***Validar usuario (POST)***

* Recibe: correo, contrasena
* Entrega: id, rol
* Nota: Con el id y el rol creas el JWT y clave secreta del rol, y eso lo mandas a la cookie

***OBTENER TODOS LOS CURSOS (GET):***

* Recibe: ----
* Query: “SELECT id AS CursoId, nombre AS NombreCurso, descripcion AS DescripcionCurso FROM Cursos”

***Obtener progreso de los cursos por alumno (POST)***

* Recibe: IdAlumno
* Entrega: una lista de todos los cursos que tiene el alumno pero con estos datos

{

"IdAlumno": 2,

"IdCurso": 1,

"NombreCurso": "Capacitacion inicial para asesores de cobranza de expertis",

"ProgresoCurso": 0,

"ExamenHabilitado": false,

"NotaFinal": 0,

"CantidadIntentos": 0,

"EnlaceExamen": <https://docs.google.com/forms/d/e/1FAIpQLSeC6RsWbdxtzyK4qwOw4XMJWl-H7A_8Mi8nJc6_Zgx2MJrrlQ/viewform?usp=sharing>

}

* Nota: por el momento como hay solo 1 curso solo devuelve 1 , si hay mas cursos , trae una array de objetos con ese formato

***ObtenerYActualizarProcesos (PUT)***

* Recibe: (@IdAlumno , @IdCurso):
* Entrega: “MENSAJE: SE CONSUMIO UN INTENTO”
* Nota: Deshabilita el examen (examenHabilitado = 0). Incrementa el número de intentos en 1 (Intentos = Intentos + 1).

***CambiarEstado (PUT)***

* Recibe: (@IdAlumno , @IdCurso):
* Entrega: “MENSAJE: SE CAMBIO EL ESTADO”
* Nota: HACE ESTO:  
  UPDATE Progresos

SET examenHabilitado = 1

WHERE alumnoId = @idUsuario AND cursoId = @idCurso;

***Validar usuario (POST)***

* Recibe: correo, contrasena
* Entrega: id, rol
* Nota: Con el id y el rol creas el JWT y clave secreta del rol, y eso lo mandas a la cookie

***ObtenerProgresoCurso (POST):***

* Recibe: CursoId
* ENTREGA : entrega un array de objetos con estos datos:   
  "UsuarioId": 1,

"NombreUsuario": "Juan Pérez",

"VideosVistos": 11,

"Intentos": 1,

"NotaFinalCurso": 0,

"CantidadCursos": 11,

"NombreCurso": "Capacitacion inicial para asesores de cobranza de expertis"

* Nota: entrega los datos de los usuarios que tenga el id del curso que se mando

***IngresarNotas (PUT)***

* Recibe: IdAlumno, IdCurso, nota
* Entrega: MENSAJE: NOTA INGRESADA
* Nota: INGRESA O ACTUALIZA LA NOTA QUE TIENE EL ALUMNO EN DICHO CURSO

***ObtenerVideos (POST)***

* Recibe: idUsuario, idCurso
* Entrega: TODOS LOS VIDEOS DE UN CURSO EN ESPECIFICO , EL ARRAY DE LOS VIDEOS TENDRAN ESTE FORMATO:  
   "VideoId": 1,

"Titulo": "CONSULTA CLIENTE",

"Link": "https://res.cloudinary.com/dtjsttl1h/video/upload/v1737995445/1.CONSULTA\_CLIENTE\_e006yj.mp4",

"Orden": 1,

"Estado": true,

"VideoVisto": true

***AQUÍ USAREMOS VALORES QUE SE MANDARAN POR PARAMETRO:***

***cursosAlumnos (GET)***

* Recibe: idUsuario (PARAMS)
* Entrega: ENTREGA UN ARRAY DE OBJETOS QUE TIENEN ESTE FORMATO:  
   "VideosVistos": 11,

"Usuario": "Juan Pérez",

"CursoId": 1,

"NombreCurso": "Capacitacion inicial para asesores de cobranza de expertis",

"CantidadVideos": 11,

"PrimerVideoId": 1

Nota: EL ATRIBUTO QUE DICE "PrimerVideoId" ES RESPECTO A UN CURSO, CUAL ES EL ID DEL PRIMER VIDEO

***IngresarNotas (PUT)***

* Recibe: idUsuario (PARAMS) --- PrimerVideoId , CursoId
* Entrega: MENSAJE: NOTA VIDEO ACTUALIZADO
* Nota: **Actualiza el estado del primer video** en la tabla ProgresoVideo para marcarlo como "activado" (estado = 1).

***CursoProgreso (POST)***

* Recibe: idUsuario (PARAMS) --- idCurso
* Entrega: TRAE LOS MISMO DATOS QUE HACE EL GET (***cursosAlumno***) PERO ME TRAE LOS DATOS DE UN CURSO EN ESPECIFICO, YA NO TRAE UN ARRAY DE OBJETOS, SOLO TRAE UN OBJETO

***ActualizarSiguienteVideo (POST)***

* Recibe: idVideo (PARAMS) --- idUsuario, idCurso

 **Busca el ID del siguiente video**

* Se basa en la columna orden dentro de la tabla Videos para encontrar el **siguiente video en el curso**.
* Se obtiene el **video con el siguiente número de orden**.
* Si **no hay más videos**, la variable @idSiguienteVideo será NULL.

 **Si NO hay un siguiente video:**

* Habilita el examen final (examenHabilitado = 1) en la tabla Progresos para el usuario.
* Retorna los datos del **último video** que acaba de ver el usuario.
* **Finaliza el procedimiento (RETURN)**.

 **Si hay un siguiente video:**

* **Verifica si ya está activo (estado = 1)** en la tabla ProgresoVideo:
  + Si **ya está activado**, lo devuelve y **finaliza el procedimiento**.
* **Si el video aún no está activado:**
  + Si **no existe un registro en ProgresoVideo**, lo inserta con estado = 1.
  + Si ya existe un registro, simplemente actualiza el estado = 1.

 **Devuelve los datos del siguiente video activado**.

-- Devolver el siguiente video activado

SELECT \*

FROM Videos

WHERE id = @idSiguienteVideo;

***IdVideo (POST)***

* Recibe: idVideo (PARAMS) --- idUsuario, idCurso

 **Busca el ID del siguiente video**

* Se obtiene el video con el siguiente número de orden dentro del mismo curso.
* Si no hay un siguiente video, la variable @idVideoSiguiente será NULL.

 **Obtiene la información del video actual:**

* **v.id** → ID del video.
* **v.titulo** → Título del video.
* **v.link** → Enlace del video.
* **v.orden** → Posición del video en la secuencia del curso.
* **pv.VideoVisto** → Indica si el usuario ya ha visto el video (si existe en ProgresoVideo).
* **pv.estado** → Si el video está activado (1 si lo ha visto, 0 si no).
* **@idVideoSiguiente** → ID del siguiente video en la secuencia.

 **Filtra solo los videos que el usuario ya haya visto (pv.estado = 1)**

* Si el usuario no ha visto el video, no se devolverá información.

***VideoTerminado (PUT)***

* Recibe: idVideo (PARAMS) --- idUsuario, idCurso

 **Actualiza el progreso del curso (Progresos)**

* Incrementa en +1 el campo progreso, que representa cuántos videos ha completado el usuario en el curso.
* Solo se actualiza si ya existe un registro en Progresos para ese usuario y curso.

 **Marca el video como visto (ProgresoVideo)**

* Cambia VideoVisto a 1, indicando que el usuario ya ha terminado este video.