¿Cómo compilar el binario?

Se necesita el runtime de Node para compilar la aplicación frontend con el comando *npm run build*.

Se necesita el runtime de Go para compilar toda la aplicación con el comando *go build .*

El binario construido contiene la aplicación frontend (embebida) en el binario, así como el backend.

¿Cómo ejecutar la aplicación?

La aplicación está diseñada para correr a través de un solo binario que contiene tanto el backend (desarrollado con Go) como el frontend (desarrollado con Svelte). Por la naturaleza de Go, esta aplicación puede ser desplegada en cualquier servidor reciente con Windows o Linux. Lo necesario es compilar el binario y desplegarlo como usualmente se haría con cualquier binario en la plataforma a desplegar. Aunado al binario compilado listo para ser ejecutado, se requieren variables de entorno para configurar el comportamiento de la aplicación, por ejemplo, credenciales de la base de datos para obtener/guardar la información, el puerto en dónde la aplicación estará escuchando peticiones http, entre otras, detalladas más adelante en el documento.

Equipo de cómputo para ejecutar la aplicación

El binario de la aplicación requiere muy pocos recursos para ser ejecutado. Cualquier servidor de los últimos 5 años será capaz ejecutar el binario y soportar la carga de uso, pues Go, maneja muy bien los recursos del sistema para tener un impacto mínimo.

Importación/exportación de datos de MySQL

La base de datos con la cual las pruebas de la aplicación han sido efectuadas es la versión 8.0.27, en los documentos del proyecto viene un archivo sql semilla llamado *dump.sql* con ese como su nombre lo indica es la base para configurar el servidor MySQL con las tablas

necesarias y la información de catálogos para dotar al frontend con la información necesaria para trabajar.
El comando *mysql -u$USER -p$PASSWORD < dump.sql* es el comando que yo utilizo para configurar por primera y única vez el servidor MySQL.

Para extraer la información completa de la base de datos utilizo: *mysqldump -u$USER -p$PASSWORD ops > dump.sql* para obtener toda la información al momento de ejecutar el comando que exista en la base de datos.

Obviamente al ser una base de datos, extraer información se puede hacer con el software habitual utilizado para conectarse a la base de datos y lanzar queries a la misma.

¿Cómo modificar la aplicación?

La aplicación está compuesta por backend y frontend.

Backend:

- Se necesita el runtime de Go instalado. Originalmente se hizo con la versión 1.18.3.
- Se necesita acceso a un servidor con MySQL para interactuar con la base de datos.

Frontend:

- Se necesita el runtime de Node instalado. Originalmente se hizo con la versión 16.15.0
- Se necesita acceso al backend ejecutándose para interactuar con la base de datos y funcionalidad que el backend proporciona.

Cualquier equipo de cómputo de los últimos 5 años será capaz de tener un entorno de desarrollo para el Backend y para el Frontend

Base de datos

Se utilizó MySQL en su versión 8.0.27 para el desarrollo de la aplicación, con la seguridad estándar, usuario y contraseña para acceder a los recursos necesarios.

Schema:

```
mysql> SHOW TABLES;
+----------------+
| Tables_in_ops  |
+----------------+
| catalogs       |
| catalogs_items |
| responses      |
+----------------+
3 rows in set (0.00 sec)

mysql> DESCRIBE catalogs;
+------------+----------+------+-----+-------------------+-----------------------------+
| Field      | Type     | Null | Key | Default           | Extra                       |
+------------+----------+------+-----+-------------------+-----------------------------+
| id         | bigint   | NO   | PRI | NULL              | auto_increment              |
| name       | text     | NO   |     | NULL              |                             |
| created_at | datetime | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED           |
| updated_at | datetime | YES  |     | NULL              | on update CURRENT_TIMESTAMP |
+------------+----------+------+-----+-------------------+-----------------------------+
4 rows in set (0.00 sec)

mysql> DESCRIBE catalogs_items;
+------------+----------+------+-----+-------------------+-----------------------------+
| Field      | Type     | Null | Key | Default           | Extra                       |
+------------+----------+------+-----+-------------------+-----------------------------+
| id         | bigint   | NO   | PRI | NULL              | auto_increment              |
| identifier | bigint   | NO   |     | NULL              |                             |
| label      | text     | NO   |     | NULL              |                             |
| catalog_id | bigint   | NO   | MUL | NULL              |                             |
| created_at | datetime | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED           |
| updated_at | datetime | YES  |     | NULL              | on update CURRENT_TIMESTAMP |
+------------+----------+------+-----+-------------------+-----------------------------+
6 rows in set (0.00 sec)
```

```
mysql> DESCRIBE responses;
+---------------------------------+----------+------+-----+-------------------+-----------------------------+
| Field                           | Type     | Null | Key | Default           | Extra                       |
+---------------------------------+----------+------+-----+-------------------+-----------------------------+
| id                              | bigint   | NO   | PRI | NULL              | auto_increment              |
| identifier                      | bigint   | NO   |     | NULL              |                             |
| gender                          | bigint   | NO   |     | NULL              |                             |
| age                             | bigint   | NO   |     | NULL              |                             |
| age_unit                        | bigint   | NO   |     | NULL              |                             |
| birthdate                       | text     | NO   |     | NULL              |                             |
| deathdate                       | text     | NO   |     | NULL              |                             |
| variable1                       | text     | NO   |     | NULL              |                             |
| variable2                       | text     | NO   |     | NULL              |                             |
| a                               | text     | NO   |     | NULL              |                             |
| time_a                          | bigint   | NO   |     | NULL              |                             |
| time_unit_a                     | bigint   | NO   |     | NULL              |                             |
| b                               | text     | NO   |     | NULL              |                             |
| time_b                          | bigint   | NO   |     | NULL              |                             |
| time_unit_b                     | bigint   | NO   |     | NULL              |                             |
| c                               | text     | NO   |     | NULL              |                             |
| time_c                          | bigint   | NO   |     | NULL              |                             |
| time_unit_c                     | bigint   | NO   |     | NULL              |                             |
| d                               | text     | NO   |     | NULL              |                             |
| time_d                          | bigint   | NO   |     | NULL              |                             |
| time_unit_d                     | bigint   | NO   |     | NULL              |                             |
| part_2                          | text     | NO   |     | NULL              |                             |
| time_part_2                     | bigint   | NO   |     | NULL              |                             |
| time_unit_part_2                | bigint   | NO   |     | NULL              |                             |
| surgery                         | bigint   | NO   |     | NULL              |                             |
| surgery_date                    | text     | NO   |     | NULL              |                             |
| autopsy                         | bigint   | NO   |     | NULL              |                             |
| used_foundings                  | bigint   | NO   |     | NULL              |                             |
| manner_of_death                 | bigint   | NO   |     | NULL              |                             |
| place_of_death                  | bigint   | NO   |     | NULL              |                             |
| multiple_pregnancy              | bigint   | NO   |     | NULL              |                             |
| hours_alive                     | bigint   | NO   |     | NULL              |                             |
| weight_in_grams                 | bigint   | NO   |     | NULL              |                             |
| full_weeks_of_pregnancy         | bigint   | NO   |     | NULL              |                             |
| mother_age_in_years             | bigint   | NO   |     | NULL              |                             |
| woman_pregnancy_condition       | bigint   | NO   |     | NULL              |                             |
| pregnancy_contribution_to_death | bigint   | NO   |     | NULL              |                             |
| created_at                      | datetime | YES  |     | CURRENT_TIMESTAMP | DEFAULT_GENERATED           |
| updated_at                      | datetime | YES  |     | NULL              | on update CURRENT_TIMESTAMP |
+---------------------------------+----------+------+-----+-------------------+-----------------------------+
39 rows in set (0.00 sec)
```

Arquitectura de la aplicación

En el directorio base existen:

- main.go: Archivo con la configuración para inicializar la aplicación.
- handlers.go: Archivo con los handler de las peticiones http que se configuran en main.go
- helpers.go: Archivo con funcionalidad compartida para agilizar el guardado/lectura a la base de datos.
- models: Carpeta con la información necesaria para interactuar entre el backend y la base de datos.

- seed: Carpeta con el archivo *dump.sql* que permite el llenado primero de la base de datos.
- frontend: Carpeta con todo el proyecto frontend.

Directorio *frontend*:

- index.html: Archivo que contiene la base de html sobre la cual se crea la aplicación frontend.
- public: Carpeta con la información que se publicará con la aplicación frontend, por ejemplo contiene el favicon de la aplicación.
- src: Carpeta con los archivos fuente de la aplicación frontend.

Directorio *frontend/src*:

- App.svelte: Archivo con la aplicación Svelte configurada para ser visualizada.
- config.js: Archivo con la configuración de de toda la aplicación, actualmente solo guarda la configuración de la herramienta ICD-11.
- main.js: Archivo que “pega” la aplicación de Svelte con el HTML.
- assets: Carpeta con los archivos multimedia utilizados en la aplicación.
- lib: Carpeta con los componentes que constituyen toda la aplicación.

Generalidades para el buen mantenimiento de la aplicación

Conocer un poco del panorama del entorno de Go y Svelte, dado que son las principales tecnologías utilizadas en el desarrollo de de la aplicación.

Glosario de tecnologías utilizadas en el proyecto

Backend:

- Go: https://go.dev
- gin-gonic: https://github.com/gin-gonic/gin
- cors: github.com/gin-contrib/cors
- mysql-driver: github.com/go-sql-driver/mysql
- godotenv: github.com/joho/godotenv
- sqlc: https://sqlc.dev

Frontend:

- Node: https://nodejs.org/en
- svelte: https://svelte.dev
- svelte-material-ui: https://sveltematerialui.com
- moment: https://momentjs.com