

## ANEXO EXAMEN DE CERTIFICACIÓN

|                 |                                                              |
|-----------------|--------------------------------------------------------------|
| Plan de estudio | Desarrollo de aplicaciones Fullstack JavaScript Trainee V2.0 |
| Anexo           | Sistema de Adopción de Animales                              |

### Prueba de Certificación: Sistema de Adopción de Animales

¡Felicidades por llegar hasta aquí! En esta ocasión, pondrás a prueba todas tus habilidades de desarrollo **fullstack** para construir un sistema de adopción de animales. Este proyecto abarcará desde la configuración de un servidor en **Express** y la conexión con **PostgreSQL** mediante **Sequelize**, hasta la creación de un frontend que consuma los endpoints de tu backend. El objetivo principal es desarrollar una aplicación que permita gestionar animales que se encuentran en adopción, sus especies, sus razas, así como la información de las personas interesadas en adoptarlos. Además, se deberá contemplar la **autenticación de usuarios** con **encriptación de contraseñas**.

---

#### Instrucciones Generales

1. **Duración:** 6 horas.
  2. **Formato de Entrega:**
    - Código fuente de la aplicación en una carpeta comprimida (.zip).
    - Documentación breve (README) con:
      - Estructura del proyecto.
      - Instrucciones para la instalación y ejecución (backend y frontend).
  3. **Restricciones:**
    - No subir los archivos a GitHub ni a ningún otro repositorio público.
  4. **Evaluación:**
    - Se utilizará una rúbrica que considerará aspectos técnicos (configuración de base de datos, endpoints, seguridad, uso de Sequelize, etc.) y diseño (usabilidad, orden, claridad en el código y la interfaz).
-

## Descripción del Caso

La organización “**Pet’s Home**” desea contar con un sistema en línea que gestione todo el proceso de **adopción de animales**. En este sistema, deben poder registrarse usuarios, hacer login y ver los animales disponibles para adopción, filtrándolos por distintas características (especie o raza, por ejemplo).

Habrán dos tipos de usuarios:

- **Administrador:** capaz de ver y gestionar (crear, editar o eliminar) información relacionada con animales y adopciones.
- **Usuario normal:** puede ver los animales disponibles y solicitar la adopción. Además, puede ver el estado de sus solicitudes de adopción.

Se requiere una base de datos en **PostgreSQL** que almacene toda la información. Para la conexión y modelado se utilizará **Sequelize**. La contraseña de cada usuario debe estar encriptada (por ejemplo, con **bcrypt**).

Tu tarea es **desarrollar el servidor** (con Node.js y Express), **configurar la base de datos** y **crear el frontend** (puedes usar cualquier herramienta tanto en versión vanilla como motores de plantillas como Handlebars) que consuma los endpoints para las operaciones de CRUD y visualización.

---

## Objetivos Generales

1. **Desarrollar un servidor con Express** que exponga endpoints REST para la gestión de:
    - Usuarios.
    - Especies y razas.
    - Animales en adopción.
    - Proceso de adopción (solicitudes, estados, etc.).
  2. **Implementar la base de datos en PostgreSQL** usando **Sequelize** para las distintas entidades del sistema.
  3. **Diseñar un frontend** (SPA o aplicación web tradicional) que consuma los endpoints creados y muestre la información de forma amigable e intuitiva.
  4. **Encriptar las contraseñas de los usuarios** (con bcrypt u otra librería equivalente).
-

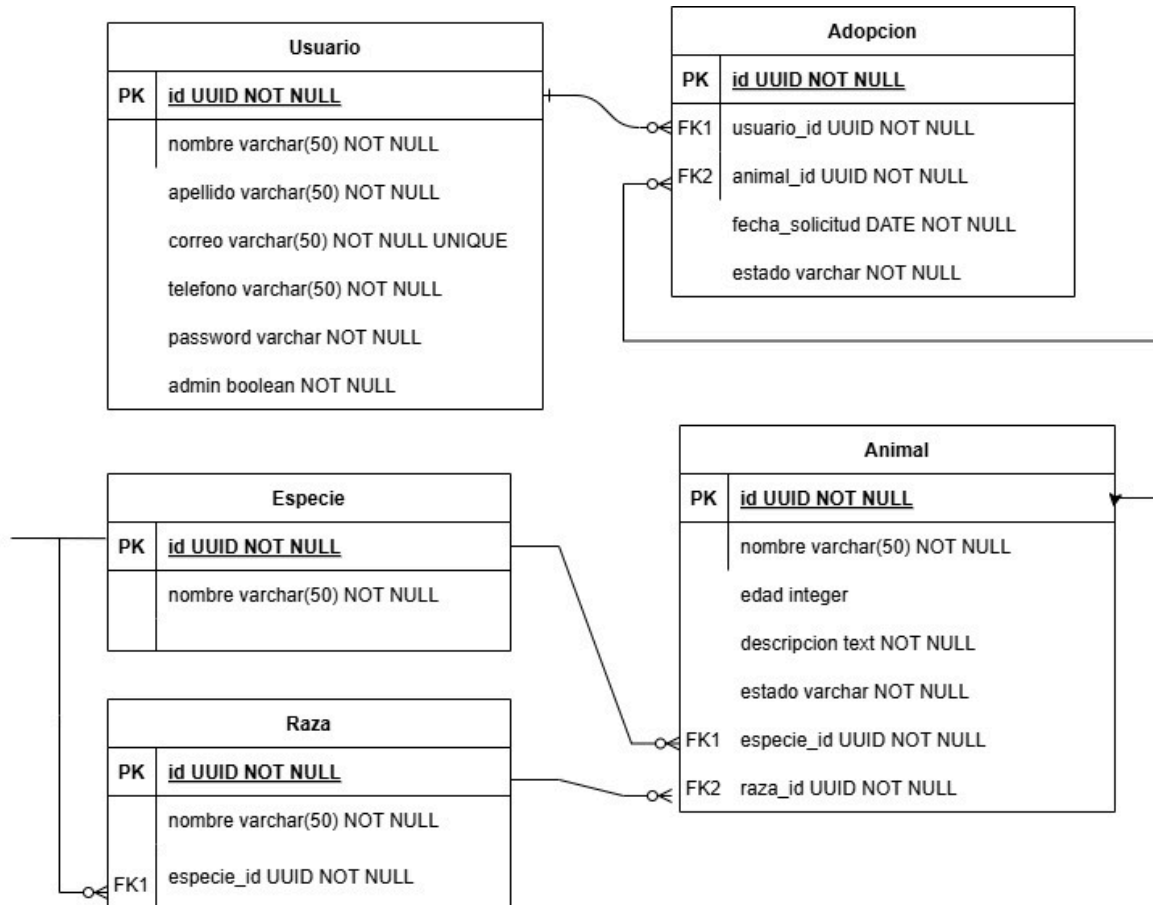
## Requerimientos Funcionales

### 1. Autenticación de Usuarios

- **Registro:**
  - Formulario para registrar un usuario con los campos obligatorios:
    - Nombre
    - Apellido
    - Correo
    - Teléfono
    - password (encriptada en la base de datos)
    - admin (boolean)
- **Inicio de Sesión:**
  - Posibilidad de iniciar sesión con el correo y la contraseña.
  - Validar la contraseña usando la encriptación correspondiente.
- **Control de Acceso:**
  - Los usuarios no autenticados podrán ver la **página de inicio**, el **listado de animales** y, si se desea, la **página de contacto**.
  - Para realizar solicitudes de adopción, deberá haber iniciado sesión.
  - Solo el usuario con rol de **administrador** puede gestionar (crear, editar o eliminar) la información de:
    - Usuarios (cambio de rol, por ejemplo).
    - Animales.
    - Adopciones (ver todas las solicitudes, aprobar, rechazar, etc.).

## 2. Gestión de Animales

### Diagrama Entidad Relación (ERD)



- **Entidades Principales:**

- **Animal:**

- Nombre
    - Edad
    - Descripción (opcional)
    - Relación con **Especie** (FK)
    - Relación con **Raza** (FK)
    - Estado (disponible, adoptado, etc.)

- **Especie:**

- Nombre (ej: "Perro", "Gato", "Conejo" ...)
    - Relación 1 a N con **Raza** (una especie puede tener muchas razas).

- **Raza:**

- Nombre (ej: "Labrador", "Siamés" ...)
    - Relación N a 1 con **Especie** (una raza pertenece a una sola especie).

- **Acciones:**

- Crear nuevos animales (solo admin).
  - Editar y eliminar animales (solo admin).
  - Listar todos los animales con la opción de **filtrar** por:
    - Especie
    - Raza
    - Estado (si se desea)
    - (Opcional) Rango de edad, si quieres ir más allá.

### 3. *Procesos de Adopción*

- **Solicitud de Adopción:**
  - Un usuario autenticado puede solicitar adoptar un **Animal** que esté en estado “disponible”.
  - Esto deberá generar un registro en una tabla de **Adopciones** o **Solicitudes** que relacione:
    - ID del animal
    - ID del usuario solicitante
    - Fecha de la solicitud
    - Estado de la solicitud (pendiente, aprobada, rechazada)
- **Revisión y Gestión de las Solicitudes:**
  - Los **administradores** podrán ver **todas** las solicitudes:
    - Cambiar estado: aprobar o rechazar.
  - El **usuario** puede ver **solo** sus solicitudes de adopción y su estado actual.

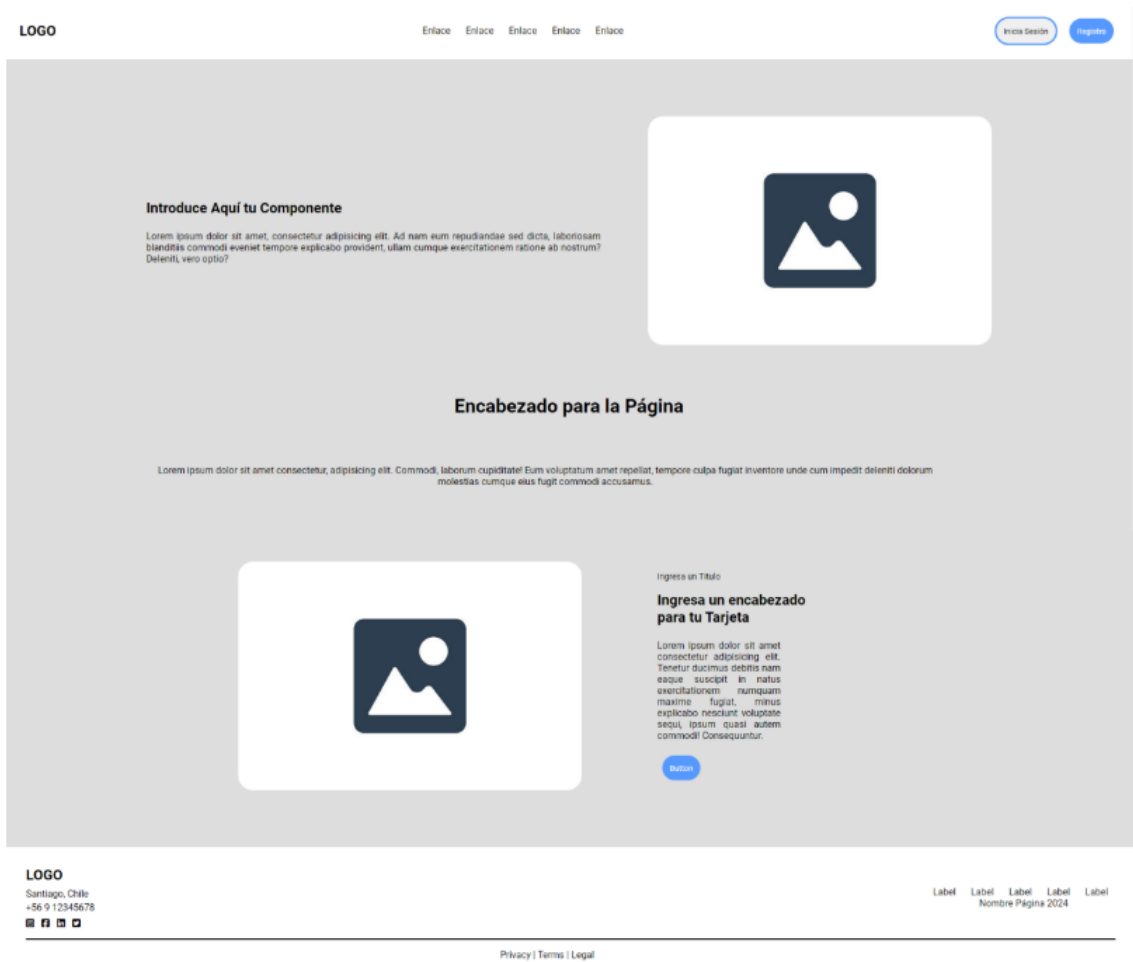
### 4. *Vistas y Navegación*

- **Página de Inicio (/)**
  - Breve presentación del sitio y/o slider/carousel con animales destacados.
  - Botón o enlace para iniciar sesión / registrarse.
- **Página de Registro (/registro)**
  - Formulario para la creación de usuarios.
- **Página de Login (/login)**
  - Formulario para inicio de sesión.
- **Página de Animales (/animales)**
  - Listado de todos los animales disponibles.
  - Barra o panel de filtros por especie, raza y/o rango de edad.
  - Al hacer clic en un animal, mostrar el detalle (información específica).
- **Página de Detalle de Animal (/animales/:id)**
  - Muestra la información del animal seleccionado.
  - Botón para solicitar adopción (si está disponible y el usuario está autenticado).

- **Página de Mis Solicitudes** (/mis-solicitudes)
    - Lista de las solicitudes del usuario actual.
    - Estado de cada solicitud (pendiente, aprobada, rechazada).
  - **Página de Administración** (/admin)
    - **Vista de Usuarios** (/admin/usuarios)
      - Lista de todos los usuarios.
      - Posibilidad de cambiar el rol (admin / usuario).
    - **Vista de Animales** (/admin/animales)
      - Crear, editar y eliminar animales.
    - **Vista de Solicitudes** (/admin/solicitudes)
      - Listar todas las solicitudes.
      - Aprobar o rechazar una solicitud.
  - **Página de Contacto** (opcional) (/contacto)
    - Formulario de contacto (correo, asunto, mensaje).
    - Mensaje de confirmación si se envía con éxito.
-

Vistas:

Estas vistas son meramente orientativas, siéntete libre de diseñarlas como prefieras:

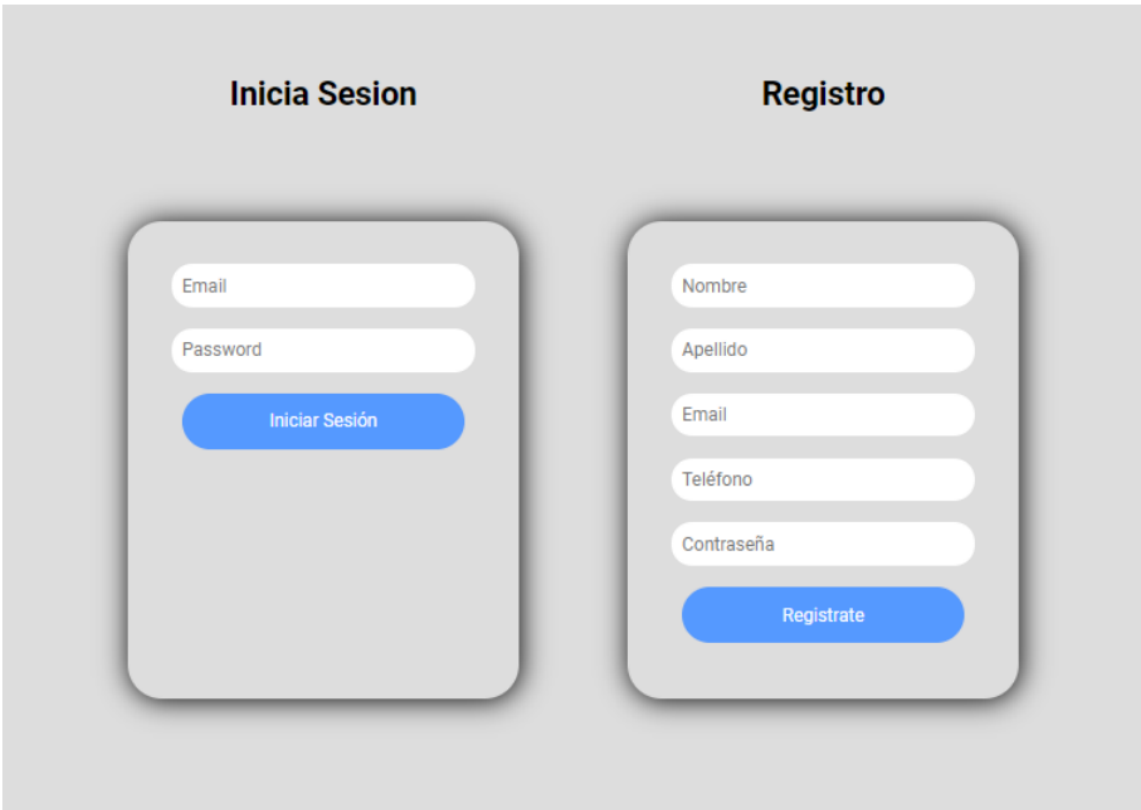


Ruta: / (home)





Ruta contacto: /contact



Ruta Login: /login

Ruta Registro: /registro

| Nombre      | Apellido  | Email                         | Teléfono               |
|-------------|-----------|-------------------------------|------------------------|
| Patricia    | Thompson  | lwarner@gmail.com             | 829.776.0800x8272      |
| Brenda      | Brady     | gillespiedonald@yahoo.com     | (929)656-0015x777      |
| Jennifer    | Baker     | annette26@yahoo.com           | (021)319-5038          |
| Jessica     | Roman     | eric30@galloway.org           | 0536610191             |
| Ronald      | Farley    | garyzavala@yahoo.com          | 001-622-000-2832x923   |
| Paul        | Doyle     | fperry@hotmail.com            | 001-227-856-1735x71091 |
| Miranda     | Murphy    | lovegarrett@gmail.com         | 535-500-0564           |
| Christopher | Thomas    | ewerner@yahoo.com             | (790)176-5085x3636     |
| Kimberly    | Andersen  | larry04@gmail.com             | 737.132.9434x07612     |
| Laura       | Armstrong | mary44@yahoo.com              | (516)345-3230x3742     |
| Dana        | Johnson   | xrivero@peters-hays.com       | (185)192-6258x8218     |
| Andrea      | Frost     | josecott@cooper-white.info    | 903-298-1286           |
| Christine   | Jenkins   | lisa04@hotmail.com            | 560.092.5425x16881     |
| Kerry       | Fowler    | ronald52@gmail.com            | (163)350-7751x860      |
| Theresa     | Houston   | william99@thompson.biz        | (682)291-5302x56739    |
| Laura       | Hall      | shaunphillips@yahoo.com       | (640)649-5318          |
| Christine   | Moore     | kelly13@stuart.com            | 202.028.9464x5098      |
| Kent        | Miller    | hrobles@dunn.com              | (304)303-3430x6642     |
| Michael     | Abbott    | gomezjennifer@smith-lara.info | 537-144-1153x284       |
| Mitchell    | Howard    | nleonard@clark.com            | (607)169-3654          |
| Jane        | Green     | conniemedina@marshall.com     | (224)796-1437x07820    |

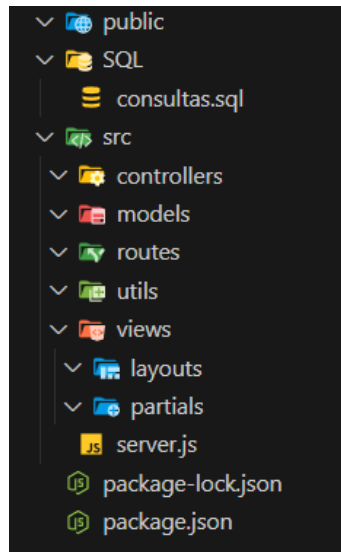
Usuarios Registrados: /admin/users

---

## Requerimientos Técnicos

### 1. Backend en Express:

- Estructura recomendada de carpetas:



(Esta estructura es netamente referencial, puedes emplear otra si lo deseas, pero debes considerar y primar el orden y legibilidad de tu proyecto)

### 2. Consultas a la base de datos:

- Insertar un nuevo Animal
  - Actualizar el estado de un Animal
  - Eliminar un Usuario
  - Contar animales en función de su estado
  - Listar usuarios ordenados por su apellido
  - Obtener las solicitudes de adopción junto con los nombres de los animales y usuarios
  - Listar las razas y sus respectivas especies
  - Realizar un registro completo de adopción
-

## Entregables

1. **Archivo comprimido** (.zip) con la **estructura completa del proyecto** (backend y frontend o estructura monolítica).
2. **Nombre del proyecto:** `adopcion_animales_<nombre_estudiante>`
3. **README** (documentación breve) que incluya:
  - Descripción del proyecto.
  - Instrucciones de instalación y ejecución (cómo correr migraciones de Sequelize, cómo iniciar la API, cómo iniciar el servidor frontend).
  - Resumen de las rutas disponibles en la API (endpoints).
  - Credenciales de un usuario administrador (para pruebas).
4. **Scripts de Base de Datos** para crear y poblar las tablas necesarias, o en su defecto, las migraciones y seeders correspondientes.
5. **El desarrollo de esta prueba es estrictamente individual**, no compartas tu código con otros compañeros.

Para realizar el requerimiento, el Scrum Master menciona lo siguiente:

- Utilizar Bootstrap para enriquecer visualmente la página HTML
- Utilizar NodeJS para poder trabajar con JS fuera del Navegador
- Utilizar NPM para la administración de paquetes en su proyecto
- Utilizar ExpressJS para construir y levantar un servicio rest
- Utilizar las librerías necesarias para conectar a base de datos
- Tener en cuenta que el sitio debe ser responsivo al momento de implementar la vista
- Para la construcción del sistema se debe plantear la construcción de una API Rest con la consideración de la implementación de recursos estáticos que permitan consumir los EndPoints de la API y de esta forma construir el frontend como se plantea en el mockup con los requerimientos funcionales que se definen previamente.
- Aplicar seguridad para el sistema mediante JWT a los endpoints correspondientes (no es necesario implementar una tabla para este proceso, basta con validar credenciales de acceso mediante datos hardcoreados en el código JS)

## Recomendaciones:

### 1. Requerimientos y Entregables:

- Lee atenta y cuidadosamente cada de los requerimientos y entregables antes de comenzar.

### 2. Tecnologías por implementar:

- HTML 5
- CSS 3
- Bootstrap
- PostgreSQL
- Node Js
- NPM
- Express
- PG
- Sequelize
- Json Web Token

### 3. Instalación de Librerías:

- Instala las librerías necesarias con el siguiente comando en la terminal de tu proyecto:  
  
npm install express axios libreria3 libreria4 ....

### 4. Uso de Git:

- Puedes usar Git para controlar las versiones mientras desarrolles, pero ¡No subas el proyecto a GitHub, GitLab u otra plataforma similar!
- a. git init #para iniciar un proyecto con GIT
- b. git add . #agrega cambios al proyectoTalent Digital CASO
- c. git commit -m "mensaje" #agrega Los cambios a GIT con un commit
- d. git checkout id commit #regresa a una versión anterior del proyecto

### 5. Archivo .gitignore:

- Recuerda mantener el archivo .gitignore agregando la carpeta node\_modules, .env y lo que estimes conveniente

## 6. Restaurar y Realizar Backups en PostgreSQL:

- **Backup:** Para crear un respaldo de tu base de datos, puedes usar el comando `pg_dump`. Este comando genera un archivo de respaldo que incluye toda la estructura y los datos.

```
pg_dump -U [usuario] -d [nombre_base_datos] -F c -b -v -f [ruta_backup]
```

- **Restaurar:** Para restaurar una base de datos desde un archivo de respaldo, puedes usar el comando `pg_restore`.
- `pg_restore -U [usuario] -d [nombre_base_datos] -v [ruta_backup]`

### *Notas:*

2. Asegúrate de que el usuario tenga permisos suficientes para realizar backups y restauraciones.
3. El archivo de respaldo se genera con la extensión `.dump` si usas el formato personalizado `-F c`.

**Enlace a la Documentación Oficial:** Para más información, puedes consultar la [documentación oficial de PostgreSQL sobre `pg\_dump` y `pg\_restore`](#).

**¡MUCHO ÉXITO EN ESTE GRAN DESAFÍO!**