

# UACM

Universidad Autónoma  
de la Ciudad de México

**NADA HUMANO ME ES AJENO**

## **DIAGRAMA DE CLASES**

ANALISIS Y MODELAMIENTO DE SOFTWARE

**Entrada estudiantil mediante QR**

Valadez Carmona Guadalupe Yamileth

Rodríguez Cervantes Kevin Manzur

Cruz Ovando Cristela Adelaida

Rodríguez Sánchez Diana Fabiola

Romero Cervantes Fátima Daniela

# HISTORIAL DE VERSIONES

Fecha	Versión	Descripción	Autores
13-09-2024	0.5	<ul style="list-style-type: none"><li>○ Versión preliminar del diagrama de clases.</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
19/09/2024	1.0	<ul style="list-style-type: none"><li>○ Identificación de las clases.</li><li>○ Identificación de los atributos.</li><li>○ Selección de los métodos.</li><li>○ Relación entre las clases.</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
20/09/2024	1.15	<ul style="list-style-type: none"><li>○ Diagramas de clases</li><li>○ Diagrama de entidad relacional</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
04-10-2024	2.0	<ul style="list-style-type: none"><li>○ Actualización del apartado “Definiciones, acrónimos y abreviaturas”.</li></ul> Eliminación del apartado “GUI obtención de QR end user”.	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
10-10-24	2.5	<ul style="list-style-type: none"><li>○ Rediseño de los diagramas de casos de uso.</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
12-10-24	2.6	<ul style="list-style-type: none"><li>○ Actualización de diagramas de clases.</li><li>○ Modificación de requisitos.</li><li>○ Selección de plataforma.</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela
22-11-24	3.00	<ul style="list-style-type: none"><li>○ Actualización de los métodos implementados.</li><li>○ Implementación de nuevos métodos.</li><li>○ Actualización de los parámetros.</li><li>○ Implementación de documentación en código.</li></ul>	Guadalupe Yamileth, Cristera Adelaida Diana Fabiola, Kevin Manzur, Fátima Daniela

## Índice

Demo()	1
Interfaz1	1
lectura	3
comprobar	4
newQr	5
dbr()	6
dbv	6
Diagrama de clases	8
Diagrama de secuencia	9
Definiciones, acrónimos y abreviaturas	10
Herramienta de diagrama	10
Diagrama de clases	10
Diagrama de secuencia	10
Bibliografía	10

## Demo()

Esta clase contiene el método principal del programa, que es el punto de entrada al ejecutar la aplicación. Al ejecutarse, se muestra una interfaz gráfica al usuario.

Clase			Demo()
Objetos			Descripción
OInterfaz1			- Implementado para lanzar la interfaz gráfica.
Modificadores	Tipo de dato	Atributo	Descripción
-	-	-	-
Método			Descripción
main()			- Método principal del programa, que es el punto de entrada * al ejecutar la aplicación.
-			-
<u>Requisitos</u>			
• Diseñar un método, que permita lanzar la interfaz gráfica 1.			
<u>Riesgos</u>			
• Existan más de dos métodos main en el proyecto. • Error al configurar el método main en el ejecutado de Net Beans. • Ocurre alguna excepción al ejecutar el método.			

## Interfaz1

La interfaz gráfica funciona en conjunto con el escáner de códigos QR. El escáner va a enviar el identificador obtenido. La clase enviará dicho identificador a la clase comprobar, la cual determinará si es un End User o un Visitante.

En caso afirmativo, la interfaz mostrará los datos del usuario (fotografía, nombre, carrera) junto con la leyenda "Autorizado". En caso contrario, se presentará la leyenda "Denegado", lo que permitirá al personal de seguridad decidir si se le otorga acceso como visitante, considerando los requisitos necesarios para ello.

Diseñar un formulario, que permita registrar la información de un visitante, cumpliendo los requisitos establecidos. El formulario contiene los campos:

1. Entrada para el nombre.
2. Entrada para el apellido paterno.
3. Entrada para el apellido materno.
4. Entrada para establecer el motivo.
5. Botón que permita seleccionar una imagen de los archivos del dispositivo. El tipo permitido del archivo: ".jpg, .jpeg, .png".

Todo esto, lo enviará a la clase comprobar.

Clase		interfaz0()
Objetos		Descripción

OComprobar			Enviar y recibir información de la base de datos.
OLectura			Conexión con el escáner QR, envía el identificador obtenido del QR.
Modificadores	Tipo de dato	Atributo	Descripción
private	String	mensaje	Almacena el mensaje de error detectado.
private	int	duración	Duración por defecto para el nuevo código QR del visitante.
private	File	selectedImage	Para seleccionar una imagen de los archivos.
Método			Descripción
regresaVentana()			Oculto el formulario para registrar a un visitante y sus botones asignados. Muestra los botones, contenedores de texto para mostrar la información de un QR escaneado.
seleccionarImagenActionPerformed()			El método se llama cuando se presiona el botón de seleccionar imagen
GuardarInfoVisitante()			Extrae la información ingresada de los contenedores de texto del formulario. Muestra por consola un mensaje, en el cual indica que faltan campos para llenar. Lee la imagen (identificación del visitante) como un array de bytes, en caso de error, mostramos por consola el error. Realizamos una comprobación, para evitar guardar información con campos vacíos. Si todos los campos de texto están llenos y la imagen ya fue seleccionada, se la enviamos a la clase comprobar, para que guarde la información en la DB de visitantes. Después de 3 segundos, limpia los campos de texto y el contenido de la imagen lo hace nulo.
VentanaEndUser()			Hace visibles los contenedores de texto y los botones para ingresar y guardar la información de un visitante. Ocultamos los contenedores, donde mostramos la información de un usuario. Recibimos el identificador contenido del escáner, el identificador ya viene formateado. Enviamos el identificador a la clase comprobar, la cual va a determinar si pertenece a un end user, visitante o no existe el identificador. En caso de que el identificador exista, mostrará la información del usuario en pantalla. En esta, también determinamos si el usuario puede o no acceder al plantel.

VentanaRegistrar()	Ocultamos los contenedores y botones para mostrar información de un usuario. Mostramos botones para registrar y guardar la información de un visitante. Además, mostramos los contenedores para la información del visitante, y limpiamos su contenido. Mostramos el botón para seleccionar un archivo de tipo imagen.
<u>Requisitos</u>	
<ul style="list-style-type: none"> <li>• Se mostrará información de los visitantes registrados en el día, la información a mostrar es el nombre, hora en que se registró, además, cada registro debe contener el botón para generar un PDF con la información registrada.</li> <li>• En el momento en que un End User accede al plantel, la pantalla debe de mostrar los campos: fotografía, nombre completo y área a la que pertenece.</li> <li>• Mostrar el mensaje en caso favorable: “Acceso autorizado”, en caso contrario, mostrar “Acceso denegado”. Dicho mensaje se mostrará en la pantalla durante 5 segundos, pasado el tiempo se quitará solo, al igual que la información del usuario.</li> <li>• La fotografía tiene que verse completa, en el espacio asignado en la UI.</li> <li>• La UI tiene que ser responsiva.</li> <li>• Los campos para el formulario de registro a visitantes tienen que ser llenados obligatoriamente, de no estarlo no se enviará información para su registro.</li> <li>• Solo se pueden cargar archivos de tipo imagen, de formatos jpg, png de un máximo de 5 megas.</li> <li>• La duración, tiene un valor máximo de 4.</li> <li>• Se le solicitara al visitante, que indique cuál es su motivo de visita.</li> <li>• Las imágenes escaneadas, se le aplicara un formato de compresión de imágenes. Con el objetivo de disminuir su peso.</li> </ul>	
<u>Riesgos</u>	
<ul style="list-style-type: none"> <li>• El mensaje y la información del anterior usuario no se quita automáticamente.</li> <li>• No se muestra ningún mensaje.</li> <li>• Error al mostrar la fotografía.</li> <li>• La fotografía es demasiado grande.</li> <li>• Error al recibir datos de la DB.</li> <li>• Datos de la DB, más grandes de los planteados en la clase.</li> <li>• El archivo de imagen, pesa más de lo requerido.</li> <li>• Corrompe el archivo, al realizar a compresión de la imagen.</li> </ul>	

## lectura

Se conecta con el escáner físico, cuando se escanea un código QR, este obtiene la dirección web. El método extrae información de dicho enlace, y envía solo el identificador a la GUI. El identificador es de tipo numérico.

Clase			lectura()
Objetos			Descripción
Scanner			Objeto para leer la entrada del usuario por consola.
Modificadores	Tipo de dato	Atributo	Descripción

private	identificador	int	Información extraída de tipo numérica, almacenada en el código QR.
Método			Descripción
readQRCode()			El escáner lee un QR, obtiene el enlace web almacenado y envía lo obtenido para ser formateado.
formatea(String enlaceAnalizado)			Formatea el enlace web obtenido, extrae el identificador almacenado dentro del enlace, y retorna solo el identificador.
<u>Requisitos</u>			
<ul style="list-style-type: none"> <li>• Escanear un QR.</li> <li>• Extraer el enlace web almacenado en el QR.</li> <li>• Solo retornar el identificador.</li> <li>• Formatear el enlace web, para extraer el identificador "id" o "identi".</li> </ul>			
<u>Riesgos</u>			
<ul style="list-style-type: none"> <li>• Interrupción inesperada con el escáner.</li> <li>• Requiere establecer un idioma específico en el teclado del dispositivo, para leer de manera correcta el enlace web.</li> <li>• Formatear el enlace web extraído.</li> <li>• Establecer límite de caracteres para el enlace web escaneado.</li> </ul>			

## comprobar

Realiza las comprobaciones con las bases de datos para los End User y los visitantes. La clase puede devolver información específica por consulta; a su vez, puede enviar información a las clases de base de datos.

Clase			comprobar ()
Objetos			Descripción
OSeridorUACM			Conexión con el servidor de la UACM, almacena información de los End User.
OServidorVis			Conexión con el servidor de Visitas, exclusivamente para los Visitantes.
Modificadores	Tipo de dato	Atributo	Descripción
private	String	info	Retorna un Sting, si el identificador se encuentra en alguna de las bases de datos.
private	String	QR	Amacena la información de un nuevo QR para los visitantes.
Método			Descripción
buscaMat(String identificador)			Realiza una consulta en las diferentes bases, determina si el identificador aparece en una determinada base de datos. Esta comprobación, se la enviamos a la interfaz 1.
getNombreVisitante (String identificador)			Obtenemos el nombre completo de la DB, se la enviamos a la GUI para ser mostrada.
getNombreEndUser(String identificador)			Retorna la información encontrada de un Visitante en la DB, y la envía para mostrar en la UI

<code>getOcupacionEndUser(String identificador)</code>	Obtenemos la ocupación o el cargo que tiene un End User, y la enviamos a la GUI para ser mostrada.
<code>getImgEndUser(String identificador)</code>	Obtenemos la imagen almacenada en la base de datos para un End User, y la enviamos a la GUI para ser mostrada.
<code>getVisitanteImg(String id)</code>	Obtenemos la imagen almacenada en la base de datos para un Visitante, y la enviamos a la GUI para ser mostrada.
<code>registra (String nombre,String apellidoM,String apellidoP, byte[] imagenBytes,String motivo,int duracion)</code>	Envía la información recuperada del formulario, la cual es requerida para brindarle el acceso a un usuario. Finalmente, la envía a la base de datos para su almacenamiento.
<u><i>Requisitos</i></u>	
<ul style="list-style-type: none"> <li>• Conexión exitosa con la base de datos End User, y de Visitantes.</li> <li>• Todos los métodos tienen que retornar información.</li> <li>• Se utilizarán condiciones, para determinar si es un estudiante, visitante, o un usuario no registrado.</li> <li>• Crear un método, el que permita enviar una imagen. Esta es la fotografía almacenada en la DB de End User.</li> <li>• Antes de enviar la información del formulario para visitantes, se tiene que generar un QR, el cual se almacenará en la base de datos.</li> </ul>	
<u><i>Riesgos</i></u>	
<ul style="list-style-type: none"> <li>• Error de conexión con alguna de las DB.</li> <li>• Error o excepción al enviar archivos.</li> </ul>	

## newQr

Permite la generación de un código QR único para un visitante. Los códigos QR almacena un enlace web, el cual tiene como método GET, un identificador, este se ha proporcionado como parámetro. Se debe crear el QR como una imagen, la cual será almacenada en la base de datos.

Clase			<code>newQr()</code>
Objetos			Descripción
-			-
Modificadores	Atributo	Tipo de dato	Descripción
-	-	-	-
Método			Descripción
<code>newQr(String identificador)</code>			Genera un QR, el cual almacena un enlace web.
<u><i>Requisitos</i></u>			
<ul style="list-style-type: none"> <li>• Genera códigos QR únicos.</li> <li>• Los códigos QR son generados aleatoriamente, tomando la hora del sistema.</li> <li>• Códigos QR utilizados para invitados.</li> <li>• Permite generar nuevos códigos QR para invitados con una duración de acceso de 1 hora, o en caso de ser requerido, se amplía la duración máxima a 4 horas.</li> </ul>			



<u>Riesgos</u>
<ul style="list-style-type: none"> <li>• Error a generar un QR para el visitante.</li> <li>• Le asigna una duración mayor a 4, para el QR.</li> <li>• Los QR se repiten, ya que no toma la hora del sistema para generarlos aleatoriamente.</li> </ul>

## dbr()

Conexión con la base de datos, realiza consulta a la base de datos de los End User.

Almacena la información de los End User. La base de datos simula la información almacenada por parte de la UACM. Toda la información almacenada es ficticia, es utilizada para pruebas de funcionamiento.

Clase			dbr()
Objetos			Descripción
-			-
Modificadores	Tipo de dato	Atributo	Descripción
private	String	databaseUrl	Nombre de la base de datos.
private	String	username	Usuario con el que se conectara a la base de datos.
private	String	password	Contraseña del usuario
Método			Descripción
buscaEnd(String identificador)			Realiza una búsqueda en la base de datos, comprueba si existe el identificador en la columna matricula. De existir, retorna un string, informando que sí existe; en caso contrario, retorna un string indicando que no existe.
getEndUserNombre(String identificador)			Retornamos el nombre completo del End User almacenado en la DB.
getEndUserOcupacion(String identificador)			Retornamos la ocupación del End User.
EndUserImg(String identificador)			Obtiene la fotografía almacenada en la DB, y la retorna para ser mostrada en la interfaz.
<u>Requisitos</u>			
<ul style="list-style-type: none"> <li>• Almacenar archivos de tipo “.jpe, .jpeg, .png, .jpg”, en la DBR.</li> <li>• Manejar los errores o excepciones al realizar una operación.</li> <li>• Obtener información específica de un End User, usando como condicional su identificador.</li> </ul>			
<u>Riesgos</u>			
<ul style="list-style-type: none"> <li>• No existe un manejo de errores al conectarte a la DBR.</li> <li>• El usuario asignado para realizar consultas en la base de datos, no tiene permisos por parte de los administradores.</li> <li>• Acceder a una tabla, la cual no tenemos acceso.</li> <li>• El archivo puede tener un peso excesivo.</li> </ul>			

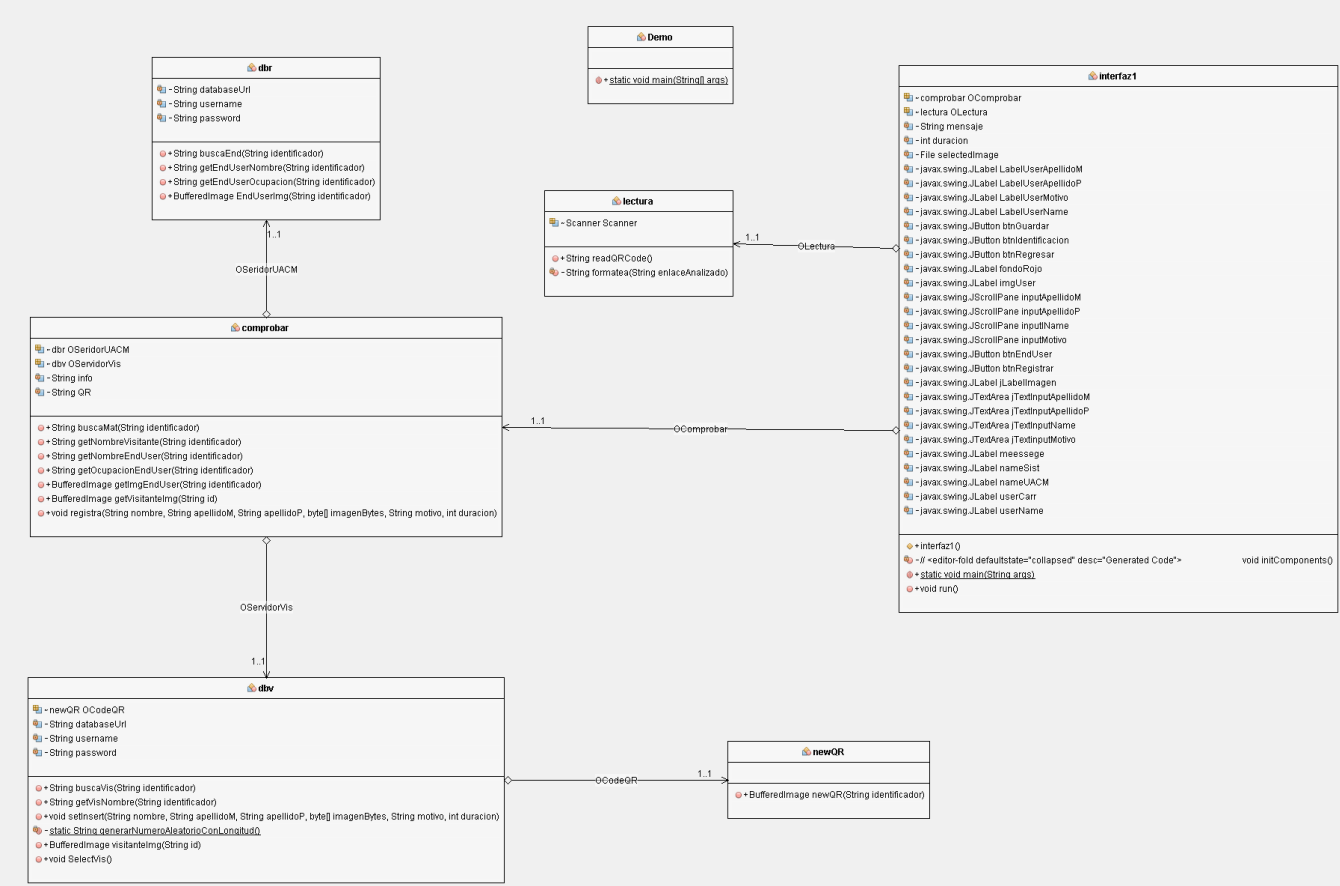
## dbv

Conexión con la base de datos, realiza consultas y agrega información a la base de datos de visitantes.

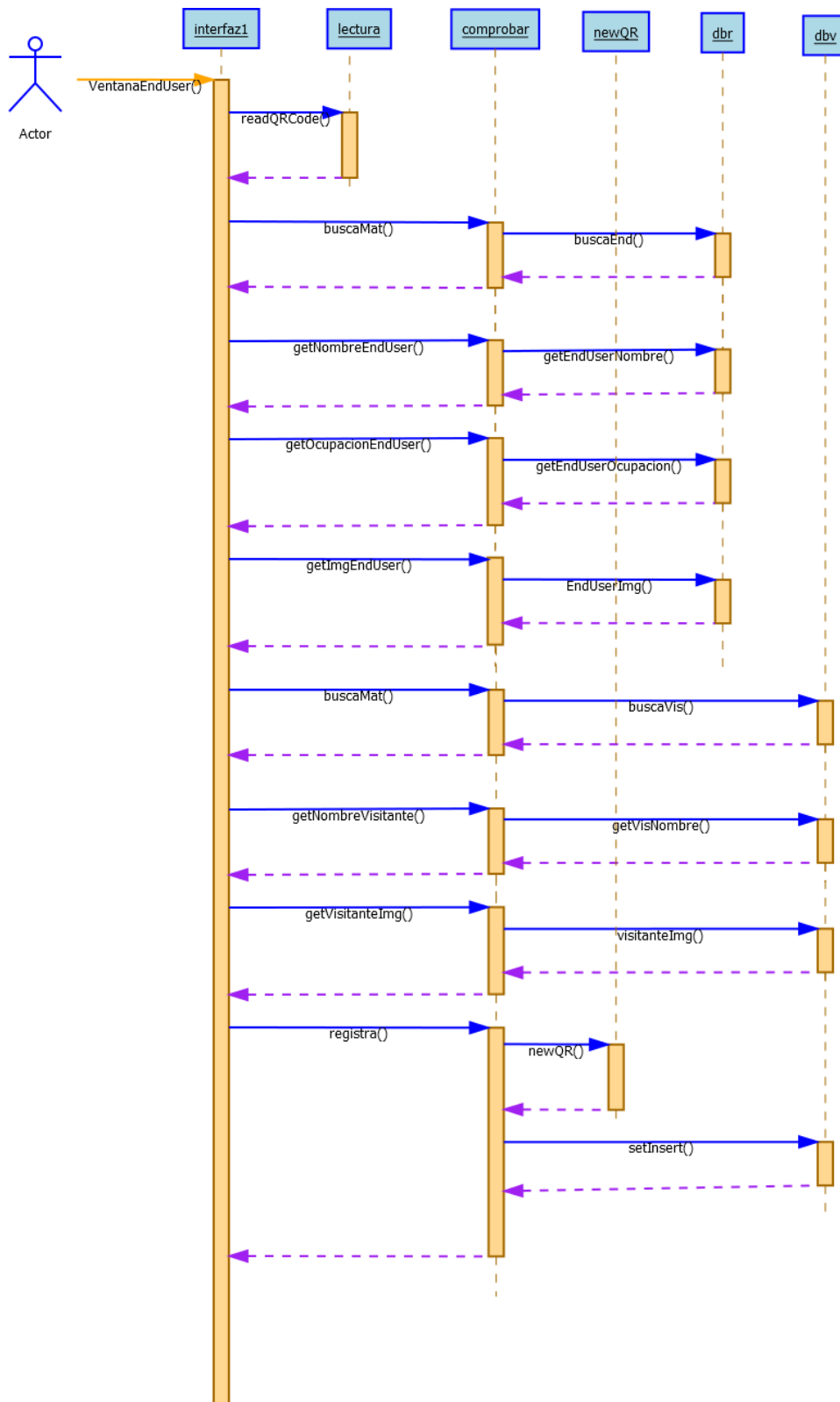
Es utilizada para almacenar la información de cada visitante en el plantel Cuauhtémoc. Toda la información almacenada es ficticia, es utilizada para pruebas de funcionamiento.

Clase			dbv()
Objetos			Descripción
OCodeQR			Genera un nuevo código QR para un visitante, recibe como parámetro el enlace web, este contiene un identificador numérico aleatorio.
Modificadores	Tipo de dato	Atributo	Descripción
private	String	databaseUrl	Nombre de la base de datos.
private	String	username	Usuario con el que se conectara a la base de datos.
private	String	password	Contraseña del usuario
Método			Descripción
buscaVis(String identificador)			Realiza una búsqueda en la base de datos, comprueba si existe el identificador en la columna id. De existir, retorna un string, informando que sí existe; en caso contrario, retorna un string indicando que no existe.
getVisNombre(String identificador)			Retorna el nombre completo almacenado en la base de datos, solo retorna el nombre que coincida con el identificador numérico pasado como parámetro.
getInsert(String nombre,String apellidoM,String apellidoP, byte[] imagenBytes,String motivo,int duracion, String QR)			Registra a un nuevo visitante, y guarda su información a la DB.
generarNumeroAleatorioConLongitud()			Genera un número aleatorio con una longitud aleatoria entre 5 y 15
visitanteImg(String id)			Muestra la imagen de la identificación proporcionada en su registro.
SelectVis()			Retorna los registros, del mismo día en que se esté utilizando. No muestra registros de días anteriores.
<u>Requisitos</u>			
<ul style="list-style-type: none"> <li>Almacenar archivos de tipo “.jpe, .jpeg, .png, .jpg”, en la DBV.</li> <li>Manejar los errores o excepciones al realizar una operación.</li> <li>Obtener información específica de un Visitante, usando como condicional su id.</li> <li>Retornar solo los registros de un día.</li> </ul>			
<u>Riesgos</u>			
<ul style="list-style-type: none"> <li>No existe un manejo de errores al conectarte a la DBV.</li> <li>El usuario asignado para realizar consultas en la base de datos, no tiene permisos por parte de los administradores.</li> <li>Acceder a una tabla, la cual no tenemos acceso.</li> <li>Retornar todos los registros realizados, de días anteriores.</li> </ul>			

# Diagrama de clases



## Diagrama de secuencia



dbv

## Definiciones, acrónimos y abreviaturas

- I. SCAM-QR:** Sistema de Control de Acceso Mediante QR.
- II. Campus:** Área de instalaciones universitarias donde se realizan actividades académicas y administrativas.
- III. Servidor:** Sistema informático que proporciona recursos y servicios a otros ordenadores a través de una red.
- IV. Base de Datos:** Conjunto organizado de datos almacenados electrónicamente, permitiendo su gestión y actualización.
- V. Normativas:** Reglas y directrices establecidas por una autoridad para regular comportamientos y acciones.
- VI. Políticas:** Normas que regulan las actividades y comportamiento dentro de la institución.
- VII. UI (User Interface):** UI significa Interfaz de Usuario. Se refiere a la parte del software con la que los usuarios interactúan directamente. El diseño de UI se enfoca en la disposición visual y la presentación de los elementos en la pantalla.
- VIII. UX (Experiencia de Usuario):** UX Se refiere a la experiencia general del usuario al interactuar con el software. El diseño de UX abarca aspectos más amplios que solo la apariencia y se centra en cómo se siente el usuario durante el uso del producto.
- IX. QA (Aseguramiento de la Calidad):** Es un proceso integral que se enfoca en asegurar que el software cumpla con los estándares de calidad y que funcione correctamente según los requisitos definidos.
- X. Formador:** Es un profesional encargado de capacitar a los usuarios, desarrolladores, y otros miembros del equipo sobre el uso de software, herramientas o metodologías específicas.
- XI. End User:** Usuarios finales, pertenecen a la UACM, los cuales utilizarán el programa, estos constan de estudiantes, docentes, personal administrativo, personal de limpieza, jardineros, personal de seguridad.
- XII. Visitante:** Usuario final, el cual no pertenece de ninguna manera al plantel educativo.
- XIII. DBR:** Base de datos de la UACM.
- XIV. DBV:** Base de datos de los visitantes.

## Herramienta de diagrama

### *Diagrama de clases*

Se utiliza la herramienta siguiente, la cual nos permite generar diagramas de clases a partir de código. El nombre de IDE es Apache NetBeans, versión del IDE 12.5, dentro de la cual, utilizamos el plugin "UML".

### *Diagrama de secuencia*

Se utiliza la herramienta siguiente, Dia.

## Bibliografía

- A.U.S. Gustavo Torossi. Diseño de Sistemas. El proceso unificado de desarrollo de Software.
- Cervantes, Velasco, Castro; Arquitectura de Software. Conceptos y Ciclo de Desarrollo; Cengage Learning, 2016.