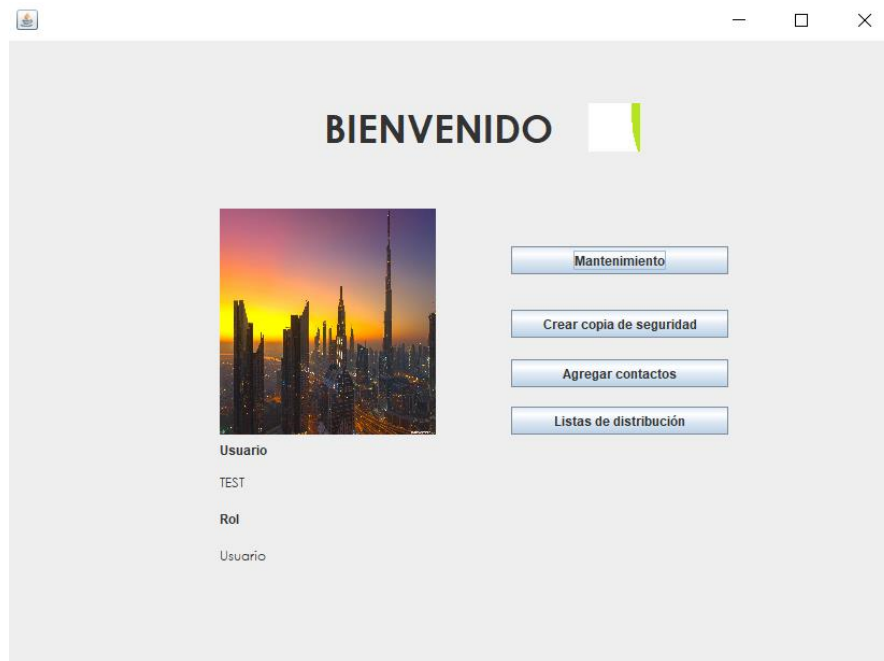
public void ActualizarDescriptorMaster_Despues(String usuario)
{...38 lines }
```


MANUAL DE USUARIO

En esta nueva versión del proyecto se hace uso del mismo proceso para registrarse como usuario o administrador. Por este motivo se omitirán esta serie de pasos, pero de igual forma se recomienda la consulta del manual de usuario anterior.

1. PÁGINA PRINCIPAL

Luego de acceder al menú principal del programa haciendo uso de sus credenciales de usuario, podrá observar el menú con las nuevas opciones implementadas.



-Haga clic en el botón agregar contactos

2. SECCIÓN DE CONTACTOS

Luego de haber abierto la sección de abrir contactos podrá observar el siguiente formulario.

The screenshot shows a web application window titled 'CONTACTOS' with a 'TEST' status. The interface is divided into two main sections: 'BUSCAR CONTACTOS' (Search Contacts) and 'RESULTADOS' (Results). The 'BUSCAR CONTACTOS' section contains three input fields labeled 'Usuario', 'Nombre', and 'Apellido', each followed by a 'BUSCAR' button. The 'RESULTADOS' section features a dropdown menu and an 'AGREGAR' button. Below the search fields is a button labeled 'ADMINISTRAR CONTACTOS'.

En esta sección podrá realizar tres tipos de búsqueda, por usuario, por nombre y por apellido. Los registros que coincidan con los parámetros especificados serán almacenados en el combo box de resultados.

Ejemplo de ingreso por usuario

This screenshot shows the same 'CONTACTOS' application window, but with the search results populated. The 'Usuario' field now contains the text 'AAA'. The 'RESULTADOS' dropdown menu displays the text 'AAA / test3 / test3 / test3'. The 'AGREGAR' button remains visible next to the dropdown. The 'ADMINISTRAR CONTACTOS' button is also present at the bottom.

- Llenar la caja de texto de usuario con el usuario a buscar
- Haga clic en el botón buscar
- Seleccione el usuario del combo box
- Haga clic en agregar para guardar el contacto

Ejemplo de ingreso por nombre



The screenshot shows a web application window titled "CONTACTOS" with a subtitle "TEST". The window has a search section on the left and a results section on the right. The search section is labeled "BUSCAR CONTACTOS" and contains three input fields: "Usuario", "Nombre", and "Apellido". The "Nombre" field is filled with "test2". Below these fields is a "BUSCAR" button. The results section is labeled "RESULTADOS" and contains a dropdown menu showing "CCC / test2 / test2 / test2" and an "AGREGAR" button. At the bottom right, there is a button labeled "ADMINISTRAR CONTACTOS".

- Llenar la caja de texto de nombre con el nombre de usuario a buscar
- Haga clic en el botón buscar
- Seleccione el usuario del combo box
- Haga clic en agregar para guardar el contacto

Ejemplo de ingreso por apellido



The screenshot shows the same "CONTACTOS" application window. In this example, the "Apellido" field in the "BUSCAR CONTACTOS" section is filled with "test1". The "RESULTADOS" section shows the dropdown menu with "BBB / test1 / test1 / test1" selected. The "AGREGAR" button is visible next to the dropdown. The "ADMINISTRAR CONTACTOS" button remains at the bottom right.

- Llenar la caja de texto de apellido con el apellido de usuario a buscar
- Haga clic en el botón buscar

- Seleccione el usuario del combo box
- Haga clic en agregar para guardar el contacto

Luego de haber agregado al contacto seleccionado saldrá un mensaje de confirmación.

3. BUSCAR CONTACTOS

The screenshot shows a web application window titled 'ADMINISTRACIÓN CONTACTOS'. Below the title, there is a sub-header 'TEST'. Underneath, the word 'AGREGADOS' is displayed. A table with one row contains the following data: 'TEST', 'AAA', '23/10/2020 19:34:01 | TEST', and a dropdown arrow. To the right of the table are three buttons: 'Buscar', 'Modificar', and 'Eliminar'. Below the table, there are two input fields: 'Usuario transacción' and 'Fecha transacción'. The 'Usuario transacción' field has a dropdown arrow. To the right of these fields is an 'Actualizar' button. At the bottom, there is a label 'Contacto' followed by a dropdown arrow.

-Haga clic en buscar

Se desplegarán en el combo box todos los registros asociados a dicho usuario logueado.

4. MODIFICAR CONTACTOS

ADMINISTRACIÓN
CONTACTOS
TEST

AGREGADOS

TEST	AAA	23/10/2020 19:34:01	TEST	...
------	-----	---------------------	------	-----

Buscar Modificar Eliminar

Usuario transacción: TEST Fecha transacción: 25/10/2020 20:23:55 Actualizar

Contacto: AAA

- Seleccione el usuario deseado del combo box
- Haga clic en modificar
- Modifique los campos expuestos en el luego de darle modificar
- Haga clic en actualizar para guardar los nuevos cambios del registro

5. ELIMINAR USUARIO

ADMINISTRACIÓN
CONTACTOS
TEST

AGREGADOS

TEST	AAA	23/10/2020 19:34:01	TEST	...
------	-----	---------------------	------	-----

Buscar Modificar Eliminar

Usuario transacción: Fecha transacción: Actualizar

Contacto: ---

Exito

Contacto eliminado exitosamente!!!

Aceptar

- Seleccione el usuario deseado del combo box
- Haga clic en eliminar

Saldrá un mensaje de confirmación cuando el registro haya sido eliminado

6. SECCIÓN DE LISTAS

Luego de haber abierto la sección de listas podrá observar el siguiente formulario.



En esta sección podrá realizar una búsqueda de las listas asociadas. Los registros que coincidan con los parámetros especificados serán almacenados en el combo box de resultados.

7. MODIFICAR LISTAS



-Seleccione la lista deseada del combo box

- Haga clic en modificar
- Modifique los campos expuestos en el luego de darle modificar
- Haga clic en actualizar para guardar los nuevos cambios del registro

8. ELIMINAR USUARIO

The screenshot displays a web application window titled "LISTAS DE DISTRIBUCIÓN" with a subtitle "PRUEBA". The interface includes a search bar with a dropdown menu showing "Amigos" and "IPRU". To the right of the search bar are buttons for "Buscar", "Modificar", and "Eliminar". Below the search bar, there are input fields for "Descripción" and "Fecha de creación", followed by an "Actualizar" button. At the bottom, there are labels for "Nombre:" and "Número de usuarios:", and a large button labeled "Ingreso de nueva lista de distribución". A modal dialog box is centered on the screen, titled "Exito", with a close button (X) in the top right corner. The dialog contains an information icon (i) and the text "Lista eliminada exitosamente!!!". Below the text is an "Aceptar" button.

- Seleccione la lista deseada del combo box
- Haga clic en eliminar

Saldrá un mensaje de confirmación cuando el registro haya sido eliminado