Desafío de Desarrollo ASE Athletics

Evaluación de Desarrollador Full Stack

> **¡Bienvenido a tu evaluación técnica!** Este documento describe todo lo que necesitas para construir una impresionante plataforma de análisis de fútbol que demuestre tus habilidades de desarrollo full-stack.

Guía de Inicio Rápido

Tu Misión: Construir una aplicación web de análisis de fútbol
Cronograma: 10 días máximo *(¡bonificación por entrega temprana!)*
Puntuación: 280 puntos totales disponibles
Entregables: Repositorio GitHub + demostración en vivo

Cómo Se Ve el Éxito

- Una aplicación web funcional con autenticación de usuario
- Sistema completo de gestión de jugadores con integración de base de datos
- Panel interactivo con gráficos significativos
- Búsqueda y filtrado que realmente funcione
- Código limpio y profesional que nos alegraría mantener

El Proyecto: Plataforma de Análisis de Fútbol

Construirás un espacio de trabajo digital para ojeadores y entrenadores de fútbol - piensa en herramientas como las que usan los clubes profesionales para analizar jugadores y tomar decisiones estratégicas.

Concepto Central

Los clubes de fútbol modernos dependen en gran medida del análisis de datos. Tu aplicación ayudará a los profesionales del fútbol a:

- **Gestionar bases de datos de jugadores** con estadísticas completas
- **Visualizar tendencias de rendimiento** a través de paneles interactivos
- **Comparar jugadores** lado a lado para decisiones de reclutamiento
- **Generar reportes de scouting** con evaluaciones estructuradas
- **Buscar y filtrar** grandes conjuntos de datos eficientemente

Por Qué Esto Importa en ASE Athletics

Esta evaluación refleja proyectos reales en los que trabajarás en ASE Athletics. Desarrollamos herramientas que ayudan a las organizaciones deportivas a tomar decisiones basadas en datos, y este desafío refleja el tipo de problemas que resolverás diariamente.

Tu Stack de Desarrollo

Tecnologías Requeridas

Framework Frontend *(elige uno)*

- React.js con Redux/Context API
- Vue.js con Vuex
- Debe incluir enrutamiento apropiado (React Router/Vue Router)

Framework Backend *(requerido)*

- Node.js con Express o NestJS
- Principios de diseño de API RESTful
- Autenticación basada en JWT
- Validación de entrada (Joi, Yup, o similar)

Base de Datos *(elige una)*

- PostgreSQL *(recomendado para datos relacionales)*
- MongoDB *(bueno para esquemas flexibles)*
- MySQL o SQLite *(alternativas aceptables)*

Solución de Estilos *(elige una)*

- Tailwind CSS *(moderno utility-first)*
- Material-UI *(librería de componentes React)*
- Bootstrap *(familiar y confiable)*
- CSS personalizado *(muestra tus habilidades de diseño)*

Visualización de Datos *(elige una)*

- Chart.js *(simple y efectivo)*
- D3.js *(poderoso y flexible)*
- Recharts *(específico para React)*

Requisitos de Arquitectura

- **Diseño responsivo** que funcione en móvil y escritorio
- **Separación limpia** entre frontend y backend
- **Manejo adecuado de errores** en toda la aplicación
- **Mejores prácticas de seguridad** para autenticación y validación de datos
- **Documentación de API** usando Swagger/OpenAPI o Postman

Activos de Datos Proporcionados por ASE Athletics

Hemos preparado conjuntos de datos completos para impulsar tu desarrollo:

Base de Datos de Jugadores

Archivos: `player_statistics_detailed.json` + `players_Data_production.json`

- **100+ jugadores profesionales** con perfiles completos
- **Métricas de rendimiento:** goles, asistencias, apariciones, precisión de pases
- **Atributos físicos:** velocidad, tiro, pase, calificaciones defensivas
- **Detalles de contrato:** información salarial, fechas de finalización de contrato
- **Valuaciones de mercado** e historial de transferencias

Marco de Scouting

Archivo: `scout_report.json`

- **Plantillas de reportes** usadas por scouts profesionales
- **Sistema de calificación:** escala estandarizada de 1-10 en todos los atributos
- **Estructura de evaluación:** contexto del partido, fortalezas, debilidades, recomendaciones

Sistema de Diseño

Archivo: `ui_guidelines.json`

- **Paleta de colores completa** con códigos hex y pautas de uso
- **Especificaciones tipográficas:** familias de fuentes, tamaños, pesos
- **Estándares de componentes:** botones, tarjetas, formularios, tablas
- **Pautas de espaciado y diseño** para consistencia

Todos los archivos disponibles a través del repositorio GitHub de ASE Athletics o un ZIPFILE proporcionado

---

Especificación de Características y Puntuación

Sistema de Autenticación Central (15 puntos)

Construye un sistema seguro de gestión de usuarios con:

- **Flujo de registro** con validación de email
- **Autenticación de login** usando tokens JWT
- **Rutas protegidas** que requieren autenticación
- **Gestión de sesiones** con logout apropiado
- **Seguridad de contraseñas** usando hash bcrypt

Mantenlo simple - autenticación básica email/contraseña sin sistemas complejos de roles

Arquitectura de API Backend (25 puntos)

Desarrolla una API RESTful con estos endpoints esenciales:

Gestión de Jugadores

- `GET /api/players` - Lista paginada de jugadores con filtrado
- `GET /api/players/:id` - Detalles de jugador individual
- `POST /api/players` - Crear nuevo perfil de jugador
- `PUT /api/players/:id` - Actualizar jugador existente
- `DELETE /api/players/:id` - Eliminar jugador del sistema

Búsqueda y Descubrimiento

- `GET /api/players/search?q={term}` - Búsqueda de texto en datos de jugadores
- `GET /api/players?position={pos}&team={team}` - Filtrado multi-parámetro

Soporte de Análisis

- `GET /api/dashboard/stats` - Métricas agregadas para panel

Enfócate en operaciones CRUD sólidas con validación apropiada y respuestas de error

Diseño y Gestión de Base de Datos (20 puntos)

Crea un esquema de base de datos bien estructurado:

Tablas Esenciales

```
-- Autenticación de usuario
users: id, email, password_hash, name, created_at

-- Datos centrales de jugador
players: id, name, position, age, team, nationality,
         height, weight, goals, assists, appearances,
         contract_salary, contract_end, market_value,
         created_at, updated_at

-- Atributos de jugador (tabla separada opcional)
player_attributes: player_id, pace, shooting, passing,
                   defending, dribbling, physicality

-- Reportes de scouting
scout_reports: id, player_id, scout_id, match_date,
               overall_rating, strengths, weaknesses,
               recommendation, created_at
```

Requisitos de Base de Datos

- **Diseño de esquema** con relaciones apropiadas
- **Scripts de siembra de datos** para cargar archivos JSON proporcionados
- **Indexación básica** para rendimiento de búsqueda
- **Archivos de migración** para configuración de base de datos

Interfaz de Gestión de Jugadores (25 puntos)

Construye un sistema completo de gestión de jugadores:

Directorio de Jugadores

- **Diseño de cuadrícula/tabla** mostrando información esencial de jugadores
- **Controles de paginación** (20-30 jugadores por página)
- **Columnas ordenables** por nombre, edad, posición, equipo, valor de mercado

- **Imágenes de perfil de jugador** (funcionalidad placeholder aceptable)

Perfiles Detallados de Jugadores

- **Información completa del jugador** con estadísticas y atributos
- **Visualización de rendimiento** usando barras de progreso o mini-gráficos
- **Información de contrato y valor de mercado**
- **Capacidades de edición** para actualizar datos de jugador

Creación y Edición de Jugadores

- **Interfaz basada en formularios** con validación apropiada
- **Todos los campos requeridos** con tipos de entrada apropiados
- **Manejo de errores** y retroalimentación de éxito
- **Placeholder de subida de imagen** (subida real es bonificación)

Eliminación Segura

- **Diálogos de confirmación** antes de eliminar jugadores
- **Manejo en cascada** para datos relacionados (reportes, etc.)

Panel de Análisis (25 puntos)

Crea un panel interactivo con insights significativos:

Indicadores Clave de Rendimiento

- **Tarjetas de resumen** mostrando total de jugadores, edad promedio, mejores rendimientos
- **Insights de mercado** como jugadores más valiosos, expiraciones de contrato
- **Distribuciones de equipos** y desglose de posiciones

Visualizaciones Interactivas

- **Distribución de goles/asistencias** por posición (gráficos de barras)
- **Demografía de edad** entre equipos (gráficos de pastel/dona)
- **Tendencias de valor de mercado** a lo largo del tiempo (gráficos de líneas)
- **Comparaciones de atributos de jugadores** (gráficos de radar)

Filtrado Dinámico

- **Actualizaciones de gráficos en tiempo real** basadas en filtros seleccionados
- **Controles de filtro** de equipo/posición/rango de edad
- **Funcionalidad de limpiar filtros** para resetear vistas
- **Capacidades de exportación** para datos de gráficos (bonificación)

Sistema de Búsqueda y Filtrado (25 puntos)

Implementa características completas de búsqueda y descubrimiento:

Búsqueda Global

- **Búsqueda instantánea** en nombres de jugadores, equipos, nacionalidades

- **Resaltado de resultados de búsqueda** para mostrar términos coincidentes
- **Sugerencias de búsqueda** o autocompletado (bonificación)

Opciones de Filtrado Avanzado

- **Filtro de posición** con capacidades multi-selección
- **Deslizador de rango de edad** para filtrado demográfico
- **Selección de equipo** con dropdown o multi-selección
- **Filtro de nacionalidad** para análisis internacional
- **Filtrado de rango de valor de mercado**
- **Filtros de rango de estadísticas de rendimiento** (goles, asistencias)

Gestión de Filtros

- **Combinar múltiples filtros** simultáneamente
- **Indicadores de filtros activos** mostrando selecciones actuales
- **Opciones de limpieza rápida** para filtros individuales o todos
- **Persistencia de estado de filtros** durante la sesión (bonificación)

Herramienta de Comparación de Jugadores (25 puntos)

Construye un sistema de análisis lado a lado de jugadores:

Interfaz de Selección de Jugadores

- **Funcionalidad multi-selección** para 2-4 jugadores
- **Arrastrar y soltar** o selección basada en checkbox
- **Búsqueda dentro de la herramienta de comparación** para encontrar jugadores rápidamente

Visualizaciones de Comparación

- **Tabla de estadísticas lado a lado** con métricas clave
- **Superposición de gráfico de radar** mostrando comparaciones de atributos
- **Gráficos de tendencia de rendimiento** para goles, asistencias a lo largo del tiempo
- **Comparación de valor de mercado** con datos históricos

Características Interactivas

- **Alternar entre categorías de métricas** (rendimiento, atributos, mercado)
- **Exportar comparación** como imagen o PDF (bonificación)
- **URLs de comparación compartibles** (bonificación)

Sistema de Reportes de Scouting (10 puntos)

Implementa creación y gestión básica de reportes:

Creación de Reportes

- **Selección de jugador** desde dropdown o búsqueda
- **Entrada de contexto de partido** (fecha, oponente, lugar)
- **Sistema de calificación de atributos** con deslizadores 1-10 o calificaciones de estrellas
- **Áreas de texto** para fortalezas, debilidades, evaluación general

- **Recomendación general** (fichar, monitorear, pasar)

Gestión de Reportes

- **Vista de lista** de todos los reportes con filtrado por jugador/scout
- **Editar reportes existentes** con pre-población de formularios
- **Eliminar reportes** con confirmación
- **Funcionalidad de exportar a PDF** (bonificación)

Implementación de Diseño Responsivo (10 puntos)

Asegúrate de que tu aplicación funcione sin problemas en todos los dispositivos:

Enfoque Mobile-First

- **Interfaces amigables al tacto** con tamaños de botón apropiados
- **Navegación colapsable** para pantallas móviles
- **Tablas responsivas** que funcionen en pantallas pequeñas
- **Displays de gráficos optimizados** para visualización móvil

Consistencia Cross-Platform

- **Espaciado consistente** y tipografía en todos los breakpoints
- **Texto legible** en todos los tamaños de pantalla
- **Contraste de color accesible** cumpliendo pautas WCAG
- **Tiempos de carga rápidos** con activos optimizados

Estándares de Calidad de Código

Arquitectura Limpia (60 puntos total)

Organización de Código (20 puntos)

- **Estructura de archivos lógica** con separación clara de responsabilidades
- **Componentes reutilizables** que siguen principios DRY
- **Convenciones de nomenclatura consistentes** para archivos, funciones, variables
- **Diseño modular** que es fácil de mantener y extender

Manejo de Errores (15 puntos)

- **Mensajes de error amigables** que guían a los usuarios
- **Manejo de errores de API** con códigos de estado HTTP apropiados
- **Fallbacks elegantes** cuando los datos no están disponibles
- **Estados de carga** para mejor experiencia de usuario

Documentación de Código (10 puntos)

- **README claro** con instrucciones de configuración y ejecución
- **Documentación de API** con descripciones de endpoints
- **Comentarios de código** explicando lógica compleja
- **Guía de configuración de entorno** con ejemplos

Prácticas de Código Limpio (15 puntos)

- **Código legible** que sigue convenciones del lenguaje
- **Niveles de abstracción apropiados** en toda la aplicación
- **Formato consistente** usando linters (ESLint, Prettier)
- **Mejores prácticas de seguridad** para autenticación y manejo de datos

---

Excelencia en Implementación Técnica

Integración de Base de Datos (15 puntos)

- **Uso apropiado de ORM/ODM** (Sequelize, Mongoose, etc.)
- **Consultas eficientes** que no causan problemas de rendimiento
- **Validación de datos** a nivel de base de datos
- **Pooling de conexiones** y manejo de errores

Calidad de Diseño de API (15 puntos)

- **Principios RESTful** con métodos HTTP apropiados
- **Formatos de respuesta consistentes** en todos los endpoints
- **Códigos de estado apropiados** para diferentes escenarios
- **Validación de solicitudes** con mensajes de error significativos

Integración Frontend-Backend (15 puntos)

- **Flujo de datos suave** entre cliente y servidor
- **Gestión de estado apropiada** en el frontend
- **Capas de servicio de API** que manejan errores elegantemente
- **Actualizaciones en tiempo real** donde sea apropiado

Oportunidades de Bonificación (30 puntos)

Implementación de Pruebas (10 puntos)

- **Pruebas unitarias** para lógica de negocio crítica
- **Pruebas de endpoints de API** con datos de prueba apropiados
- **Pruebas de componentes frontend** para características clave
- **Pruebas de integración** demostrando flujos completos

Características Avanzadas (10 puntos)

- **Funcionalidad de exportación de datos** (CSV, Excel, PDF)
- **Búsqueda avanzada** con filtros y ordenamiento
- **Actualizaciones en tiempo real** usando WebSockets
- **Optimización de rendimiento** con caché

Excelencia en Despliegue (10 puntos)

- **Despliegue de producción** de frontend y backend
- **Configuración de entorno** con gestión apropiada de secretos
- **Alojamiento de base de datos** con gestión de conexiones
- **Configuración de CDN** para activos estáticos (bonificación)

Recomendación de Estructura de Proyecto

```
ase-athletics-assessment/
├── frontend/                    # Aplicación React/Vue
│   ├── public/
│   ├── src/
│   │   ├── components/         # Componentes UI reutilizables
│   │   │   ├── common/         # Botones, inputs, layouts
│   │   │   ├── charts/         # Componentes de gráficos
│   │   │   └── forms/          # Componentes de formularios
│   │   ├── pages/              # Páginas principales de aplicación
│   │   │   ├── Dashboard/      # Panel de análisis
│   │   │   ├── Players/        # Gestión de jugadores
│   │   │   ├── Comparison/     # Comparación de jugadores
│   │   │   └── Reports/        # Reportes de scouting
│   │   ├── services/           # Comunicación con API
│   │   ├── store/              # Gestión de estado
│   │   ├── utils/              # Funciones auxiliares
│   │   └── styles/             # Estilos globales
│   ├── package.json
│   └── .env.example
├── backend/                     # Servidor API Node.js
│   ├── src/
│   │   ├── controllers/        # Manejadores de rutas
│   │   ├── models/             # Modelos de base de datos
│   │   ├── routes/             # Definiciones de rutas API
│   │   ├── middleware/         # Auth, validación, etc.
│   │   ├── services/           # Lógica de negocio
│   │   └── utils/              # Funciones auxiliares
│   ├── migrations/             # Migraciones de base de datos
│   ├── seeds/                  # Scripts de siembra de datos
│   ├── tests/                  # Pruebas de API
│   ├── package.json
│   └── .env.example
├── data/                       # Archivos JSON proporcionados
│   ├── player_statistics_detailed.json
│   ├── players_Data_production.json
│   ├── scout_report.json
│   └── ui_guidelines.json
├── docs/                       # Documentación del proyecto
│   ├── api-documentation.md
│   ├── database-schema.md
│   └── deployment-guide.md
├── docker-compose.yml          # Configuración de desarrollo local
├── README.md                   # Documentación principal del proyecto
└── .gitignore
```

Requisitos de Entrega

Lista de Verificación del Repositorio GitHub

- [ ] **Código fuente completo** para frontend y backend
- [ ] **README completo** con instrucciones de configuración
- [ ] **Ejemplos de configuración de entorno** (archivos .env.example)
- [ ] **Instrucciones de configuración de base de datos** y archivos de migración
- [ ] **Documentación de API** (Swagger UI o markdown detallado)
- [ ] **Historial de commits limpio** mostrando progresión de desarrollo
- [ ] **.gitignore apropiado** excluyendo archivos sensibles

Requisitos de Demostración en Vivo

- [ ] **Aplicación frontend desplegada** (Netlify, Vercel, etc.)
- [ ] **API backend desplegada** (Heroku, Railway, DigitalOcean)
- [ ] **Base de datos alojada** (PostgreSQL, MongoDB Atlas, etc.)
- [ ] **Todas las características funcionales** en entorno de producción
- [ ] **Credenciales de demostración** proporcionadas para pruebas

Estándares de Documentación

- [ ] **Configuración de desarrollo local** con instrucciones paso a paso
- [ ] **Documentación de esquema de base de datos** con relaciones explicadas
- [ ] **Documentación de endpoints de API** con ejemplos de solicitud/respuesta
- [ ] **Justificación de stack tecnológico** explicando tus elecciones
- [ ] **Resumen de características** con capturas de pantalla o GIFs
- [ ] **Documentación de proceso de despliegue**

Plantilla de README para tu Entrega

```
# ASE Athletics – Plataforma de Análisis de Fútbol

## Demostración en Vivo
- **Aplicación Frontend:** [Tu URL de frontend desplegado]
- **API Backend:** [Tu URL de backend desplegado]
- **Documentación de API:** [Swagger UI o URL de documentación]

## Resumen del Proyecto
[Breve descripción de tu implementación y características clave]

## Stack Tecnológico

### Frontend
- Framework: React.js / Vue.js
- Gestión de Estado: Redux / Context API / Vuex
```

```
- Estilos: Tailwind CSS / Material-UI / Bootstrap
- Gráficos: Chart.js / D3.js / Recharts
- Librerías adicionales: [Lista dependencias clave]

### Backend
- Runtime: Node.js
- Framework: Express / NestJS
- Base de Datos: PostgreSQL / MongoDB / MySQL
- Autenticación: JWT con bcrypt
- Validación: Joi / Yup
- Documentación: Swagger / OpenAPI

### DevOps y Despliegue
- Host Frontend: Netlify / Vercel
- Host Backend: Heroku / Railway
- Host Base de Datos: [Tu solución de alojamiento de base de datos]
- Control de Versiones: Git con GitHub

## Configuración de Desarrollo Local

### Prerequisitos
```bash
Node.js (v16 o superior)
PostgreSQL / MongoDB
Git
```

Configuración Backend

```
# Clonar y navegar
git clone [tu-url-repo]
cd backend

# Instalar dependencias
npm install

# Configuración de entorno
cp .env.example .env
# Editar .env con tus credenciales de base de datos

# Configuración de base de datos
npm run db:create
npm run db:migrate
npm run db:seed

# Iniciar servidor de desarrollo
npm run dev
# Servidor corre en http://localhost:5000
```

Configuración Frontend
```

```
# Navegar a frontend
cd frontend

# Instalar dependencias
npm install

# Configuración de entorno
cp .env.example .env
# Establecer REACT_APP_API_URL=http://localhost:5000

# Iniciar servidor de desarrollo
npm start
# Aplicación corre en http://localhost:3000
```

Esquema de Base de Datos

[Incluir un diagrama visual o descripción de texto detallada]

Endpoints de API

Autenticación

- `POST /api/auth/register` - Registro de usuario
- `POST /api/auth/login` - Login de usuario
- `POST /api/auth/logout` - Logout de usuario

Jugadores

- `GET /api/players` - Obtener jugadores (con paginación/filtrado)
- `GET /api/players/:id` - Obtener jugador específico
- `POST /api/players` - Crear nuevo jugador
- `PUT /api/players/:id` - Actualizar jugador
- `DELETE /api/players/:id` - Eliminar jugador

Panel

- `GET /api/dashboard/stats` - Métricas del panel

Reportes

- `GET /api/reports` - Obtener reportes de scouting
- `POST /api/reports` - Crear reporte
- `PUT /api/reports/:id` - Actualizar reporte
- `DELETE /api/reports/:id` - Eliminar reporte

Lista de Verificación de Implementación de Características

- [ ] Autenticación de Usuario y Gestión de Sesiones
- [ ] Operaciones CRUD Completas de Jugadores

- [ ] Panel de Análisis Interactivo
- [ ] Búsqueda y Filtrado Avanzado
- [ ] Herramienta de Comparación de Jugadores
- [ ] Sistema de Reportes de Scouting
- [ ] Diseño Responsivo
- [ ] Funcionalidad de Exportación de Datos

Pruebas

```
# Pruebas backend
cd backend && npm test

# Pruebas frontend
cd frontend && npm test
```

Notas de Despliegue

[Explicar tu proceso de despliegue y cualquier configuración específica]

Credenciales de Demostración

- Usuario: demo@ase-athletics.com
- Contraseña: demo123

Consideraciones de Rendimiento

[Discutir cualquier decisión de optimización que tomaste]

Mejoras Futuras

[Ideas para características adicionales o mejoras]

Estrategia de Desarrollo y Cronograma

Semana 1: Fundación (Días 1-3)

Día 1: Configuración del Proyecto

- Inicializar repositorios de frontend y backend
- Configurar entorno de desarrollo y dependencias
- Configurar base de datos y estructura básica del proyecto
- Implementar sistema básico de autenticación

Día 2: Backend Central

- Diseñar e implementar esquema de base de datos
- Crear scripts de siembra de datos para archivos JSON proporcionados
- Construir endpoints esenciales de API para CRUD de jugadores

- Implementar middleware de autenticación JWT

Día 3: Frontend Básico

- Configurar framework frontend y enrutamiento
- Crear páginas de autenticación (login/registro)
- Construir páginas básicas de lista y detalle de jugadores
- Implementar capa de servicio de API

Semana 2: Características (Días 4-7)

Día 4: Gestión de Jugadores

- Completar interfaz CRUD de jugadores
- Implementar funcionalidad de búsqueda y filtrado
- Agregar validación de formularios y manejo de errores
- Crear diseño responsivo para móvil

Día 5: Panel de Análisis

- Construir panel con métricas clave
- Implementar componentes de gráficos
- Agregar filtrado interactivo para gráficos
- Asegurar que la visualización de datos sea significativa

Día 6: Características Avanzadas

- Construir herramienta de comparación de jugadores
- Implementar sistema de reportes de scouting
- Agregar funcionalidad de exportación de datos
- Pulir interfaz de usuario e interacciones

Día 7: Pruebas y Despliegue

- Probar todas las características minuciosamente
- Desplegar frontend y backend a producción
- Verificar que toda la funcionalidad funcione en producción
- Crear documentación completa

Días Finales (8-10): Pulir y Documentación

- **Revisión de código** y refactorización
- **Optimización de rendimiento** donde sea necesario
- **Completar documentación** con capturas de pantalla
- **Pruebas finales** y corrección de errores

Métricas de Éxito y Evaluación

Qué Hace una Entrega Ganadora

Excelencia Técnica

- Todas las características principales funcionan confiablemente
- Estructura de código limpia y mantenible
- Manejo apropiado de errores en todo el sistema
- Diseño responsivo que funciona en móvil

Experiencia de Usuario

- Navegación e interfaz intuitiva
- Tiempos de carga rápidos e interacciones suaves
- Visualizaciones de datos significativas
- Apariencia profesional siguiendo pautas de diseño

Calidad de Código

- Estructura de proyecto bien organizada
- Estándares de codificación consistentes
- Separación apropiada de responsabilidades
- Documentación clara e instrucciones de configuración

Éxito en Despliegue

- Frontend y backend funcionando en producción
- Base de datos apropiadamente configurada y sembrada
- Todas las características funcionales en entorno en vivo
- Credenciales de demostración proporcionadas para pruebas fáciles

Trampas Comunes a Evitar

Problemas Técnicos

- Implementación incompleta de autenticación
- Falta de manejo de errores para fallas de API
- Diseño pobre de base de datos sin relaciones
- Diseño no responsivo que se rompe en móvil

Problemas de Calidad de Código

- Organización inconsistente de archivos/carpetas
- Valores hardcodeados en lugar de variables de entorno
- Sin validación de entrada en formularios
- Documentación faltante o poco clara

Fallas de Despliegue

- Frontend desplegado pero backend no funcionando
- Base de datos no sembrada apropiadamente con datos de prueba
- Variables de entorno no configuradas correctamente
- Endpoints de API devolviendo errores en producción

Soporte y Preguntas

Soporte Técnico

Si encuentras problemas con los archivos de datos proporcionados o tienes preguntas sobre requisitos técnicos, por favor contacta a nuestro equipo de desarrollo. Queremos que tengas éxito y estamos aquí para ayudar a aclarar cualquier confusión.

Proceso de Evaluación

1. **Revisión Inicial:** Probaremos tu demostración en vivo y revisaremos tu código
2. **Evaluación Técnica:** Evaluación contra los criterios de puntuación
3. **Revisión de Calidad de Código:** Evaluación de estructura y mejores prácticas
4. **Selección Final:** Los mejores candidatos serán invitados a entrevistas

Próximos Pasos

Los candidatos seleccionados pasarán a entrevistas finales donde discutiremos:

- Tus decisiones técnicas y compromisos
- Cómo abordaste la resolución de problemas
- Tu experiencia construyendo la aplicación
- Cómo encajarías con el equipo de ASE Athletics

Notas Finales

Esta evaluación representa el tipo de trabajo que harás en **ASE Athletics**. Estamos buscando desarrolladores que puedan construir aplicaciones prácticas y amigables que resuelvan problemas reales de negocio en el espacio de análisis deportivo.

Recuerda: Valoramos software funcional sobre código perfecto. Enfócate en construir algo funcional y bien documentado en lugar de tratar de implementar cada característica posible.

¡Buena suerte! Estamos emocionados de ver tu implementación y potencialmente darte la bienvenida al equipo de desarrollo de ASE Athletics.

Equipo de Desarrollo ASE Athletics
Evaluación Técnica v2.1 - Julio 2025