

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

Preguntas

1. Explica cómo trabajar con repositorios, incluyendo el uso de *branches*, *merge* y *conflicts*. Además, explica los comandos avanzados *Pull Request*, *Fork*, *Rebase*, *Stach*, *Clean* y *CherryPick*.

Git es una herramienta de control de versiones que permite llevar una bitácora de los cambios, corrección de bugs y funcionalidades que surgen durante el desarrollo de Software. El flujo de trabajo más simple para trabajar con esta herramienta consiste en lo siguiente:

- Se tiene un repositorio principal donde se tiene las funcionalidades listas del proyecto, en general la versión “estable.”
- Cuando se quiere trabajar en una nueva funcionalidad del código, arreglar un bug, o en general se solicita modificar el código sin que se vea afectado el repositorio principal, se hace una rama (*branche*), que consiste en una copia del proyecto en un instante determinado de tiempo y cuya idea realizar modificaciones en paralelo que luego serán integradas.
- Cuando se tiene lista la modificación se realiza la unión de la rama con el repositorio principal, mediante el comando *merge*, de esta manera se logra generar un histórico de cambios.
- En muchas situaciones existen conflictos al momento de realizar la unión de ramas, generalmente ocasionado porque se modificaron los mismos archivos en cada una de las ramas. La manera de resolverlo es someterlo a una revisión manual del código, con tal de definir los cambios que serán aceptados o rechazados.

Comandos avanzados

- **Pull Request:** se trata de un evento que ocurre cuando un contribuidor o desarrollador está listo para empezar el proceso de unir sus nuevos cambios de código alguna de las ramas del proyecto. Antes de aceptar los cambios, debe de ser revisado por una persona para determinar si está listo, o no, para ser integrado.
- **Fork (bifurcación):** se trata de una copia simple de un repositorio existente, en el cual el nuevo dueño se desconecta la base de código de los autores anteriores. A partir de ese momento se tienen dos proyectos diferentes y que pueden tener caminos muy separados de desarrollo.
- **Rebase:** es un comando para la integración de cambios entre ramas, a través de reorganizar o cambiar la base de una rama a otra. Cambia la base pareciendo que todos los cambios han sido realizados desde la rama principal que se unió. Permite tener un historial limpio de cambios.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

- **Stash:** almacena temporalmente los cambios hechos en el código en el que se está trabajando para poder trabajar en otra cosa. Guardar los cambios de esta manera resulta práctico si es necesario cambiar rápidamente de contexto y trabajar en otras cosas, pero se está en medio de un cambio en el código sin terminar.
- **Clean:** es un comando para deshacer, opera en archivos sin seguimiento, es decir, aquello que se han creado en el directorio de trabajo del repositorio, pero que no se han añadido con git add.
- **CherryPick:** permite que los *commits* de Git se elijan por referencia y se añadan al actual *HEAD* de trabajo (repositorio en el que se está posicionado el commit). De esta manera se puede elegir el *commit* de una rama y aplicarla a otra. Este comando también puede ser útil para deshacer cambios.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

2. Explica cuáles son las mejores prácticas para un servicio web tipo Rest y explica cada uno de los 4 verbos principales HTTP.

Es una buena práctica que durante el desarrollo de servicios Rest se siga con una serie de pautas sobre la nomenclatura, de tal manera que el desarrollador no tenga que estar adivinando las URL o lo que hacen y surjan de una forma más natural al leer sus nombres.

Pronombres: los *endpoints* (URL) deben de ser pronombres que indiquen de manera clara el recurso que está representado. Deben de ser consistentes y preferentemente en plural.

Verbos: proveen la acción que se llevará a cabo en los recursos basados en pronombres, los 4 verbos principales HTTP son:

- GET: solicita el estado de un recurso específico.
- POST: guarda una nueva representación de un recurso.
- PUT: actualiza el estado de un recurso existente.
- DELETE: eliminar un recurso específico.

La Figura 1 presenta un ejemplo de un sistema con las pautas de nomenclatura recién expresadas. En todos los casos se tiene un *endpoint* común en plural desde el cuál se puede llevar a cabo los métodos POST, para agregar un nuevo recurso y GET para obtener todos los recursos. Para aquellos casos en los que se necesita especificar un usuario, se añade una sub-ruta para acceder a ese recurso cuando se necesita (métodos GET, PUT y DELETE)

El diagrama muestra un conjunto de operaciones REST para un recurso llamado 'persons'. El título principal es 'persons' con el subtítulo 'Operations about users of the system'. Se listan cinco operaciones, cada una con un botón de color que indica el método HTTP: POST (verde), GET (azul), GET (azul), PUT (naranja) y DELETE (rojo). Cada operación incluye la URL y una descripción de su función.

Método	URL	Descripción
POST	/persons	Create a new user in the database. (Note: JSON representation of a new user's data should be contained in the body.)
GET	/persons/	Return a list of all users in the database.
GET	/persons/{username}	Return a user by supplying a unique user name.
PUT	/persons/{username}	Update an existing user in the database. (Note: the JSON representation of the user should be supplied in the body along with the user's user id).
DELETE	/persons/{username}	Delete an existing user whos unique username is supplied.

Figura 1.- Ejemplo conceptual de una nomenclatura adecuada de un servicio Rest.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

Proyectos

La aplicación desarrollada para los dos primeros proyectos se tratará de una aplicación para el registro de pacientes de un hospital, como requisito del cliente se tomará que cada paciente necesita tener al menos los siguientes campos:

- Nombre.
- Número de paciente.
- Personal que lo registró.
- Hospital.
- Temperatura

Para el desarrollo de estos proyectos fue necesario crear el usuario *webpatient* en la base de datos, además de darle los permisos suficientes. Con ese mismo usuario se creó una base de datos con nombre *patient_list* para guardar los registros de la aplicación. La base de datos únicamente consiste de una tabla de *patient* con los campos *full_name*, *patient_number*, *personal*, *hospital* y *temperature*.

La arquitectura del proyecto es la que se muestra en la Figura 2. El sistema completo se puede dividir en dos etapas, el primer bloque donde se va a realizar la exposición de un servicio web de tipo *Rest*, desde el cual se hará la lógica de negocios y la conexión con la base de datos. Mientras que el segundo bloque será el encargado de consumir ese servicio y entregar la vista.

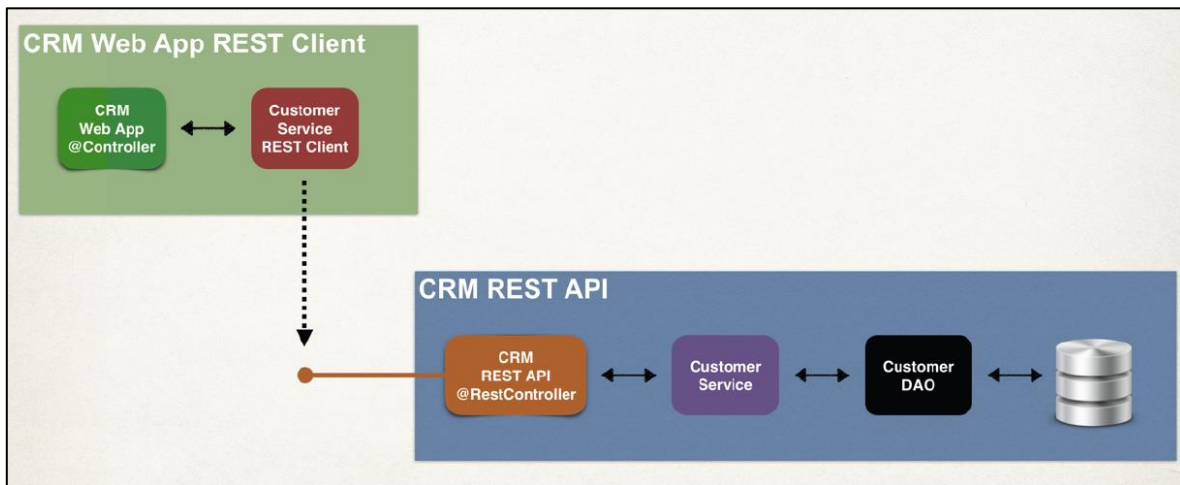


Figura 2.- Diagrama conceptual de bloques del proyecto que expone el servicio y el que lo consume.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

Parte que expone. Escribe una aplicación CRUD que utilice JDBC e Hibernate para devolver un JSON (servicio Rest) o un JSP, utilizando SpringBoot.

Esta parte del proyecto estará hecha con *SpringBoot*, y como una de las principales características de este parte del proyecto la base de datos será manejada utilizando *JDBC* e *Hibernate*. Se retornará un archivo *Json* por ser un servicio *Rest* y el puerto seleccionado fue el 7070. Los *endpoints* fueron probados en *Postman* y la Figura 3 muestran el resultado al jalar el paciente con *id* = 3.

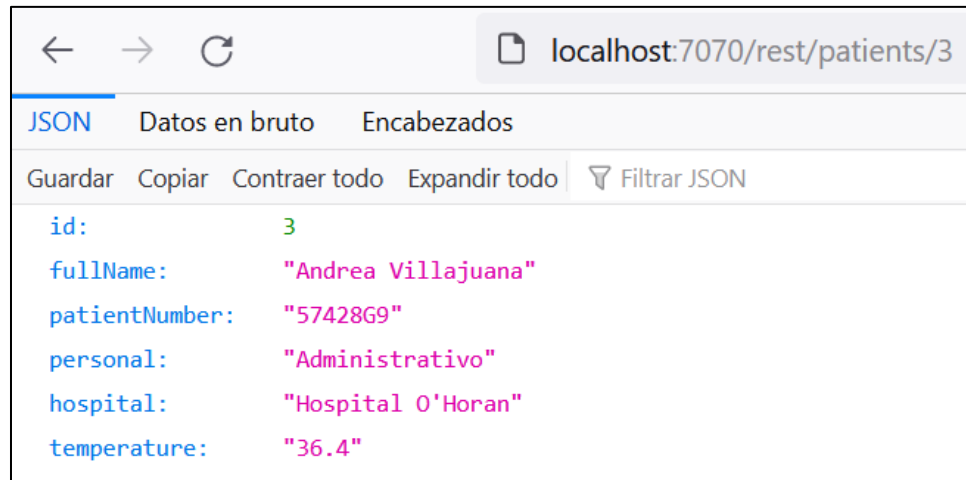


Figura 3.- Ejemplo de un request en el proyecto que expone el servicio Rest.

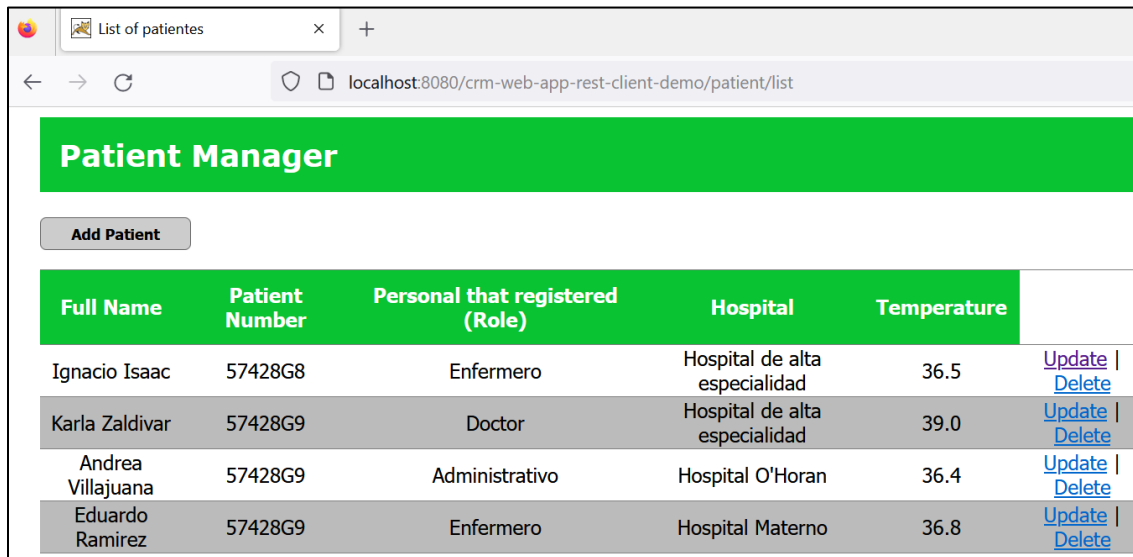
Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4

Parte que consume. Escribe un programa que funcione como cliente de la aplicación Rest creada en el punto anterior.

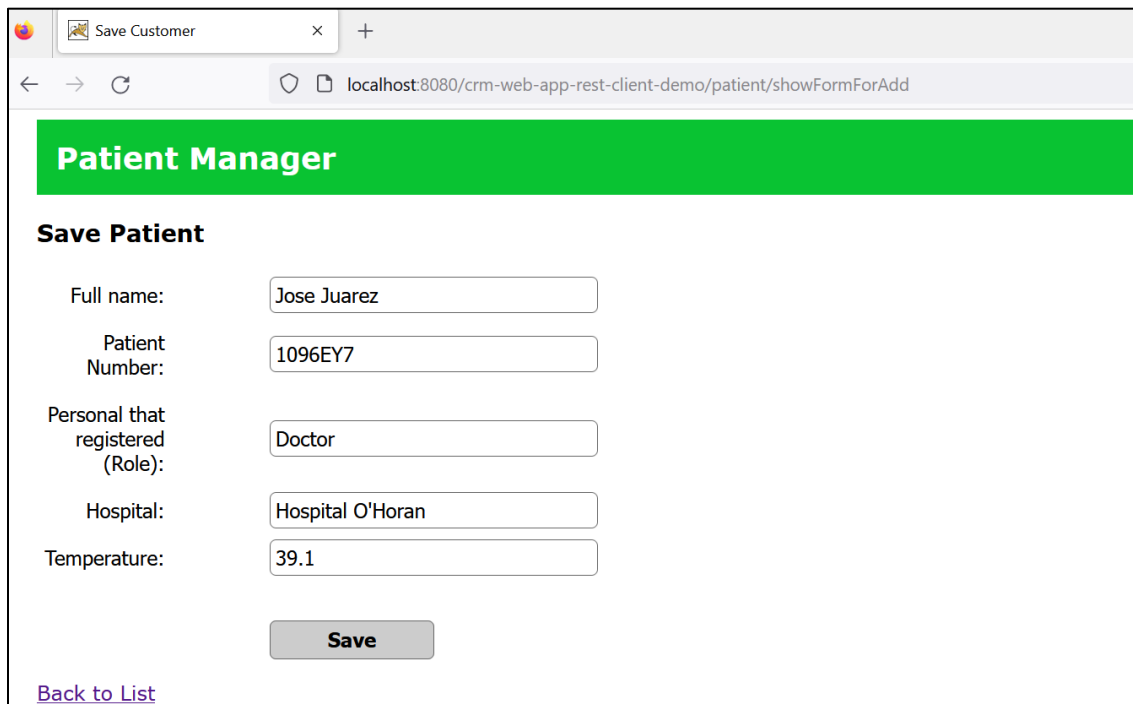
En este apartado se creó un cliente para consumir el servicio *Rest* del punto anterior, de tal manera que los *endpoints* y el modelo coincidiera con lo utilizado anteriormente. Las Figuras 4, 5, 6, 7 y 8 muestran el proyecto en funcionamiento.



The screenshot shows a web browser window with the title 'List of pacientes'. The address bar shows 'localhost:8080/crm-web-app-rest-client-demo/patient/list'. The page has a green header with the text 'Patient Manager'. Below the header is a button labeled 'Add Patient'. The main content is a table with the following columns: Full Name, Patient Number, Personal that registered (Role), Hospital, Temperature, and two action links (Update and Delete). The table contains four rows of patient data.

Full Name	Patient Number	Personal that registered (Role)	Hospital	Temperature	
Ignacio Isaac	57428G8	Enfermero	Hospital de alta especialidad	36.5	Update Delete
Karla Zaldivar	57428G9	Doctor	Hospital de alta especialidad	39.0	Update Delete
Andrea Villajuana	57428G9	Administrativo	Hospital O'Horan	36.4	Update Delete
Eduardo Ramirez	57428G9	Enfermero	Hospital Materno	36.8	Update Delete

Figura 4.- Pantalla principal de los datos en la base de datos.



The screenshot shows a web browser window with the title 'Save Customer'. The address bar shows 'localhost:8080/crm-web-app-rest-client-demo/patient/showFormForAdd'. The page has a green header with the text 'Patient Manager'. Below the header is a section titled 'Save Patient'. It contains five form fields: Full name (Jose Juarez), Patient Number (1096EY7), Personal that registered (Role) (Doctor), Hospital (Hospital O'Horan), and Temperature (39.1). Below the form fields is a 'Save' button. At the bottom left, there is a link labeled 'Back to List'.

Save Patient

Full name:

Patient Number:

Personal that registered (Role):

Hospital:

Temperature:

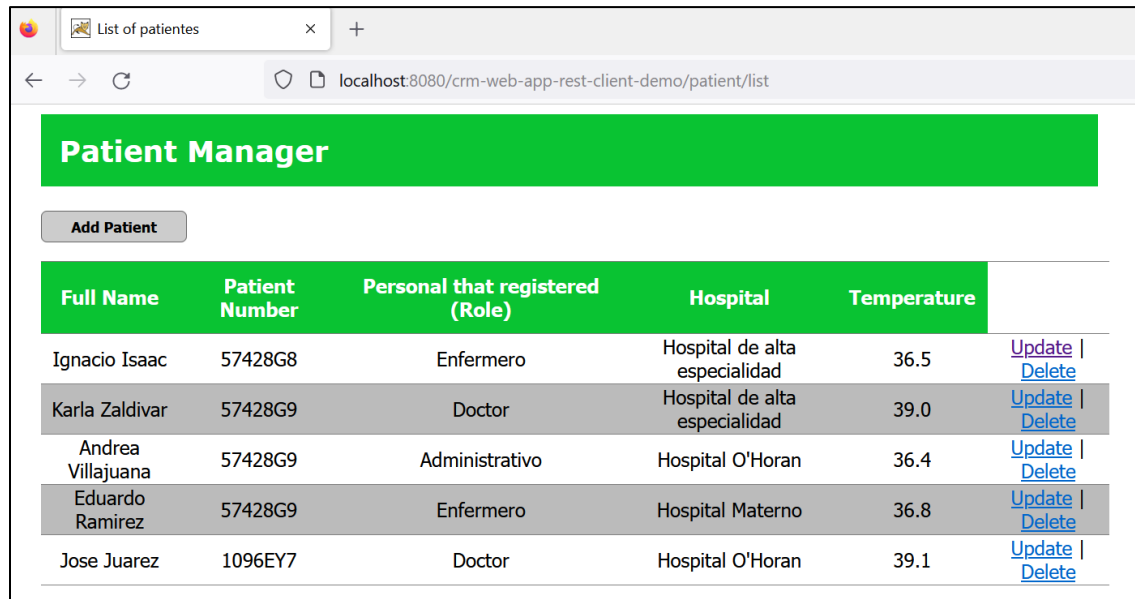
[Back to List](#)

Figura 5.- Vista para agregar un paciente.

Nombre: Ignacio Emmanuel Isaac Medina

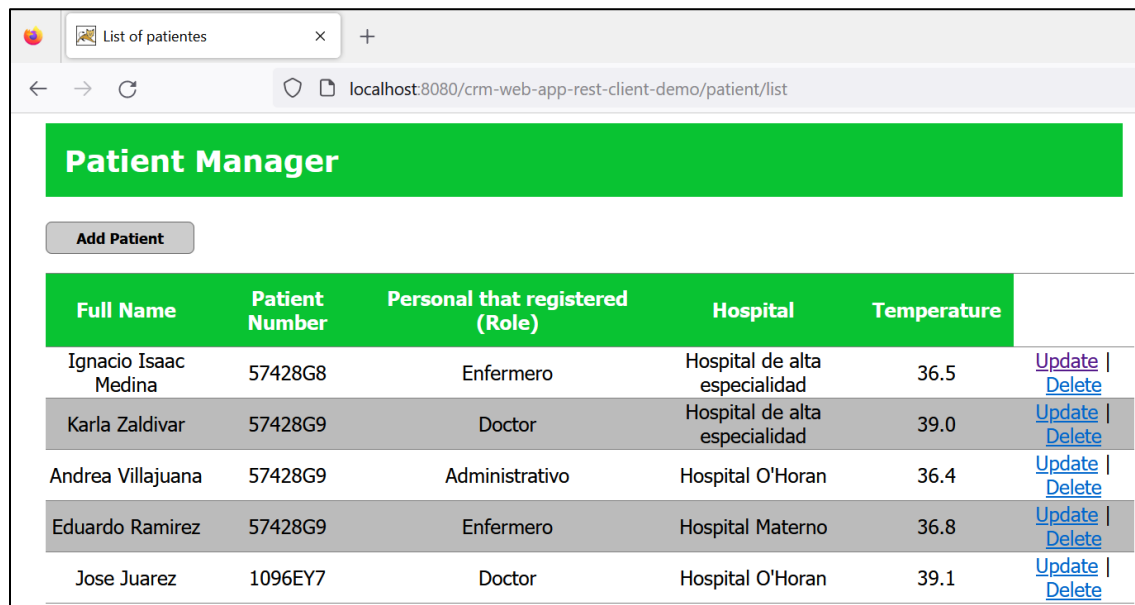
Fecha: 16-Dic-2022

Evaluación escrita Semana 4



Full Name	Patient Number	Personal that registered (Role)	Hospital	Temperature	
Ignacio Isaac	57428G8	Enfermero	Hospital de alta especialidad	36.5	Update Delete
Karla Zaldivar	57428G9	Doctor	Hospital de alta especialidad	39.0	Update Delete
Andrea Villajuana	57428G9	Administrativo	Hospital O'Horan	36.4	Update Delete
Eduardo Ramirez	57428G9	Enfermero	Hospital Materno	36.8	Update Delete
Jose Juarez	1096EY7	Doctor	Hospital O'Horan	39.1	Update Delete

Figura 6.- Tabla general después de agregar un paciente nuevo.



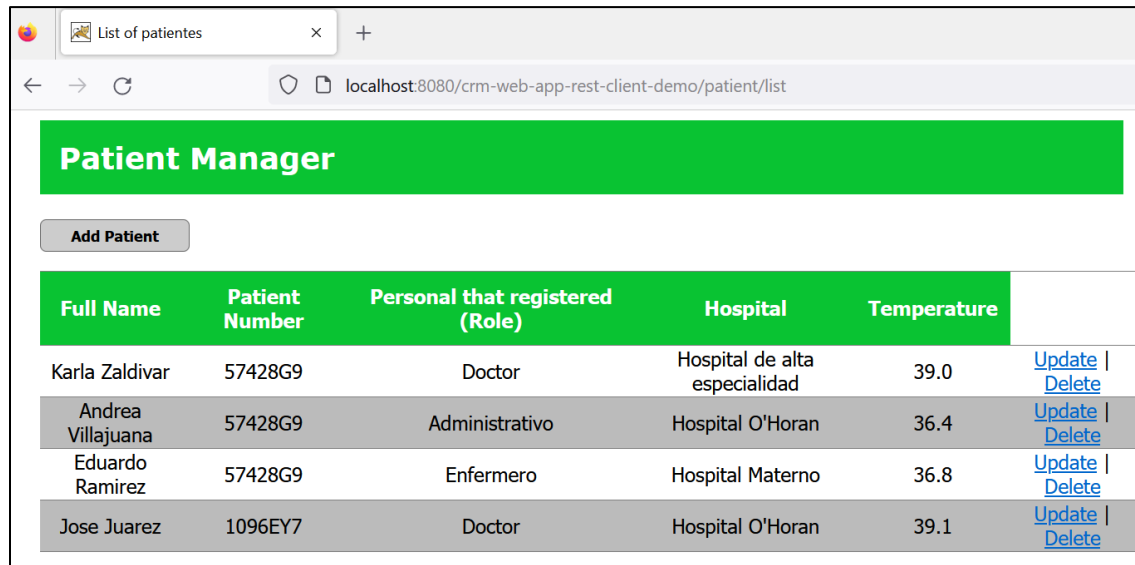
Full Name	Patient Number	Personal that registered (Role)	Hospital	Temperature	
Ignacio Isaac Medina	57428G8	Enfermero	Hospital de alta especialidad	36.5	Update Delete
Karla Zaldivar	57428G9	Doctor	Hospital de alta especialidad	39.0	Update Delete
Andrea Villajuana	57428G9	Administrativo	Hospital O'Horan	36.4	Update Delete
Eduardo Ramirez	57428G9	Enfermero	Hospital Materno	36.8	Update Delete
Jose Juarez	1096EY7	Doctor	Hospital O'Horan	39.1	Update Delete

Figura 7.- Tabla general después de modificar al paciente Ignacio Isaac.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4



Full Name	Patient Number	Personal that registered (Role)	Hospital	Temperature	
Karla Zaldivar	57428G9	Doctor	Hospital de alta especialidad	39.0	Update Delete
Andrea Villajuana	57428G9	Administrativo	Hospital O'Horan	36.4	Update Delete
Eduardo Ramirez	57428G9	Enfermero	Hospital Materno	36.8	Update Delete
Jose Juarez	1096EY7	Doctor	Hospital O'Horan	39.1	Update Delete

Figura 8.- Tabla general después de eliminar al paciente Ignacio Isaac.

Algunas de las posibles mejoras para este apartado sería la modificación del *jsp* y *css* para obtener una interfaz de usuario más amigable.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

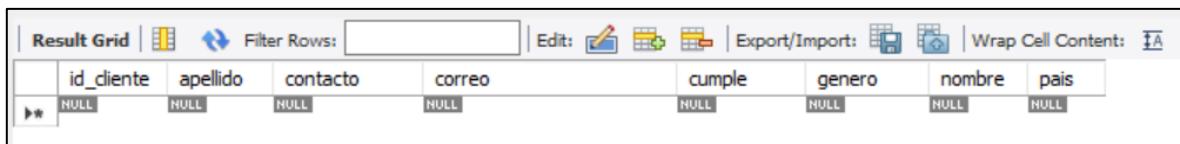
Evaluación escrita Semana 4

Modifica el proyecto de Spring Batch para que los campos estén en español

En este proyecto se modificaron los campos correspondientes al *Entity*, de tal modo que ahora los campos quedaron como:

- Id.
- Nombre.
- Apellido.
- Correo.
- Genero.
- ContactoNo.
- País.
- Cumple (Correspondiente al día de nacimiento).

De igual forma, se cambió el nombre de la base de datos por *clientes_info* y los encabezados del archivo *csv* se renombraron. Además, se cambió el filtrado de China a Russia en la clase *CustomerProcessor* del *Package* con terminación *.config*. Como se utilizó *Lombok*, únicamente se tuvo el cuidado de utilizar los *getters* correspondientes (específicamente el de *getPais*). La Figura 9 presenta la tabla en el *workbench*, de tal manera que no se aprecia ningún dato, mientras que las Figuras 10 y 11 muestran el resultado obtenido después de iniciar el proceso *batch* en el *postman* y la base de datos, respectivamente.



	id_cliente	apellido	contacto	correo	cumple	genero	nombre	pais
1	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figura 9.- Tabla de clientes_info antes de ejecutar el proyecto de Spring Batch.

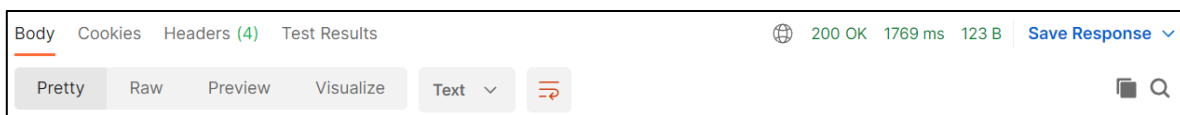


Figura 10.- Respuesta de Postman después de ejecutar el proyecto de Spring Batch.

Nombre: Ignacio Emmanuel Isaac Medina

Fecha: 16-Dic-2022

Evaluación escrita Semana 4








Result Grid		  Filter Rows:	<input type="text"/>		Edit:	  Export/Import:	  Wrap Cell Content:	
	id_cliente	apellido	contacto	correo	cumple	genero	nombre	pais
▶	37	O'Keefe	345-311-4328	jokeefe10@artisteer.com	23/02/2002	Female	Joletta	Russia
	41	Kensall	621-600-1159	akensall14@de.vu	25/04/2014	Female	Allissa	Russia
	62	O' Culligan	891-446-0089	toculligan1p@biglobe.ne.jp	14/11/2019	Male	Tait	Russia
	70	Hedgeman	204-819-8853	ghedgeman1x@topsy.com	30/08/2016	Female	Gilberte	Russia
	71	Hacquoil	649-415-9535	ahacquoil1y@rambler.ru	25/08/1995	Male	Alf	Russia
	78	Dillow	390-205-6658	ndillow25@techcrunch.com	01/12/2015	Male	Nichole	Russia
	85	Loyley	679-369-7397	aloyley2c@free.fr	10/09/2019	Female	Ardelis	Russia
	108	Hortop	628-650-1193	ahortop2z@ifeng.com	01/05/2011	Male	Aksel	Russia
	119	Raeside	930-689-9684	eraeside3a@vimeo.com	30/01/2019	Female	Erinna	Russia
	127	Ducarne	632-872-6702	mducarne3i@trellian.com	28/08/2004	Male	Morris	Russia
	130	Spencers	704-208-3568	mspencers3l@opensource.org	14/04/1997	Female	Marcela	Russia
	134	Burwood	820-936-5324	lburwood3p@sakura.ne.jp	21/10/2010	Female	Leanna	Russia
	141	Bremmell	615-587-7396	hbremmell3w@ibm.com	01/04/2006	Male	Horacio	Russia
	185	Asple	459-883-4118	tasple54@gmpg.org	29/03/2005	Female	Trudi	Russia
	204	Willey	534-493-3593	dwilley5n@ocn.ne.jp	23/11/2013	Female	Dre	Russia
	235	Backshaw	989-874-0324	lbackshaw6i@bigcartel.com	03/05/2001	Female	Luella	Russia
	243	Fleckness	702-727-4959	jfleckness6q@artisteer.com	29/06/2020	Female	Joelly	Russia
	301	Barenskie	602-555-8524	fbarenskie8c@fastcompany.com	10/12/2003	Male	Francklin	Russia
	310	Saunder	490-760-0450	rsaunder8l@usa.gov	02/03/2012	Female	Rea	Russia

Figura 11.- Tabla de clientes_info antes de ejecutar el proyecto de Spring Batch.