🧠 Lógica de Funcionamiento del Modelo de Datos – Proyecto Nikola Tesla

---

🔹 1. Entidad Usuario

Rol funcional:

Representa a la persona registrada en la plataforma (solo rol *estudiante* por defecto).

Campos principales:

Campo	Descripción	Tipo	Notas
`id`	Identificador único	int	PK
`nombre`	Nombre del usuario	varchar(100)	Obligatorio
`apellido`	Apellido del usuario	varchar(100)	Obligatorio
`correo`	Correo electrónico	varchar(150)	Único, usado para login
`numero_telefonico`	Teléfono opcional	varchar(15)	Puede ser NULL
`contraseña`	Clave encriptada	varchar(255)	Guardada con hashing
`fecha_registro`	Fecha de creación	datetime	Generada automáticamente

Lógica asociada:

- Cada usuario puede **registrarse** y **autenticarse** mediante su correo y contraseña.
- Un usuario puede **inscribirse en varios cursos** (relación N:M).
- Toda la información de progreso o inscripción depende del usuario.

🔹 2. Entidad Curso

Rol funcional:

Representa cada curso disponible (por ejemplo, *Python Básico*, *IA Aplicada*, etc.).

Campos principales:

Campo	Descripción	Tipo
`id`	Identificador único	int
`titulo`	Nombre del curso	varchar(150)
`descripcion`	Detalles del curso	text
`nivel`	Nivel de dificultad (Básico, Intermedio, Avanzado)	varchar(50)
`duracion`	Tiempo estimado (horas/semanas)	varchar(50)
`imagen`	URL o nombre de archivo	varchar(255)
`estado`	Indica si está activo o no	boolean

Lógica asociada:

- Cada curso puede **contener varios módulos** (1:N).
- Cada curso puede **tener muchos inscritos** (a través de la tabla `Inscripcion`).
- Un curso solo debe estar visible si `estado = true`.

🔹 3. Entidad Modulo

Rol funcional:

Divide el curso en partes más pequeñas o unidades temáticas.

Campos principales:

Campo	Descripción	Tipo
`id`	Identificador único	int
`curso_id`	FK del curso al que pertenece	int
`titulo`	Nombre del módulo	varchar(150)
`descripcion`	Breve explicación del módulo	text

Lógica asociada:

- Cada módulo pertenece **a un solo curso**.
- Los módulos sirven como **contenedores de contenidos** (videos, PDFs, etc.).
- Permite al estudiante avanzar módulo por módulo y medir progreso.

---

4. Entidad Contenido

Rol funcional:

Guarda los recursos didácticos de cada módulo (videos, lecturas, PDFs, ejercicios, etc.).

Campos principales:

Campo	Descripción	Tipo
`id`	Identificador único	int
`modulo_id`	FK hacia el módulo	int
`tipo`	Tipo de recurso (video, pdf, enlace, etc.)	varchar(50)
`url_archivo`	Dirección o nombre del archivo	varchar(255)

Lógica asociada:

- Cada contenido pertenece **a un solo módulo**.
- Permite que la app cargue dinámicamente el material de estudio.
- El Front-End puede listar los contenidos según el módulo activo.

5. Entidad Inscripcion

Rol funcional:

Registra la relación entre un estudiante y un curso.
Permite saber **quién está matriculado**, **desde cuándo** y **cuál es su progreso**.

Campos principales:

Campo	Descripción	Tipo
`id`	Identificador único	int
`usuario_id`	FK del usuario inscrito	int
`curso_id`	FK del curso inscrito	int
`fecha_inscripcion`	Fecha en que se matriculó	datetime
`progreso`	Porcentaje de avance	decimal(5,2)

Lógica asociada:

- Cada registro une un `usuario` con un `curso`.
- Un mismo usuario puede tener **múltiples inscripciones** (uno por curso).
- El campo `progreso` se actualiza según las actividades completadas.
- Si el curso se elimina, se eliminan las inscripciones asociadas (cascade delete).

🔁 6. Relaciones y flujo de información

Relación	Tipo	Descripción
Usuario ⇄ Inscripcion	1:N	Un usuario puede estar inscrito en varios cursos
Curso ⇄ Inscripcion	1:N	Un curso puede tener muchos estudiantes
Curso ⇄ Modulo	1:N	Un curso se divide en varios módulos
Modulo ⇄ Contenido	1:N	Cada módulo tiene varios recursos o materiales

📊 **Flujo general:**

1. El usuario se **registra** → se crea en la tabla `usuario`.
2. El usuario **visualiza cursos activos** desde la tabla `curso`.
3. Si desea inscribirse, se crea un registro en `inscripcion` (con `fecha_inscripcion` y `progreso = 0`).
4. Al ingresar a un curso, se consultan sus **módulos** y los **contenidos asociados**.
5. Conforme el usuario avanza, se actualiza el campo `progreso` en la inscripción.

🧱 7. Ejemplo de flujo lógico

Supongamos que Ana se registra:

- Se inserta un registro en `usuario` con sus datos.

Ana se inscribe en "Python Básico":

- Se crea un registro en `inscripcion` con `usuario_id = Ana`, `curso_id = Python Básico`.

Cuando Ana ve el **Módulo 1**, el sistema carga:

- De `modulo` todos los módulos del curso "Python Básico".
- De `contenido`, los archivos relacionados con ese módulo.

Cuando termina el 50% del curso:

- Se actualiza `inscripcion.progreso = 50.00`.