Guía de Proyectos Finales

Curso: Desarrollo Web con ASP.NET MVC y PostgreSQL

Evaluación: 20 puntos
Despliegue en contenedores con Docker y DigitalOcean

Proyecto 1 – FreelancerHub

Mini plataforma donde clientes publican proyectos y consultores aplican con propuestas. Flujo básico de publicación, postulación y contratación.

Requerimientos:

- Autenticación con Identity (registro, login, logout).
- Roles: Admin, Cliente, Consultor.
- CRUD Proyectos con estado.
- Postulaciones y aceptación de propuesta.
- Mensajería básica entre Cliente y Consultor.
- Dashboard por rol.
- Validaciones y autorización por rol.

Modelo de Datos (mínimo sugerido):

- ApplicationUser
- Proyecto {Id, Titulo, Descripcion, PresupuestoQ, FechaLimite, Estado, ClienteId}
- Postulacion {Id, MontoQ, Propuesta, Fecha, ProyectoId, ConsultorId, Aceptada}
- Mensaje {Id, Texto, Fecha, ProyectoId, RemitenteId}

Criterios de Aceptación:

- Cliente puede crear proyecto, consultor aplicar y cliente aceptar propuesta.
- Roles y permisos funcionales.
- UI clara con Bootstrap 5.

Entregables:

- Repositorio GitHub con código.
- Migraciones EF Core.
- Dockerfile y docker-compose.yml.
- Subdominio activo: freelancer.tudominio.com.

Proyecto 2 – MesaListo

Sistema de reservas de restaurantes con roles Restaurante y Cliente. Permite registrar restaurantes, mesas y reservas con validación de disponibilidad.

Requerimientos:

- Identity + roles (Admin, Restaurante, Cliente).
- CRUD Restaurante y Mesa.
- Reserva con validación de disponibilidad.
- Gestión de Reseñas post-reserva.
- Dashboard por restaurante.
- Correo de confirmación (MailKit o consola).

Modelo de Datos (mínimo sugerido):

- Restaurante {Id, Nombre, Direccion, Telefono}
- Mesa {Id, Codigo, Capacidad, RestauranteId}
- Reserva {Id, FechaHora, MesaId, ClienteId, Estado}
- Resena {Id, Puntuacion, Comentario, ReservaId, ClienteId}

Criterios de Aceptación:

- Reserva exitosa y única por horario/mesa.
- Dashboard muestra ocupación semanal.
- UI con Bootstrap.

Entregables:

- Repo con código fuente.
- Dockerfile y compose.
- Subdominio: mesas.tudominio.com.

Proyecto 3 – EduMentor

Portal de microcursos con módulos cortos. Estudiantes se inscriben, completan módulos y generan certificados en PDF.

Requerimientos:

- Identity + roles (Admin, Instructor, Estudiante).
- CRUD Curso y Módulo.
- Inscripción y progreso de módulos.
- Dashboard estudiante e instructor.
- Certificado PDF al 100% completado.

Modelo de Datos (mínimo sugerido):

- Curso {Id, Titulo, Descripcion, Nivel}
- Modulo {Id, CursoId, Titulo, Contenido, Orden}
- Inscripcion {Id, CursoId, EstudianteId, Fecha}
- ProgresoModulo {Id, ModuloId, EstudianteId, FechaCompletado}

Criterios de Aceptación:

- Registro e inscripción funcional.
- Porcentaje de avance correcto.
- Generación PDF funcional.

Entregables:

- Repo con código y migraciones.
- Despliegue en Docker.
- Subdominio: edu.tudominio.com.

Proyecto 4 – TeamPoints

Sistema de puntos y recompensas internas. RH asigna puntos por logros y empleados los canjean por premios.

Requerimientos:

- Identity + roles (Admin, RH, Colaborador).
- CRUD Logros y Premios.
- Transacciones de puntos (ganar/canjear).
- Leaderboard semanal.
- Historial de transacciones.

Modelo de Datos (mínimo sugerido):

- TransaccionPuntos {Id, UsuarioId, Tipo, Puntos, Fecha, Ref}
- Logro {Id, Nombre, Puntos}
- Premio {Id, Nombre, CostoPuntos}

Criterios de Aceptación:

- Asignación y canje funcional.
- Leaderboard actualizado.
- Seguridad y UI limpia.

Entregables:

- Repositorio en GitHub.

- Dockerfile, compose y despliegue.
- Subdominio: points.tudominio.com.

Rúbrica de Evaluación (20 puntos)

Criterio	Ponderación
Seguridad (Identity + Roles + Autorización)	4 pts
Modelo de datos y migraciones	3 pts
CRUDs funcionales con scaffolding	4 pts
Lógica de negocio mínima funcional	5 pts
Interfaz y usabilidad (Bootstrap)	2 pts
Despliegue con subdominio y HTTPS	2 pts