SERVICIOS REALIZADOS:

Method: POST

Endpoint: localhost:8080/chequeras/asignar-chequera/1

Descripción: Crea una nueva chequera, con sus cheques y asocia la chequera a una cuenta.

PathParams:

Id tipo **Long**: Id de la cuenta a la que se le asignara la chequera.

Body:

chequera: Json con los atributos para crear una chequera ej.

```
{
    "cuenta_id": "1",
    "fechaAsignacion": "2023-02-02"
}
```

Código:

```
@RequestMapping(value = "/asignar-chequera/{id}", method = RequestMethod.POST)
public String asignarChequera(@RequestBody chequera chequera, @PathVariable("id") Long cuentaId) {
        try {
                Optional<cuenta> c = cuentaRepo.findById(cuentaId);
                if(!c.isPresent()) {
                        throw new Exception(String.format("La cuenta %d no existe", cuentaId));
                }
                chequera.setFechaAsignacion(new Timestamp(new Date().getTime()));
                chequera.setCuenta(c.get());

                List <cheque> listaCheques = new ArrayList<cheque>();

                for (int i = 0; i < 10; i++) {
                        cheque ch = new cheque();
                        ch.setNumeroCheque(i);
                        ch.setChequera(chequera);
                        listaCheques.add(ch);
                }
                chequera.setCheques(listaCheques);
                chequeraRepo.save(chequera);
                return String.format("Se asignó la chequera a la cuenta %d",chequera.getCuenta().getCuenta_id());
        }catch(Exception e) {
                return "Error al asignar cuenta: " + e;
        }
}
```

Method: GET

Endpoint: localhost:8080/clientes

Descripción: Obtiene todos los clientes creados.

Response:


```
[
    {
        "cliente_id": 1,
        "nombre": "SIRIN",
        "direccion": "29021901",
        "telefono": "47380582",
        "cuentas": [
            {
                "cuenta_id": 1,
                "fechaApertura": "2023-02-03T02:43:46.127+00:00",
                "tipoCuenta_id": 2,
                "monto": 500.0,
                "chequeras": [
                    {
                        "chequera_id": 1,
                        "cheques": [
                            {
                                "cheque_id": 1,
                                "numeroCheque": 0,
                                "beneficiario": null,
                                "monto": null,
                                "fecha": null
                            },
                            {
                                "cheque_id": 2,
                                "numeroCheque": 1,
                                "beneficiario": null,
                                "monto": null,
                                "fecha": null
                            },
                            {
                                "cheque_id": 3,
                                "numeroCheque": 2,
                                "beneficiario": null,
                                "monto": null,
                                "fecha": null
                            },
                            {
                                "cheque_id": 4,
                                "numeroCheque": 3,
                                "beneficiario": null,
                                "monto": null,
                                "fecha": null
                            },
                            {
                                "cheque_id": 5,
```

```
                    "numeroCheque": 4,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                },
                {
                    "cheque_id": 6,
                    "numeroCheque": 5,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                },
                {
                    "cheque_id": 7,
                    "numeroCheque": 6,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                },
                {
                    "cheque_id": 8,
                    "numeroCheque": 7,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                },
                {
                    "cheque_id": 9,
                    "numeroCheque": 8,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                },
                {
                    "cheque_id": 10,
                    "numeroCheque": 9,
                    "beneficiario": null,
                    "monto": null,
                    "fecha": null
                }
            ],
            "fechaAsignacion": "2023-02-03T02:45:17.334+00:00"
        }
    ],
    "tipoCuenta": 2
},
{
    "cuenta_id": 2,
```

```
                "fechaApertura": "2023-02-03T02:43:47.081+00:00",
                "tipoCuenta_id": 2,
                "monto": 500.0,
                "chequeras": [],
                "tipoCuenta": 2
            }
        ]
    },
    {
        "cliente_id": 2,
        "nombre": "SIRIN",
        "direccion": "29021901",
        "telefono": "47380582",
        "cuentas": []
    }
]
```


Código:

```
@RequestMapping("/clientes")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class clienteCtrl {

        @Autowired
        private ClienteRepository clienteRepo;

        @Autowired
        private CuentaRepository cuentaRepo;

        @RequestMapping(value = "", method = RequestMethod.GET)
        public List<cliente> obtenerTodos(){
                List<cliente> clientes = clienteRepo.findAll();
                return clientes;
        }
```

Method: POST

Endpoint: localhost:8080/clientes/crear

Descripción: Crea un nuevo cliente

Body:

cliente: Json con los atributos para crear un cliente ej.


```
{
    "nombre": "SIRIN",
    "direccion": "29021901",
    "telefono": "47380582"
}
```

Código:

```
@RequestMapping("/clientes")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class clienteCtrl {

	@Autowired
	private ClienteRepository clienteRepo;

	@Autowired
	private CuentaRepository cuentaRepo;
	@RequestMapping(value = "/crear", method = RequestMethod.POST)
	@ResponseStatus(HttpStatus.CREATED)
	public String crearNuevo(@RequestBody cliente cliente) {
		try {
			cliente c = clienteRepo.save(cliente);
			return String.format("Se creó el cliente %d - %s",c.getCliente_id() ,c.getNombre());
		}catch(Exception e) {
			return "Error al crear el cliente: " + e;
		}

	}
```

Method: DELETE

Endpoint: localhost:8080/clientes/1

Descripción: Elimina un cliente.

PathParams:

Id tipo **Long**: Id del cliente a eliminar.

Código:

```java
@RequestMapping("/clientes")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class clienteCtrl {

	@Autowired
	private ClienteRepository clienteRepo;

	@Autowired
	private CuentaRepository cuentaRepo;

	@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
	public String eliminar(@PathVariable("id") Long clienteId) {
		try {

			clienteRepo.deleteById(clienteId);
			return String.format("Se eliminó el cliente %d",clienteId);
		}catch(Exception e) {
			return "Error al eliminar el cliente: " + e;
		}
```

Method: GET

Endpoint: localhost:8080/cuentas

Descripción: Obtiene todas las cuentas creadas.

Código:

```java
@RequestMapping("/cuentas")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class cuentaCtrl {

	@Autowired
	private CuentaRepository cuentaRepo;

	@Autowired
	private ClienteRepository clienteRepo;

	@RequestMapping(value = "", method = RequestMethod.GET)
	public List<cuenta> obtenerTodos(){
		return cuentaRepo.findAll();
	}
```

Method: POST

Endpoint: localhost:8080/cuentas/asignar-cuenta/1

Descripción: asocia la cuenta a un cliente.

PathParams:

Id tipo **Long**: Id de la cuenta a la que se le asignara la chequera.

Body:

cuenta: Json con los atributos para asignar una cuenta a un cliente ej.

```
{
    "cliente_id": "1",
    "fechaApertura": "2023-02-02",
    "tipoCuenta": "2",
    "monto": 500.00
}
```

Código:

```
@RequestMapping("/cuentas")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class cuentaCtrl {

        @Autowired
        private CuentaRepository cuentaRepo;

        @Autowired
        private ClienteRepository clienteRepo;

        @RequestMapping(value = "/asignar-cuenta/{id}", method = RequestMethod.POST)
        public String asignarCuenta(@RequestBody cuenta cuenta, @PathVariable("id") Long clienteId) {
                try {
                        cliente c = clienteRepo.getById(clienteId);
                        if(c == null) {
                                throw new Exception(String.format("El cliente %d no existe", clienteId));
                        }
                        cuenta.setFechaApertura(new Timestamp(new Date().getTime()));
                        cuenta.setCliente(c);
                        cuenta cuent = cuentaRepo.save(cuenta);
                        return String.format("Se asignó la cuenta al cliente %d",cuent.getCliente().getCliente_id());
                }catch(Exception e) {
                        return "Error al asignar cuenta: " + e;
                }
        }
```

Method: DELETE

Endpoint: localhost:8080/cuentas/1

Descripción: Elimina una cuenta.

PathParams:

Id tipo **Long**: Id de la cuenta que se eliminara.

Código:

```java
@RequestMapping("/cuentas")
@CrossOrigin(origins = "*", allowedHeaders = "*")
public class cuentaCtrl {

	@Autowired
	private CuentaRepository cuentaRepo;

	@Autowired
	private ClienteRepository clienteRepo;
	@RequestMapping(value = "/{id}", method = RequestMethod.DELETE)
	public String eliminar(@PathVariable("id") Long cuentaId) {
		try {

			cuentaRepo.deleteById(cuentaId);
			return String.format("Se eliminó el cliente %d",cuentaId);
		}catch(Exception e) {
			return "Error al eliminar el cliente: " + e;
		}

	}

}
```