Sistema de elegibilidad de materias

El trabajo consiste en desarrollar un sistema de elegibilidad de materias que permita a los estudiantes conocer a qué asignaturas pueden anotarse según su historial académico. Para ello deberán modelar un esquema que contemple materias organizadas por semestre, con requisitos previos de dos tipos: examen aprobado y curso aprobado.

El sistema deberá incluir autenticación con diferentes roles (estudiante y administrador), un historial académico para cada usuario, un verificador de elegibilidad que indique las materias disponibles y explique las causas de no elegibilidad, así como un panel de administración para gestionar materias, previas y horarios.

Tecnologías a utilizar: Node.js, Express, MongoDB/Mongoose, JSON Web Tokens, EJS, WebSockets (socket.io).

Alcance funcional mínimo (MVP)

1. Autenticación y roles
   - Registro/inicio de sesión con JWT (expiración y refresh).
   - Roles: *Estudiante* y *Administrador*.
2. Modelo académico
   - Materia (código, nombre, créditos, semestre de dictado, horarios).
   - Previas: dos tipos de requisitos: *Examen aprobado* y *Curso aprobado*.
3. Historial del estudiante
   - Materias cursadas y aprobadas (con estado: pendiente,en curso, cursado (curso aprobado, examen pendiente), aprobado).
   - Importación de historial o carga manual.
   - Créditos obtenidos.
4. Verificador de elegibilidad
   - Dado el historial, calcular en qué materias puede anotarse (cumple previas definidas).
   - Mostrar causas de no elegibilidad por materia (p. ej., falta aprobar un examen o curso previo).
5. Panel Admin
   - ABM de materias y previas.
   - Gestión de horarios y semestres.
   - Vista del plan de estudios y dependencias.
6. Selección de materias.
   - Selección de materias para cursar durante el semestre que permita ver carga horaria y posibles choques de horario,

Requisitos técnicos y de calidad

- Seguridad: contraseñas con hash, JWT con expiración, rutas protegidas por rol, validación de inputs.
- Arquitectura: separación en capas (`/models`, `/services`, `/controllers`, `/routes`, `/views`, `/sockets`).
- EJS para vistas principales (auth, tablero estudiante, panel admin, listados).
- WebSockets para notificaciones de cambios (ejemplo: nueva materia disponible).
- Logs (morgan + logs de dominio) y manejo de errores centralizado.
- .env con variables: `MONGO_URI`, `JWT_SECRET`, `PORT`, etc.
- Datos seed para pruebas locales (`npm run seed`).
- Docs: README con setup, decisiones de diseño y endpoints.

Deploy

- Se debe entregar con link público en alguna de las siguiente plataformas o similar: Replit, Netlify, Vercel, Render,etc.