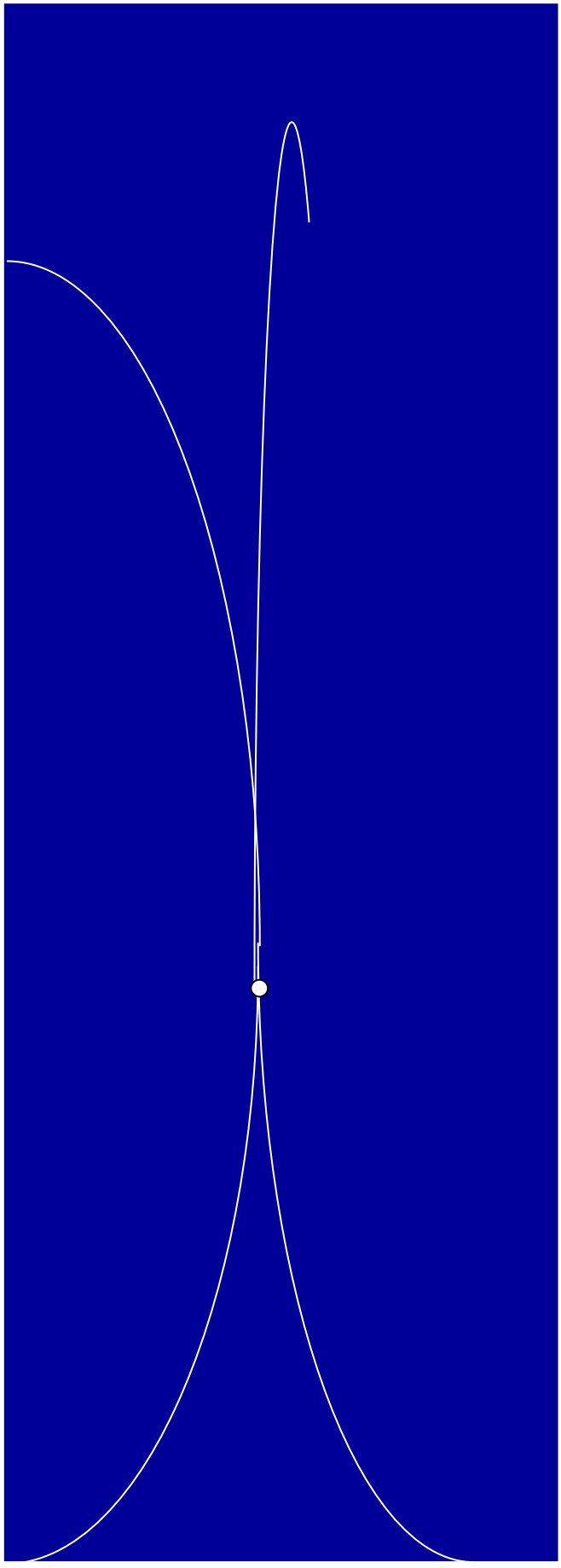

Programa educativo
INGENIERÍA EN SISTEMAS COMPUTACIONALES

Grupo
Especialidad DFS

Nombre de la materia
ARQUITECTURA DE SERVICIOS

Nombre del alumno
ANA PAULA BARAJAS CAMPOS
JAZMÍN BERENICE VEGA GÓMEZ

Nombre del trabajo
Segundo Avance del Proyecto

Nombre del Profesor
ROBERTO SUÁREZ ZINZUN

Fecha
23 de Marzo de 2023

SEGUNDO AVANCE DEL PROYECTO. (Documentación de Servicios).

- **SERVICIOS DE EDIFICIOS:**

Nombre del servicio: Edificios REST

Tipo de servicio: Entidad

URI: /Edificios

Plataforma de desarrollo: Python 3.7, Flask 2.2.3

Operaciones:

1. Registro de un nuevo edificio.
2. Consulta de todos los edificios.
3. Consulta de edificio por su id.
4. Actualización de datos de un edificio.
5. Eliminación de un edificio.

Operación: Registrar Edificio

Elemento	Valor
Método de Acceso	POST
Actor(es)	Administrador
Proceso	Registro de un edificio en la base de datos para poder asociar las salas que hay en él.

Entrada	{
	“nombreEdificio”: String,
	“descripción”: String
	}

Salida	{
	“estatus”: String,
	“mensaje”: String
	}

Operación: Consultar Edificios

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Consulta de todos los edificios existentes y activos en la base de datos.
Entrada	N/A

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“edificios”: [
	{
	“id”: int,
	“nombreEdificio”: String,
	“descripcion”: String
	}
	]
	}

Operación: Consultar Edificio individual

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Consulta de un edificio con base a su id.
Entrada	idEdificio(int)-En la cadena de consulta

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“edificio”:
	{
	“id”: int,
	“nombreEdificio”: String,
	“descripción”: String
	}
	}

Operación: Modificar Edificio

Elemento	Valor
Método de Acceso	PUT
Actor(es)	Administrador
Proceso	Modificación de los datos de un edificio. Validaciones: Verificar que el edificio que se desea modificar exista y esté en estatus de “Activo” en la base de datos.

Entrada	{
	“idEdificio”: int,
	“nombre”: String,
	“descripcion”:String
	}

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

Operación: Eliminar Edificio

Elemento	Valor
Método de Acceso	UPDATE
Actor(es)	Administrador
Proceso	Eliminación lógica de un edificio. Validaciones: Verificar que el edificio que se desea eliminar exista en la base de datos y no se encuentre ya en estatus de “Inactivo”
Entrada	idEdificio(int)-En la cadena de consulta

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

- **SERVICIOS DE SALAS:**

Nombre del servicio: Salas REST

Tipo de servicio: Entidad

URI: /Salas

Plataforma de desarrollo: Python 3.7, Flask 2.2.3

Operaciones:

1. Registro de una nueva sala.
2. Consulta de todas las salas.
3. Consulta de sala por su id.
4. Consultar salas por edificio.
5. Actualización de datos de una sala.
6. Eliminación de una sala.
7. Agregar mobiliario a sala.
8. Editar mobiliario.
9. Eliminar mobiliario de sala.

Operación: Registrar Sala

Elemento	Valor
Método de Acceso	POST
Actor(es)	Administrador
Proceso	Registro de una nueva sala en la base de datos.
Entrada	{ “nombreSala”: String, “descripción”: String, “capacidad”: int, “edificio”: String, “mobiliario”:[{ “numModelo”:String,

	“nombre”:String, “descripcion”:String, “cantidad”:int }] }

Salida	{
	“estatus”: String,
	“mensaje”: String
	}

Operación: Consultar Salas

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Consulta de todas las salas existentes en la base de datos.
Entrada	N/A

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“salas”: [
	{
	“id”: int,
	“nombreSala”: String,
	“descripción”: String,
	“capacidad”: int,
	“edificio”: String,
	“mobiliario”: [
	{
	“numModelo”:String,
	“nombre”:String,
	“descripcion”:String,
	“cantidad”:int
	}]
	}
	]
	}

Operación: Consultar Sala individual

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Consulta de una sala con base a su id.
Entrada	idSala(int)-En la cadena de consulta

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“sala”:
	{
	“id”: int,
	“nombreSala”: String,
	“descripción”: String,
	“capacidad”: int,
	“edificio”: String,
	“mobiliario”: [
	{
	“numModelo”:String,
	“nombre”:String,
	“descripcion”:String,
	“cantidad”:int
	}]
	}
	}

Operación: Consultar Salas por Edificio

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Consulta de todas las salas existentes en la base de datos.
Entrada	idEdificio(int)-En la cadena de consulta
Salida	{ “estatus”: String,

	“mensaje”: String,
	“salas”: [
	{
	“id”: int,
	“nombreSala”: String,
	“descripción”: String,
	“capacidad”: int
	}
	]
	}

Operación: Modificar Sala

Elemento	Valor
Método de Acceso	PUT
Actor(es)	Administrador
Proceso	Modificación de los datos de una sala. Validaciones: Verificar que la sala que se desea modificar exista en la base de datos y esté en estatus de “Activo”

Entrada	{
	“idSala”:int
	“nombreSala”: String,
	“descripción”: String,
	“capacidad”: int
	}

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

Operación: Eliminar Sala

Elemento	Valor
Método de Acceso	UPDATE
Actor(es)	Administrador
Proceso	Eliminación lógica de una sala. Validaciones:

	Verificar que la sala que se desea eliminar exista en la base de datos y no esté ya en estatus de “Inactivo”
Entrada	idSala(int)-En la cadena de consulta

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

Operación: Agregar Mobiliario a Sala

Elemento	Valor
Método de Acceso	POST
Actor(es)	Administrador
Proceso	Registro de mobiliario. Validaciones: Verificar que la sala a la que se desea agregar mobiliario exista y esté en estatus de “Activo”. Verificar que el mobiliario no exista aun en la sala.

Entrada	{
	“idSala”: int,
	“numModelo”:String,
	“nombre”: String,
	“descripcion”: String,
	“cantidad”: int
	}

Salida	{
	“estatus”: String,
	“mensaje”: String
	}

Operación: Actualizar Mobiliario

Elemento	Valor
Método de Acceso	PUT
Actor(es)	Administrador

Proceso	Modificación de los datos del mobiliario de una sala. Se validará que el mobiliario a modificar exista en la sala y que la sala se encuentre en estado de “Activo”.

Entrada	{
	“idSala”: int,
	“numModelo”:String,
	“nombre”: String,
	“descripcion”: String,
	“cantidad”: int
	}

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

Operación: Eliminar Mobiliario

Elemento	Valor
Método de Acceso	DELETE
Actor(es)	Administrador
Proceso	Se eliminará el mobiliario (de la sala que se seleccionó). Se validará que el mobiliario a eliminar exista en la sala.

Entrada	{
	“idSala”: int,
	“numModelo”:String
	}

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

- **SERVICIOS DE RESERVAS:**

Nombre del servicio: Reservas REST

Tipo de servicio: Entidad/Tarea

URI: /Reservas

Plataforma de desarrollo: Python 3.7, Flask 2.2.3

Operaciones:

1. Consulta de disponibilidad de reserva.
2. Agregar una Reserva.
3. Consulta de las reservaciones existentes.
4. Consulta de reserva por su id.
5. Actualización de la reserva realizada.
6. Eliminar Reserva.

Operación: Consulta de disponibilidad de Reserva.

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Mostrar el estatus de disponibilidad de una reserva para una sala en una fecha y hora.

Entrada	{
	“sala”:int,
	“fechaInicio”:Date,
	“fechaFin”:Date,
	“horaInicio”:String,
	“horaFin”:String
	}

Salida	{ “estatus”: String, “mensaje”: String, “reserva”:

	{
	“id”: int,
	“tipo”:String,
	“motivo”:String,
	“estatus”: String
	}
	}

Operación: Agregar Reserva

Elemento	Valor
Método de Acceso	POST
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Registrar una reserva de sala en la base de datos. Validar que la sala donde se desea hacer la reserva esté disponible y esté activa.

Entrada	{
	“tipo”:String,
	“motivo”:String,
	“horaInicio”: String,
	“horaFin”: String,
	“fechaInicio”: Date,
	“fechaFin”: Date,
	“sala”: int
	}

Salida	{
	“estatus”: String,
	“mensaje”: String
	}

Operación: Consulta de las reservaciones existentes.

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Mostar los datos de las Reservas que se han hecho en la Base de Datos.

Entrada	N/A

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“reservas”: [
	{
	“idReserva”:int,
	“tipo”:String,
	“motivo”:String,
	“horaInicio”: String,
	“horaFin”: String,
	“fechaInicio”: Date,
	“fechaFin”: Date,
	“estatus”: String,
	“sala”: String
	}]
	}

Operación: Consulta de reserva por su id.

Elemento	**Valor**
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Mostar la reserva filtrándola por su id. Validar que exista la reserva en la Base de Datos.
Entrada	idReserva(int)-En la cadena de consulta
Salida	{ “estatus”: String, “mensaje”: String, “reserva”: { “idReserva”:int, “tipo”:String, “motivo”:String, “horaInicio”: String, “horaFin”: String, “fechaInicio”: Date, “fechaFin”: Date,

	“estatus”: String, “sala”: String } }

Operación: Consulta de salas disponibles en una fecha y hora determinada.

Elemento	Valor
Método de Acceso	GET
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Mostar las salas que no tienen reservas activas en una fecha y hora determinadas.

Entrada	{
	“horaInicio”: String,
	“horaFin”: String,
	“fechaInicio”: Date,
	“fechaFin”: Date
	}

Salida	{
	“estatus”: String,
	“mensaje”: String,
	“salas”: [
	{
	“idSala”: int,
	“nombre”:String
	}]
	}

Operación: Actualización de la reserva realizada.

Elemento	Valor
Método de Acceso	PUT
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Modificación de o los datos de la reserva de la sala que ya se seleccionó previamente. Validaciones: Verificar que las horas y fechas no se empalmen con la de alguna otra reserva activa.

Entrada	idReserva: int horaInicio: String (en caso de querer modificarlo) horaFin: String (en caso de querer modificarlo) fecha: Date (en caso de querer modificarlo) estatus: String (en caso de querer modificarlo) sala: int (en caso de querer modificarlo)

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}

Operación: Eliminar Reserva.

Elemento	Valor
Método de Acceso	UPDATE
Actor(es)	Alumno, docente, coordinador, administrador
Proceso	Eliminación lógica de la reserva seleccionada (por su id). Validar que la reserva exista en la Base de Datos y no esté en estado de “Cancelado”
Entrada	idReserva(int)-En la cadena de consulta

Salida	{
	“estatus”: String,
	“mensaje”: String,
	}