(v0.0.1 - 28-05-2024)

Team Lead: Ramiro Grisales

Prueba de desempeño NestJS

Propósito:

Como equipo de formación Riwi, queremos evaluar las habilidades de desarrollo backend en NestJS de un aspirante a Junior developer, específicamente en el diseño e implementación de API RESTful, manejo de bases de datos relacionales, y aplicación de buenas prácticas de NestJS. Esto nos permitirá determinar si el candidato posee los conocimientos técnicos y la capacidad de resolución de problemas necesarios para contribuir efectivamente en proyectos reales y en entornos laborales.

Metodología:

La prueba técnica se estructurará como una historia de usuario Scrum con una duración estimada de 8 horas. Esta metodología ágil permite simular un entorno de trabajo realista, donde los desarrolladores reciben tareas con plazos definidos y deben gestionar su tiempo y recursos de manera eficiente.

Se permitirá el uso de documentación oficial, ejemplos de código abierto y herramientas de IA como apoyo, fomentando así la investigación y el aprendizaje autónomo. Sin embargo, se enfatizará la importancia de comprender y explicar el código generado, evitando el plagio y la dependencia excesiva de herramientas externas.

La evaluación se centrará en la capacidad del candidato para aplicar los conocimientos teóricos en la práctica, resolver problemas de manera creativa y demostrar un dominio sólido de las tecnologías y conceptos fundamentales de NestJS.

Esta metodología no solo evalúa las habilidades técnicas del candidato, sino también su capacidad de adaptación, gestión del tiempo y aprendizaje autónomo, cualidades esenciales para un Junior developer exitoso.

Política Buen uso de la IA e Internet

Buen Uso de la IA

- **Asistencia en Código:** La IA se podrá utilizar para obtener sugerencias y correcciones en el código, pero no para resolver tareas completas de manera autónoma.
- **Aprendizaje y Recursos:** Los coders podrán utilizar herramientas de IA para acceder a materiales de aprendizaje y documentación relevante.
- **Optimización y Mejora:** Se permite el uso de la IA para optimizar y mejorar el código ya desarrollado por el coder, siempre y cuando se mantenga la integridad del proceso de aprendizaje.

Alcance de la IA

- **Código y Desarrollo:** La IA puede asistir en la escritura, depuración y optimización del código.
- **Investigación:** Los coders podrán utilizar la IA para buscar información y resolver dudas técnicas, puntuales y acotadas.
- **Documentación:** La IA puede ayudar a generar documentación y comentarios en el código.

Límites de la IA

- **Prohibido el Plagio:** No se permite la generación de soluciones completas o la copia de código de fuentes externas sin la debida atribución.

- **Autonomía del Coder:** La IA no debe sustituir la capacidad del coder para resolver problemas; debe actuar como una herramienta de apoyo.
- **Evaluación de Habilidades:** La IA no puede ser utilizada para responder a preguntas de evaluación de manera directa, ya que el objetivo es evaluar las competencias del coder.

Buen Uso de Internet

- **Investigación y Recursos:** Se permite el uso de internet para buscar información, acceder a documentación y utilizar recursos educativos.
- **Comunicación:** Los coders pueden utilizar internet para comunicarse con el equipo de evaluación o pedir asistencia técnica si es necesario.
- **Desarrollo Colaborativo:** El uso de plataformas de desarrollo colaborativo en línea (como GitHub) está permitido para la gestión de proyectos.

Alcance de Internet

- **Acceso a Documentación:** Los coders podrán consultar documentación oficial y recursos educativos en línea.
- **Resolución de Problemas:** Se permite el uso de foros y comunidades técnicas para la resolución de dudas específicas y acotadas siempre que se referencie.
- **Actualizaciones y Herramientas:** Los coders pueden descargar e instalar herramientas y actualizaciones necesarias para el desarrollo del assessment.

Límites de Internet

- **Fuentes Confiables:** Se debe verificar la fiabilidad de las fuentes utilizadas. No se permite el uso de contenido no autorizado o pirata.
- **Prohibido el Fraude:** No se permite buscar o utilizar soluciones completas o respuestas directas a los problemas planteados en el assessment.
- **Seguridad y Privacidad:** Se debe garantizar la seguridad y privacidad de la información personal y de los datos del assessment. No se permite compartir información confidencial.

Monitoreo y Cumplimiento

- **Supervisión:** El uso de IA e internet será monitoreado por el TL para asegurar el cumplimiento de los lineamientos.
- **Consecuencias:** Cualquier violación a estos lineamientos resultará en una evaluación adicional y posibles sanciones, incluyendo la descalificación del assessment.

Ética y Responsabilidad

- **Uso Ético:** Se espera que todos los coders usen la IA e internet de manera ética y responsable.
- **Responsabilidad Individual:** Cada coder es responsable de su propio trabajo y del cumplimiento de estos lineamientos.

Rubricas de evaluación:

- Dirígete a Moodle para consultar las Rubricas de la prueba

Historia de usuario a entregar al Coder (Enunciado):

¡Bienvenido Coder a tu prueba de desempeño de framework (NestJS)!

Las reglas:

- Tienes 8 horas máximo para la culminación de la prueba. (2pm a 10pm) con 3 descansos (4:00 am a 4:20 am, 06:00 apm a 06:20 pm y 08:00 pm a 08:20 pm)
- Puedes usar documentación oficial, ejemplos de código abierto y herramientas de IA como apoyo, evitando el plagio.
- Debes tener la capacidad de explicar tu código.
- Debes realizar la prueba de forma individual.
- Se deben cumplir los criterios de aceptación de la HU.

Las metodologías:

- Trabajarás en una historia de usuario similar a un entorno de trabajo real.
- Demuestra tus conocimientos técnicos y tu capacidad para resolver problemas.

- ¡Buena suerte y manos a la obra!

Historia de usuario:

Empresa: eSports Arena

Contexto: eSports Arena, una empresa especializada en la organización y gestión de torneos de videojuegos a nivel nacional, está buscando mejorar la administración de sus eventos mediante una API que les permita gestionar torneos, jugadores y resultados de manera más eficiente.

Tarea Asignada:
Como desarrollador en la plataforma de torneos de **eSports Arena**, necesito implementar las funcionalidades que permitan gestionar torneos y jugadores. Esta API debe permitirme registrar nuevos jugadores, crear, actualizar y eliminar torneos, añadir jugadores a torneos, y consultar la información de los eventos y sus participantes. También debe ofrecer la capacidad de generar aleatoriamente los emparejamientos de las competencias y registrar los resultados, incluyendo el ganador, el perdedor y los puntajes obtenidos. Además, se requiere la posibilidad de consultar estos resultados por torneo, aplicando filtros por puntaje, ordenando por clasificación y utilizando paginación.

Tiempo Estimado: 8 horas

Criterios de Aceptación:

Persistencia de Datos:

- Implementar TypeORM con PostgreSQL o MySQL para el almacenamiento de los datos.
- Modelar correctamente las tablas para torneos, jugadores y resultados.
- Bonus: Si se implementan scripts de poblamiento (seeds) o migraciones para facilitar el proceso.
- Usar "soft delete" para los registros que se eliminen.

Gestión de Rutas (Routing):

- Crear rutas RESTful para las operaciones CRUD de torneos, jugadores y resultados.
- Asegurarse de que las rutas sean claras y sigan las buenas prácticas de APIs REST.

Manejo de Datos y DTO:

- Validar correctamente los datos usando DTOs para las operaciones de creación y actualización.
- Manejar adecuadamente los parámetros de URL, query params y los datos enviados en el cuerpo de las solicitudes.

Códigos de Respuesta:

- Implementar códigos de respuesta HTTP adecuados para cada operación (201, 200, 400, 404, 500, etc.).

Documentación del Proyecto:

- Incluir un archivo README que explique cómo configurar y ejecutar el proyecto.
- Utilizar Swagger para documentar la API y su uso.

Control de Versiones con GitFlow:

- Utilizar la metodología GitFlow para gestionar las ramas y versiones del proyecto.
- Asegurarse de que los commits sean descriptivos y sigan las buenas prácticas.

Paginación, Filtros y Ordenamiento:

- Implementar paginación en las consultas de torneos, jugadores y resultados.
- Añadir filtros básicos como la consulta por puntaje en los resultados de los jugadores de un torneo. Bonus: Incluir filtros adicionales en otras consultas.
- Incluir opciones de ordenamiento para los resultados de los jugadores en un torneo. Bonus: Implementar ordenamientos adicionales en otras consultas.

Validación de Solicitudes:

- Implementar validación mediante x-api-key en los headers para asegurar que las solicitudes de creación de torneos sean confiables.

Uso de Funcionalidades del Framework:

- Usar "guards" para proteger las rutas que requieran seguridad o autenticación.

Entregables:

Código del Proyecto:

- Subir el proyecto a un repositorio en GitHub(Asegúrate que el repo este publico).
- Cargar a Moodle un archivo TXT con el enlace al repositorio y un .ZIP de los mismo.
- Solo se revisarán commits que cumplan con la fecha límite de la prueba.

Instrucciones:

- Incluir un archivo README con instrucciones claras sobre cómo instalar, configurar y ejecutar la aplicación(No pretendas que el TL tenga que hacer ingeniera inversa para lograr levantar tu aplicativo, asegúrate de incluir apikeys, variables de entorno para levantar, o cualquier recurso para levantar el proyecto).
- Explicar cómo ejecutar migraciones o scripts de poblamiento de datos (seeds) si fueron implementados.

Documentación de la API:

- Documentar todas las rutas y funcionalidades de la API utilizando Swagger para facilitar la interacción con la misma.

Notas Adicionales:

1. Se valorará la claridad y organización del código, así como la correcta implementación de las herramientas y prácticas solicitadas.

2. Se valora que todo el código tanto comentarios, commits e instrucciones sean en ingles.

Actividades Extras (Opcionales):

Estas actividades adicionales no son obligatorias, pero sumarán puntos extra si se implementan de manera efectiva.

Autenticación y Autorización:

- Implementar autenticación mediante JWT (JSON Web Token) para proteger las rutas de creación, actualización y eliminación de torneos, jugadores y resultados.
- Agregar roles de usuario (ej. "admin" y "jugador") y permisos que limiten las acciones de acuerdo con el rol.

Notificaciones en Tiempo Real:

- Implementar WebSockets o cualquier solución de notificación en tiempo real para enviar actualizaciones sobre los resultados de los torneos o los emparejamientos a los jugadores inscritos.

Pruebas Unitarias y de Integración:

- Desarrollar pruebas unitarias para validar las funcionalidades de la API (crear, actualizar, eliminar y consultar torneos, jugadores y resultados).
- Implementar pruebas de integración para asegurar que el sistema funcione correctamente de principio a fin.

Despliegue en la Nube:

- Desplegar la API en una plataforma de servicios en la nube como Heroku, Railway AWS o DigitalOcean.
- Proveer una URL pública para la API que permita interactuar con ella en tiempo real.

Generación de Informes:

- Implementar una funcionalidad que permita generar informes en formato PDF o CSV con los resultados y estadísticas de los torneos.
- Incluir un endpoint que descargue dichos informes.