Evaluación FrontEnd

¿Qué pretendemos?
Con la presente evaluación buscamos incorporar a nuevos miembros a nuestro equipo para que se desempeñen como FrontEnd Angular evaluando sus conocimientos y el desempeño de cada postulante para resolver un problema orientado a un escenario real.

En el día a día los tiempos de producción son muy importantes, es por eso que deben realizar los ejercicios en el menor tiempo posible, teniendo presente siempre la funcionalidad, sintaxis clara y una interfaz intuitiva y visualmente aceptable.

Tiempo estimado: 1 día (8hs) a partir de la presentación de los enunciados Teórico y Práctico.

Criterios	Descripción
tiempo	en situaciones reales cumplir con el cliente en los márgenes de tiempos asignados es de vital importancia, no se pueden exceder los límites asignados, en caso de presentarse antes de tiempo quedan registradas las marcas temporales en el versionado
funcionalidad	existen requisitos por parte de los clientes con respecto a lo que se espera que el sistema haga, el cumplimiento de estos requerimientos será evaluado en base a la cantidad (están presentes) y calidad (estén cumpliento con el requerimiento solicitado)
sintaxis	el trabajo en equipo y con colegas es importante y gran parte de la interacción y pruebas se realizan sobre la lectura del código, el cual genera un proyecto donde cada persona coloca su aporte, por este motivo el código debe cumplir con un estándar de notación en la definición y uso de variables, funciones, indentación de código y comentarios cuando sea necesario..
interfaz	se debe considerar que la interfaz es la carta de presentación de un proyecto generado por el esfuerzo de varios integrantes y debe ser agradable (disposición de elementos, colores y fuentes de manera armoniosa y siguiendo algún criterio) para el cliente e intuitiva (fácil de entender y de utilizar)

Teoría:

Criterios de evaluación: Ser detallados al responder las consignas. No extenderse más de lo necesario. Utilizar sus propios conocimientos, sin buscar las respuestas en la web.

Forma de presentación: PDF. Adjuntar al proyecto de la parte Práctica.

Responder:

1- ¿Qué es un componente? ¿De qué manera realizaría el intercambio de información entre 2 componentes?

2- ¿Cómo gestionaría el diseño responsivo de un componente?

3- Nombre 3 directivas estructurales. Ejemplifique el uso de 1

4- ¿Cómo protegería el acceso a las rutas de un componente?

5- ¿Qué es un observable? De un ejemplo de definición y uso.

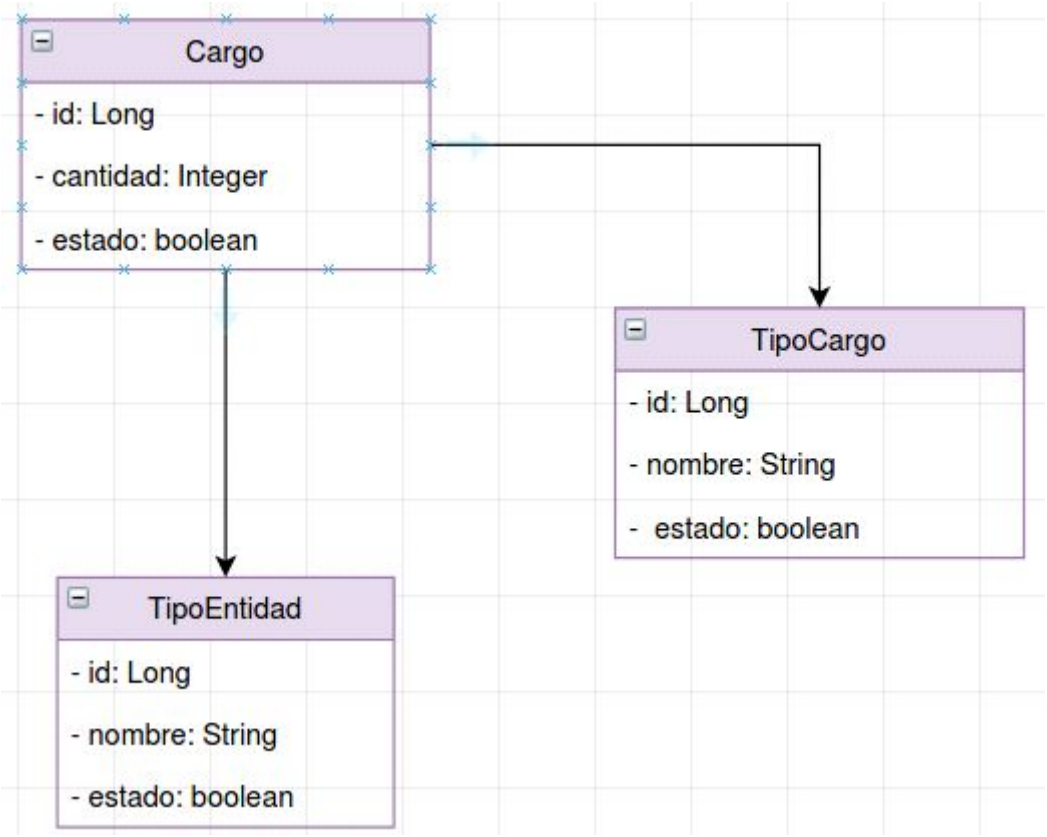
6- Diferencia entre constructor y OnInit.

Ejercicio:

Consideraciones: Interfaz clara y amigable, creatividad en el diseño de los elementos o componentes y buenas prácticas de codificación.
Serán consideradas el uso Bootstrap, Diseño Responsivo, Material, etc.

Forma de presentación: El proyecto debe ser subido a GitHub o GitLab listo para descargar y correr. Cualquier aclaratoria debe estar en el archivo README.md del repositorio.

Deberán realizar las interfaces de los CRUD de las entidades del siguiente diagrama.



Ejemplos
Cargo - Ejemplo: 1 tesorero para una sociedad anónima, 3 secretarías para una SAS, 1 director para la fundación…

- Campos requeridos: Tipo de cargo, tipo de entidad y cantidad.

TipoCargo – Ejemplo: Presidente, Vicepresidente, Secretario, director, tesorero…

- Campos requeridos: Nombre.

TipoEntidad - Ejemplo: Sociedad anónima, Sociedad de acción simplificada, Fundación…

- Campos requeridos: Nombre.

Endpoints

url= https://stage.sys.meyrnr.misiones.gob.ar/fe-1.0/

TipoEntidad

- POST url/tipo_entidad
  {
  "nombre":"SAS"
  }

- PUT url/tipo_entidad/{id}
  {
  "nombre":"SAS"
  }

- GET url/tipo_entidad

- GET url/tipo_entidad/{id}

- DELETE url/tipo_entidad/{id}

TipoCargo

- POST url/tipo_cargo
  {
  "nombre":"Presidente"
  }

- PUT url/tipo_cargo/{id}
  {
  "nombre":"Presidente"
  }

- GET url/tipo_cargo

- GET url/tipo_cargo/{id}

- DELETE url/tipo_cargo/{id}

Cargo

- POST url/cargo
  {
  "cantidad":2,
  “tipoCargo”:{“id”:1},
  “tipoEntidad”:{“id”:2}
  }

- PUT url/cargo/{id}
  {
  "cantidad":2,
  “tipoCargo”:{“id”:1},
  “tipoEntidad”:{“id”:2}
  }

- GET url/cargo

- GET url/cargo/{id}

- DELETE url/cargo/{id}

Endpoint Path	Request Method
/fe-1.0/resources/tipo_cargo/{id}	DELETE
/fe-1.0/resources/tipo_cargo/{id}	GET
/fe-1.0/resources/tipo_cargo/{id}	PUT
/fe-1.0/resources/cargo	GET
/fe-1.0/resources/cargo	POST
/fe-1.0/resources/application.wadl	GET
/fe-1.0/resources/javaee8	GET
/fe-1.0/resources/cargo/{id}	DELETE
/fe-1.0/resources/cargo/{id}	GET
/fe-1.0/resources/cargo/{id}	PUT
/fe-1.0/resources/tipo_entidad	GET
/fe-1.0/resources/tipo_entidad	POST
/fe-1.0/resources/tipo_cargo	GET
/fe-1.0/resources/tipo_cargo	POST
/fe-1.0/resources/tipo_entidad/{id}	DELETE
/fe-1.0/resources/tipo_entidad/{id}	GET
/fe-1.0/resources/tipo_entidad/{id}	PUT