**HOJA DE RESPUESTA**

|  |  |
| --- | --- |
| 1. **Información de la aplicación** | |
| Plan de Estudio | Desarrollo de Aplicaciones Full Stack Javascript Trainee V2.0 |
| Nombre Ejecutante |  |
| Rut Ejecutante |  |
| Fecha Ejecución |  |

|  |
| --- |
| 1. **Instrucciones de llenado** |
| Marque con una “X” en el casillero que mejor se ajusta a lo observado en la evidencia para cada uno de los aspectos a evaluar enumerados en la rúbrica. En donde, TL: Totalmente Logrado, L: Logrado, ML: Medianamente Logrado, NL: No Logrado. |

|  |  |  |  |
| --- | --- | --- | --- |
| 1. **Resultado Final** | | | |
| Puntaje Obtenido |  | Calificación Obtenida |  |
| Fecha Corrección |  | | |
| Nombre Corrector |  | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. **Rúbrica** | | | | | |
| **#** | **Aspectos a Evaluar** | **TL** | **L** | **ML** | **NL** |
| **Consultas a Base de Datos** | | | | | |
| 1 | Selecciona las columnas requeridas para presentar la información solicitada | Todas las columnas requeridas por el problema son seleccionadas | En su mayoría las columnas son las requeridas, salvo excepciones menores | Sólo selecciona algunas de las columnas requeridas | No realizado o bien la consulta arroja errores |
| 2 | Utiliza JOIN para relacionar la información de distintas tablas | Se realizan los JOIN necesarios con las tablas requeridas con la correcta sintaxis | Se realizan los JOIN necesarios con las tablas, pero existen errores de sintaxis o bien hubo JOINS con tablas innecesarias | Se realizan algunos JOIN necesarios, pero son insuficientes para resolver los problemas pero cumple con la sintaxis | Los JOIN son incorrectos, insuficientes y tienen errores en su sintaxis |
| 3 | Utiliza WHERE para filtrar la información requerida | Se realiza el filtrado de información requerido para resolver el problema en todos los casos | Se realiza el filtrado de información requerido para resolver el problema, pero tiene errores de sintaxis | Se realizan algunos de los filtrados requeridos pero la sintaxis es correcta | No se realizan los filtrados requeridos y hay errores en la sintaxis |
| 4 | Utiliza cláusulas de ordenamiento para presentar la información | Se realiza el ordenamiento de acuerdo con lo solicitado en el problema | Se realiza el ordenamiento de acuerdo con lo solicitado, pero tiene errores de sintaxis | - | No realiza el ordenamiento solicitado |
| 5 | Utiliza cláusulas de agrupación de información para obtener datos agregados | Se realiza la agrupación de la información en todos los casos requeridos en el problema | Se realiza la agrupación en los casos requeridos, pero tiene errores de sintaxis | - | No realiza la agrupación de la información |
| **Uso de estructuras de datos, sentencias, convenciones… (Considerar el sistema Rest para verificar estos puntos)** | | | | | |
| 6 | Utilización general del lenguaje, sintaxis, selección de tipos de datos, sentencias lógicas, expresiones, operaciones, comparaciones | Utiliza siempre y perfectamente la sintaxis del lenguaje, definición de variables, tipos de dato, construcción de expresiones, sentencias lógicas, operaciones y comparaciones | En su mayoría, utiliza siempre y perfectamente la sintaxis del lenguaje, definición de variables, tipos de dato, construcción de expresiones, sentencias lógicas, operaciones y comparaciones, salvo detalles menores. | Utiliza adecuadamente el lenguaje, pero tiene errores ya sea en las expresiones, sentencias lógicas, comparaciones, u operaciones | Existen errores de compilación, no se aprecia el dominio de la sintaxis del lenguaje |
| 7 | Utilización de sentencias repetitivas | Las sentencias repetitivas son utilizadas correctamente y de forma eficiente en todas las partes donde son requeridas | Las sentencias repetitivas son utilizadas correctamente pero no de la forma más eficiente | Las sentencias repetitivas son utilizadas, pero no se define correctamente la condición de término | No utiliza sentencias repetitivas |
| 8 | Convenciones y estilos de programación | Las variables tienen nombre representativo, se nombran adecuadamente objetos, variables y métodos, la indentación es la correcta en todo el código. El código luce ordenado | Se respeta la convención de nombre para objetos, variables y métodos, y en su gran mayoría existe buena indentación y nombres representativos. El código luce medianamente ordenado | El código luce desordenado, pero al menos se respeta la convención de nombres de objetos, variables y métodos | El código luce desordenado, no respeta convenciones ni estilos de programación |
| 9 | Utilización correcta de estructuras de datos | Utiliza estructuras de datos y las manipula de forma correcta en el código utilizando de forma adecuada los métodos provistos por el lenguaje | Utiliza estructuras de datos, pero no utiliza de forma correcta los métodos provistos por el lenguaje |  | No utiliza estructuras de datos |
| **Página web y html** | | | | | |
| 10 | Utilización de tags html, estilos y responsividad | Utiliza etiquetas html y estilos adecuadamente en todo el código, la página es responsiva y se despliega adecuadamente en un dispositivo pequeño | Utiliza etiquetas html y estilos adecuadamente, pero la página web tiene errores de responsividad | La página web no contiene todos los elementos solicitados, tiene errores y no es responsiva | No realiza el encargo |
| 11 | Utilización de Bootstrap | Utiliza bootstrap en el despliegue de elementos de input, botones y tablas, y su apariencia final se aprecia adecuada | Utiliza bootstrap en el despliegue de elementos input, botones y tablas, pero su apariencia final denota errores | Utiliza bootstrap sólo para pocos elementos | No utiliza bootstrap |
| **Lenguaje Node** | | | | | |
| 12 | Inclusión de paquetes y librerías de usuario | Se incluye paquetes Node y librerías de usuarios de manera correcta respetando la sintaxis y convenciones de nombre establecidas | Se incluye paquetes Node y librerías, pero no se respeta sintaxis o convención de nombres |  | No se incluye paquetes o librerías de usuario |
| 13 | Agrupación del código y separación por funcionalidad | El código y configuraciones se estructura y se agrupa por funcionalidad en archivos separados con las cláusulas de inclusión correctas | Algunas componentes del código se estructuran en archivos separados y se incluyen en el código principal |  | No hay separación por funcionalidad en archivos independientes |
| 14 | Utilización de funciones asíncronas | El código utiliza de forma correcta sentencias asíncronas Callback, Promise, AYNC/AWAIT, procesando las respuestas y capturando errores correctamente | El código utiliza de forma correcta sentencias asíncronas Callback, Promise, AYNC/AWAIT, pero no procesa respuestas o no captura errores correctamente |  | No utiliza funciones asíncronas |
| 15 | Lectura de parámetros de entrada | La lectura de parámetros de entrada se hace de forma correcta, se valida la entrada estándar, tipos y rangos según corresponda a la especificación | La lectura de parámetros de entrada se hace de forma correcta, pero no se valida la entrada estándar, tipos o rangos | Se lee algunos de los parámetros de entrada, y otros no. No se valida la entrada estándar | No se lee la entrada estándar |
| 16 | Funcionamiento general del aplicativo | La aplicación realiza la función esperada, aplicando algoritmos y estructuras, permitiendo el ingreso de parámetros, la lectura y escritura de archivos sin errores | La aplicación realiza la función esperada, aplicando algoritmos y estructuras, pero el archivo de salida no tiene el formato o el orden requerido | La aplicación realiza la función esperada de forma parcial, no aplica algoritmos o estructuras | La aplicación presenta errores que no permiten su ejecución |
| **Conexión a Base de Datos** | | | | | |
| 17 | Manejo de conexión a base de datos desde Node | Se ejecuta correctamente la conexión a la base de datos de manera asíncrona y con el respectivo manejo de errores | Se ejecuta correctamente la conexión a la base de datos de manera asíncrona pero no se controla errores | Se ejecuta correctamente la conexión a la base de datos, pero no se utiliza instrucción asíncrona | No se ejecuta conexión a base de datos |
| 18 | Manejo y ejecución de consultas desde Node | Se ejecuta sentencias de consultas de base de datos de manera correcta, pasando parámetros y recuperando el resultado en variables definidas | Se ejecuta sentencias de consultas de base de datos de manera correcta, pero no se pasa correctamente los parámetros de consulta o no se recibe correctamente la data recuperada | Se ejecuta sentencias de consultas de base de datos con errores que no compilan en la base de datos | No ejecuta sentencias de consulta de base de datos |
| **Uso de Express** | | | | | |
| 19 | Creación servicio Rest con Express | Crea un servicio API REST con Express que recibe parámetros y retorna un listado cuya definición está de acuerdo a las buenas prácticas de diseño REST | Crea un servicio API REST que retorna el listado a partir de parámetros, pero su definición no cumple las buenas prácticas de diseño REST | Crea un servicio API REST, pero presenta errores en su ejecución | No crea servicio API REST |
| 20 | Seguridad | Genera los procesos correspondientes para poder aplicar seguridad al servicio implementado, en el caso de web service, login de acceso, en el caso de restservice, uso de JWT para proteger lo Endpoints |  |  | No protege apropiadamente el sistema implementado |