**Demo ApiRestful de Creción de Usuarios**

**Diseñada por Eduardo Coronel**

**Diagrama de la Solución**

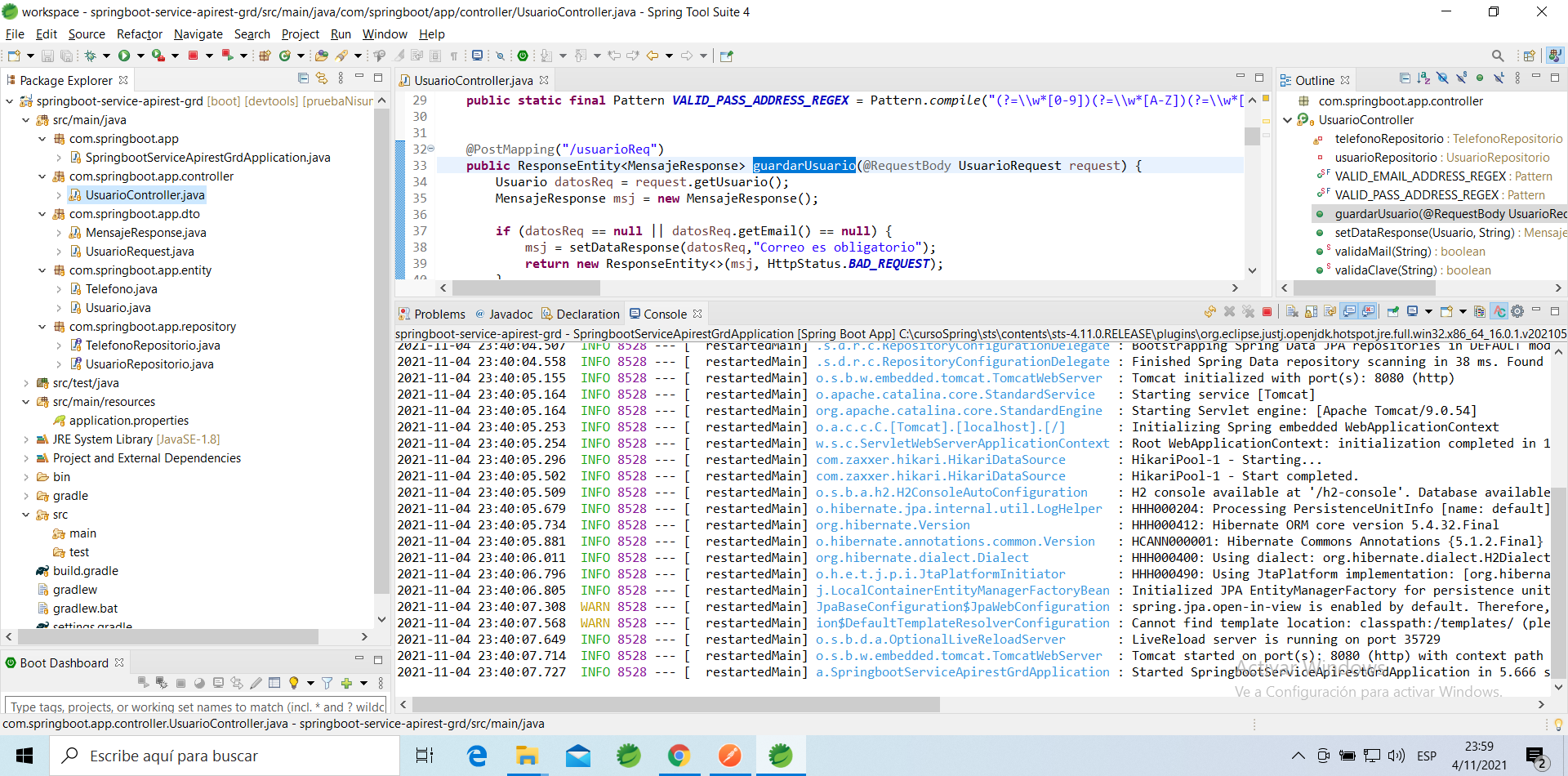
Para desarrollar la solución se creó un proyecto utilizando Java 8, Gradle y Spring.

Los datos se guardan en la BD de memoria H2.

Como IDE se utilizó SpringTool. El proyecto puede descargarse del repositorio: <https://github.com/ecoronelv/pruebaNisum.git>

Una vez que haya sido descargado puede importarse a SpringTool o a cualquier otro IDE.

A continuación, se muestra el diagrama con los paquetes y clases del proyecto.



En el paquete “entity” se encuentran las clases con las entidades Telefono y Usuario, las cuales, gracias a JPA y Hibernate crean las estructuras con los campos solicitados.

En el paquete “dto” tenemos las clases de trabajo con las que interactúa el controlador y que constituyen las estructuras request/response de la Api.

En el paquete “repository” se encuentran las interfaces requeridas para la persistencia y las consultas a la BD.

En el paquete “controller” se encuentra la clase UsuarioController donde se encuentra el método “guardarUsuario” que expone el servicio que será consumido por el cliente. Adicionalmente se encuentran otros métodos de validaciones y lógica del negocio.

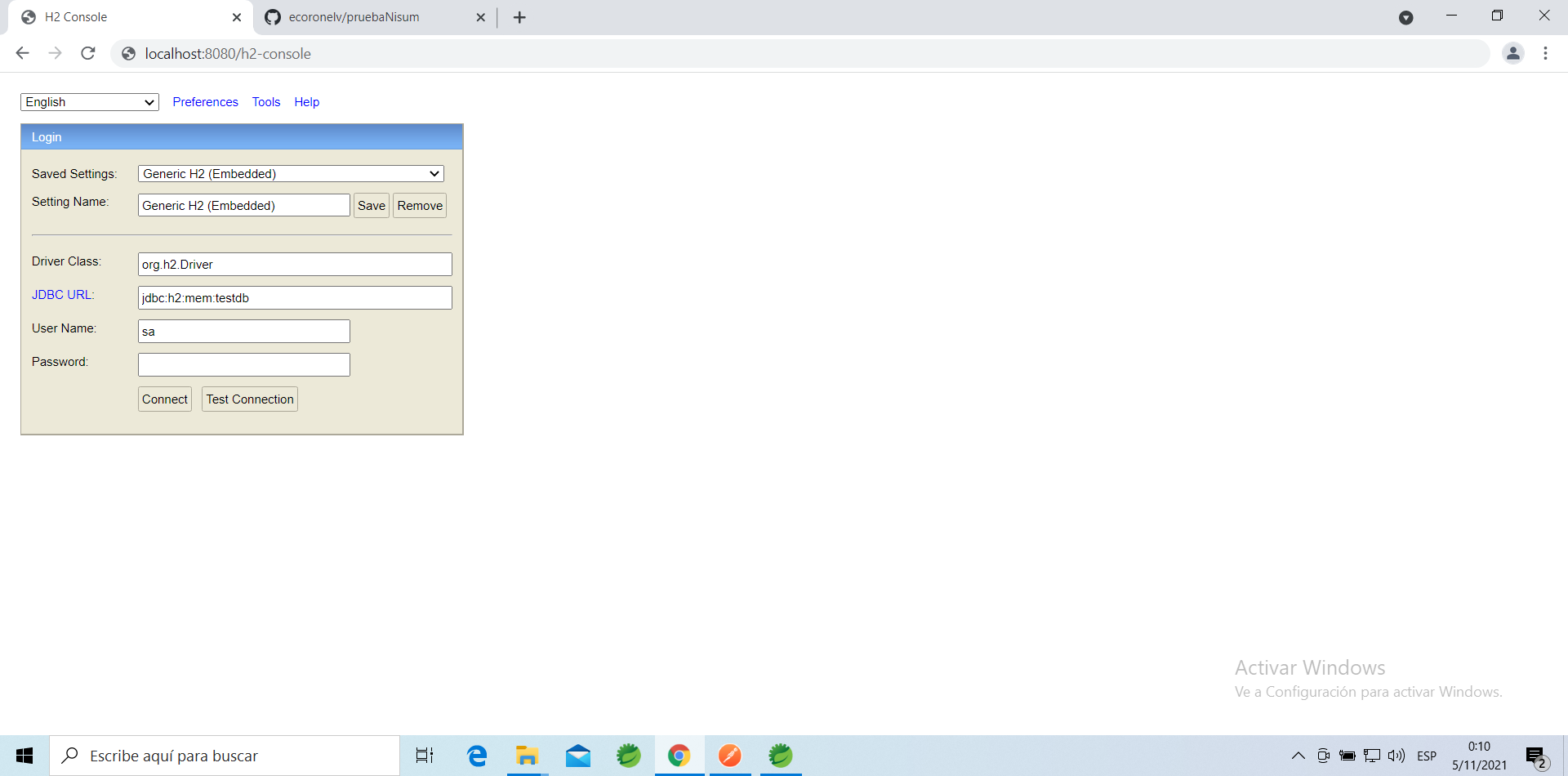
Finalmente, tenemos el archivo application.properties donde se encuentran los parámetros de configuración de la BD H2.

**Ejecución del demo**

El demo se puede ejecutar utilizando cualquier servidor, como por ejemplo Tomcat. Para el ejemplo, utilizaremos el Tomcat que está embebido en SpringTool, dando clic en el botón Run.

El puerto por defecto donde se está levantando el webserver es el 8080.

En el browser (para el ejemplo utilizaremos Google Chrome) se puede consultar la BD H2 llamada testdb con los siguientes datos:



Dar clic en Connect.

Las consultas a las tablas son:

select \* from tbl\_usuario;

select \* from tbl\_telefono;

Las cuales aparecen vacías en este momento.

Para llenarlas con datos utilizaremos postman o cualquier otro aplicativo que me permita enviar una petición JSON HTTP de tipo POST.

{

  "usuario": {

    "nombre": "Eduardo Coronel",

    "email": "edu\_coronel33@hotmail.com",

    "clave": "clavEt2e",

    "telefonos":[

    {

      "number":"0991260978",

      "cityCode":"1",

      "countryCode":"1"

    },

    {

      "number":"0991260979",

      "cityCode":"1",

      "countryCode":"1"

    }

    ]

  }

}

Una vez que se envíe el mensaje deberá aparecer:

{

    "mensaje": "Transaccion exitosa",

    "idUsuario": 1,

    "fechaCreacion": "2021-11-05T05:15:36.082+00:00",

    "fechaModificacion": "2021-11-05T05:15:36.082+00:00",

    "fechaUltimoLogin": "2021-11-05T05:15:36.082+00:00",

    "token": "63dd6106-692d-49b5-be93-808ec11bfa72",

    "active": **true**

}

En caso de que exista algún error deberá aparecer el mensaje correspondiente.

Finalmente, en la BD deberá mostrarse el registro ingresado:

