Guion Video Demostrativo – Periferia Social

1. Apertura (0:00 – 0:20)

- Presentar “Periferia Social” como la red social interna para colaboradores de Periferia IT.
- Mencionar el stack tecnológico: backend en Nest.js + Prisma, frontend en React + Vite, autenticación con JWT.
- Mostrar logo o pantalla de inicio de sesión; incluir una frase breve sobre el objetivo del proyecto (mejorar la comunicación interna).

2. Arquitectura (0:20 – 0:50)

- Exponer un diagrama simple que muestre `frontend`, `backend` y `postgres`.
- Explicar que el frontend consume la API REST (`http://localhost:3000/api`) y se sirve en `http://localhost:5173`.
- Comentar que Swagger está disponible en `http://localhost:3000/api/docs` para explorar endpoints protegidos con Bearer Token.

3. Preparación del entorno (0:50 – 1:30)

- En pantalla de terminal, demostrar:
  - `git clone` del repositorio.
  - Duplicar `env.example` a `.env` y revisar variables clave (`DATABASE_URL`, puertos, `JWT_SECRET`).
  - Recordar los prerrequisitos: Node 20+, npm 10+, Docker y Docker Compose.
- Ejecutar `docker compose up --build`, mostrar los contenedores levantándose y los logs “Executing seed” e “Iniciando servidor Nest”.

4. Backend en acción (1:30 – 2:20)

- Abrir Swagger en `http://localhost:3000/api/docs`.
- Probar `POST /api/auth/login` con `ana.romero@periferia.it` y contraseña `Periferia123!`.
- Copiar el token y usarlo para `GET /api/users/me`.
- Finalizar mostrando `GET /api/posts` y explicar que los likes son idempotentes.

5. Frontend: flujo principal (2:20 – 4:00)

- Navegador en `http://localhost:5173`:
  - Inicio de sesión con las credenciales de Ana.
  - Mostrar feed con publicaciones y contador de likes.
  - Crear una nueva publicación y confirmar que aparece en el feed.
  - Dar like a un post y evidenciar la actualización inmediata.
  - Visitar `ProfilePage` para revisar información personal sincronizada con el backend.
- Resaltar el uso de Zustand para el estado global y llamadas autenticadas al API.

6. Proceso de desarrollo (4:00 – 4:40)

- Mostrar editor (VSCode) en modo split:
  - `backend/src/posts/posts.controller.ts`: comentar la organización modular, DTOs y validaciones.

    - `frontend/src/pages/ProfilePage.tsx`: destacar componentes reutilizables, hooks y tipado en TypeScript.
- Subrayar la consistencia del stack TypeScript y la estructura clara en carpetas (`auth`, `users`, `posts`).

7. Pruebas automatizadas (4:40 – 5:20)

- En terminal:
    - `cd backend`, `npm install`, `npm run test`; mostrar salida de Jest.
    - `cd frontend`, `npm install`, `npm run test -- --run`; mostrar salida de Vitest.
- Mencionar comandos de cobertura (`npm run test:cov`, `npm run test:coverage`) y la posibilidad de correr pruebas dentro de Docker (`docker compose exec api/frontend …`).

8. Buenas prácticas DevOps (5:20 – 5:50)

- Recordar scripts útiles:
    - Backend: `npm run prisma:migrate:dev`, `npm run prisma:seed`, `npm run build`, `npm run start:prod`.
    - Frontend: `npm run build`, `npm run lint`, `npm run dev`.
- Destacar que Docker Compose orquesta API, frontend y base de datos, y que los seeds cargan usuarios de prueba automáticamente.

9. Cierre (5:50 – 6:10)

- Recapitular beneficios: solución full-stack lista para uso interno, autenticación segura, feed colaborativo, pruebas automatizadas y despliegue con un comando.
- Invitar a clonar el repositorio y personalizar `.env`.
- Terminar con diapositiva de despedida y enlace al repositorio.

Recomendaciones de producción

- Grabar en 1080p con ritmo ágil; alternar entre presentador y screencast.
- Añadir rótulos breves para comandos (`docker compose up --build`) y URLs clave.
- Utilizar zooms o resaltados para código importante.
- Incluir subtítulos cuando se muestren credenciales o comandos.
- Usar música de fondo suave y reducir volumen cuando se hable.
- Cerrar con agradecimiento y recordatorio del repositorio.