Sistema de Administración de Pasajes Interprovincial - SAPI

# Dedicatoria

Dedico este trabajo a la FIIS-UNI y a ….

# Índice

Dedicatoria 2

Índice 3

Resumen 5

Abstract 6

INTRODUCCIÓN 6

ALCANCES Y LIMITES 7

REQUERIMIENTOS 8

Requerimientos Funcionales 8

Requerimientos no Funcionales 8

Actores 8

Casos de uso 9

ANALISIS DE REQUERIMIENTOS 10

Proceso de logueo 10

Caso de uso 10

Especificación 10

Diagrama E-R 11

Análisis de la caja negra 11

Programación del servicio 12

Creación de Nuevas Rutas 14

Caso de Uso 14

Especificación 14

Análisis de Caja Negra 15

Diagrama E-R 15

Programación de Ruta 16

Caso de Uso 16

Especificación 16

Diagrama E-R 17

CRUD de Clientes 18

Análisis 18

Diseño 18

BIBLIOGRAFIA 19

# Resumen

# Abstract

# INTRODUCCIÓN

1. Contextualización del tema

Breve descripción del tema de estudio y su relevancia en el campo académico.

Justificación de la importancia del tema y su impacto en la disciplina.

2. Objetivos de la investigación

Declaración clara de los objetivos generales y específicos del estudio.

Explicación de lo que se busca lograr con la investigación.

# ALCANCES Y LIMITES

# REQUERIMIENTOS

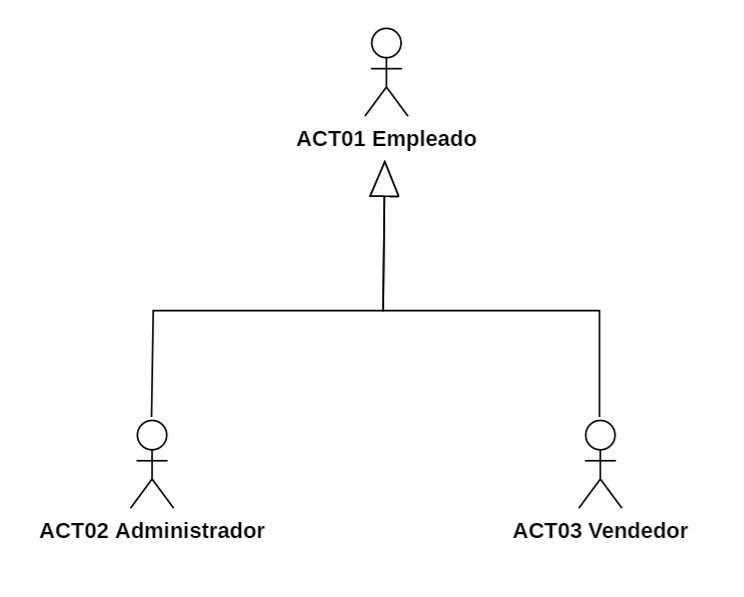
## Requerimientos Funcionales

1. El acceso al sistema será solo de personal autorizado mediante usuario y clave. Cada usuario debe tener su rol o perfil
2. El sistema debe permitir crear nuevas rutas
3. El sistema debe permitir programar viajes
4. El sistema debe permitir la venta de pasajes
5. Control de salida de buses
6. Control de llegada de buses
7. Reportes y consultas de ventas por diferentes indicadores

## Requerimientos no Funcionales

1. Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.
2. Si se identifican ataques de seguridad o brecha del sistema, el mismo no continuará operando hasta ser desbloqueado por un administrador de seguridad.
3. Desarrollado en Lenguaje Java con base de datos SQL Server.

## Actores



|  |  |  |
| --- | --- | --- |
| **CODIGO** | **NOMBRE** | **DESCRIPCION** |
| ACT01 | Empleado | Representa a todos los empleados que serán usuarios del sistema. |
| ACT02 | Administrador | Representa a la persona que realiza tareas administrativas en el sistema. |
| ACT03 | Vendedor | Representa a todos los vendedores. |

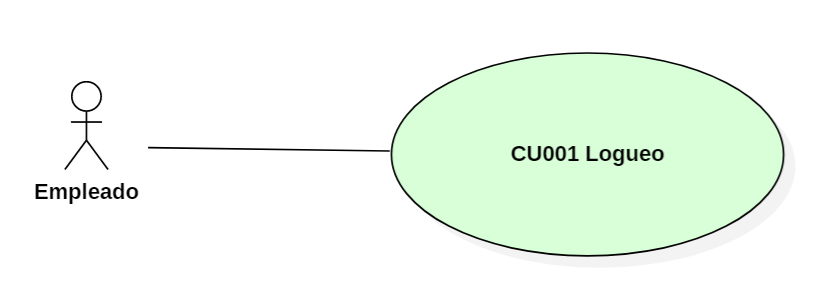
## Casos de uso

|  |  |  |
| --- | --- | --- |
| **CODIGO** | **NOMBRE** | **DESCRIPCION** |
| CU001 | Logueo | Representa el proceso de inicio de sesión de los usuarios de la aplicación. |
| CU002 | Creación de ruta | Representa el proceso para la creación de nuevas rutas. |
| CU003 | Programación de ruta | Representa la programación de salidas de buses en un día específico y hora específica.  Una ruta se puede programar varias veces en un mismo día. |
|  |  |  |

# ANALISIS DE REQUERIMIENTOS

## Proceso de logueo

### Caso de uso



### Especificación

|  |  |
| --- | --- |
| **Identificador** | **CU0001** |
| **Nombre** | Logueo o inicio de sesión |
| **Descripción** | Representa el proceso de inicio de sesión de los usuarios de la aplicación. Es necesario que exista el usuario en la base de datos y se encuentre habilitado. |
| **Actor** | ACT01 |
| **Flujo Básico** | Para poder iniciar este caso de uso, el empleado debe tener su usuario y clave a la mano.  El proceso a ejecutar es el siguiente:   1. El empleado ejecuta la aplicación. 2. La aplicación muestra la ventana de inicio de sesión o logueo. 3. El empleado ingresa su nombre de usuario. 4. El empleado ingresa su clave. 5. El usuario ejecuta el formulario. 6. Si los datos no son correctos, muestra el mensaje **"Datos incorrectos"** y regresa al punto 4. 7. Si los datos si son correctos, la aplicación muestra el formulario principal. |
| **Flujos Alternativos** | No aplica |
| **Escenarios Claves** | Si el empleado no cuenta con usuario y clave, debe informar y solicitar sus credenciales al área respectiva. |
| **Pre Condiciones** | El empleado debe tener su usuario y clave, y además deben estar activos. |
| **Post Condiciones** | Después de iniciar sesión correctamente, el empleado puede ejecutar las tareas que le corresponden según su perfil. |
| **Puntos de extensión** | No aplica. |
| **Requerimientos especiales** | La aplicación debe estar funcionando correctamente. |
| **Información adicional** | La aplicación solo puede ejecutarse desde la red interna de la empresa. |
| **Requerimientos Relacionados** | No aplica. |

### Diagrama E-R



### Análisis de la caja negra



En la clase **LogonService** se implementa el servicio validar, este servicio recibe como datos de entrada el **usuario** y la **clave**, si existen en la base de datos retorna un objeto de tipo **UsuarioDto** con los datos del usuario.

En caso de error, este se maneja a través de las propiedades **code** y **message** de la clase **LogonSevice**.

### Programación del servicio

public UsuarioDto validar(String usuario, String clave) {

UsuarioDto usuDto = null;

String query = "select u.IDEMPLEADO, u.USUARIO, '\*\*\*\*\*' CLAVE, "

+ "u.IDROL, r.NOMBRE, u.ACTIVO "

+ "from dbo.USUARIO u "

+ "join dbo.EMPLEADO e on u.IDEMPLEADO = e.IDEMPLEADO "

+ "join dbo.ROL r on u.IDROL = r.IDROL "

+ "where u.USUARIO = ? and u.CLAVE = ? "

+ "and u.ACTIVO = 1 and e.ACTIVO = 1";

PreparedStatement pstm = null;

ResultSet rs = null;

Connection cn = null;

this.setCode(1);

this.setMessage("Proceso ok!!!");

try {

cn = AccesoDB.getConnection();

pstm = cn.prepareStatement(query);

pstm.setString(1, usuario);

pstm.setString(2, clave);

rs = pstm.executeQuery();

if(!rs.next()){

rs.close();

pstm.close();

throw new SQLException("Datos incorrectos.");

}

usuDto = new UsuarioDto();

usuDto.setIdempleado(rs.getInt("IDEMPLEADO"));

usuDto.setUsuario(rs.getString("USUARIO"));

usuDto.setClave(rs.getString("CLAVE"));

usuDto.setIdrol(rs.getInt("IDROL"));

usuDto.setRol(rs.getString("NOMBRE"));

usuDto.setActivo(rs.getInt("ACTIVO"));

rs.close();

pstm.close();

} catch (SQLException e) {

this.setCode(-1);

this.setMessage(e.getMessage());

} catch (Exception e) {

this.setCode(-1);

this.setMessage("Error en el proceso, intenteo de nuevo.");

} finally {

try {

cn.close();

} catch (Exception e) {

}

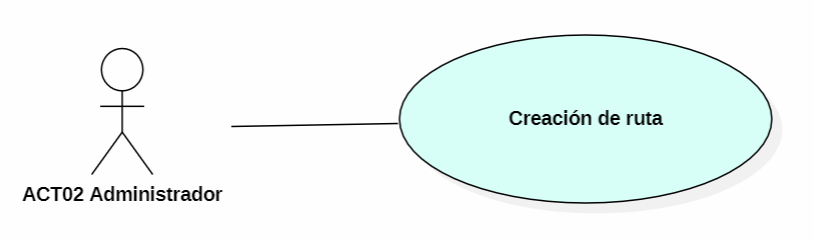
}

return usuDto;

}

## Creación de Nuevas Rutas

### Caso de Uso



### Especificación

|  |  |
| --- | --- |
| **Identificador** | **CU0002** |
| **Nombre** | Creación de ruta |
| **Descripción** | Representa el proceso para la creación de nuevas rutas.  Es necesario que exista el usuario en la base de datos y se encuentre habilitado. |
| **Actor** | ACT02 - Administrador |
| **Pre Condiciones** | Para poder iniciar el caso de uso el empleado debe haber iniciado sesión en el sistema. |
| **Flujo Básico** | El proceso a ejecutar es el siguiente:   1. El administrador ejecuta el formulario de registro de nuevas rutas. 2. El administrador debe ingresar los datos de la nueva ruta en cada uno de los controles del formulario. 3. El administrador hace click en el botón grabar, de existir algún error ir a punto 2. 4. El administrador recibe la confirmación que los datos han sido registrados correctamente. |
| **Flujos Alternativos** | No aplica |
| **Escenarios Claves** | Si el administrador no cuenta con usuario y clave, debe informar y solicitar sus credenciales al área respectiva. |
| **Post Condiciones** | Después de crear la nueva ruta correctamente ya puede ser programada en nuevas salidas. |
| **Puntos de extensión** | No aplica. |
| **Requerimientos especiales** | La aplicación debe estar funcionando correctamente. |
| **Información adicional** | La aplicación solo puede ejecutarse desde la red interna de la empresa. |
| **Requerimientos Relacionados** | No aplica. |

### Análisis de Caja Negra

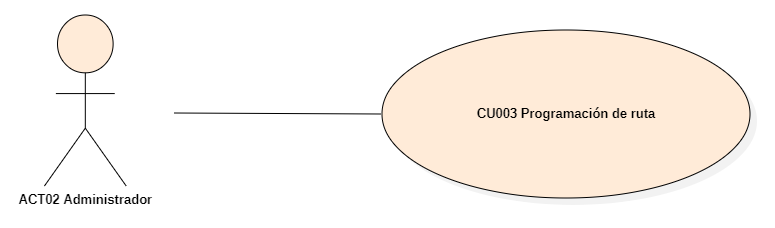


### Diagrama E-R



## Programación de Ruta

### Caso de Uso



### Especificación

|  |  |
| --- | --- |
| **Identificador** | **CU0003** |
| **Nombre** | Programación de ruta |
| **Descripción** | Representa la programación de salidas de buses en un día específico y hora específica.  Una ruta se puede programar varias veces en un mismo día. |
| **Actor** | ACT02 |
| **Pre Condiciones** | Para poder iniciar el caso de uso el empleado debe haber iniciado sesión en el sistema.  La ruta a programar debe estar creada y activa. |
| **Flujo Básico** | El proceso a ejecutar es el siguiente:   1. El empleado ejecuta el formulario de registro programación de rutas. 2. Luego se debe elegir o seleccionar la ciudad origen. 3. En base a la ciudad seleccionada en el punto anterior se debe seleccionar la ruta a programar. Se deben cargar los precios por defecto. 4. Se ingresa la fecha de salida. 5. Se debe ingresar la hora de salida. 6. Se debe seleccionar el bus. 7. Si es necesario se deben modificar los precios. 8. Finalmente se debe ejecutar el formulario, si no hay errores debe mostrar el mensaje con el código de la programación, en caso contrario, se debe realizar las correcciones y volver a ejecutar el formulario. |
| **Flujos Alternativos** | No aplica |
| **Escenarios Claves** | Si el empleado no cuenta con usuario y clave, debe informar y solicitar sus credenciales al área respectiva.  Ya deben estar cargas las rutas en el sistema. |
| **Post Condiciones** | Después de programar la ruta, se debe habilitar para que los usuarios puedan comprar sus pasajes.. |
| **Puntos de extensión** | No aplica. |
| **Requerimientos especiales** | La aplicación debe estar funcionando correctamente. |
| **Información adicional** | La aplicación solo puede ejecutarse desde la red interna de la empresa. |
| **Requerimientos Relacionados** | No aplica. |

### Diagrama E-R



## CRUD de Clientes

### Análisis



### Diseño



# BIBLIOGRAFIA

Kroll, P., & Kruchten, P. (2024). *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP.* Addison-Wesley Professional. Retrieved from https://books.google.com.pe/books/about/The\_Rational\_Unified\_Process\_Made\_Easy.html?id=Ea8qVou5ltEC&redir\_esc=y