Contenido

ENTIDAD BANCO

```
package edu.mtisw.payrollbackend.entities;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import jakarta.persistence.*;

@Entity
@Table(name = "banco")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class BancoEntity {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(unique = true, nullable = false, name = "id")
   private Long id;
}
```

ENTIDAD COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.entities;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;

@Entity
@Table(name = "comprobanteIngresos")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class ComprobanteIngresosEntity implements Serializable {

   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(unique = true, nullable = false, name = "id")
   private Long id;

   @Column(name = "antiguedad_laboral")
   private int antiguedadLaboral; // años en el empleo actual

   @Column(name = "gastos_ultimos_12_meses")
   private String gastosUltimos12Meses; // gastos mensuales últimos 12 meses

   @Column(name = "ingreso_mensual")
   private int ingresoMensual;

   @Column(name = "ingresos_ultimos_12_meses")
   private String ingresosUltimos12Meses;

   @Column(name = "saldo")
   private int saldo; // saldo en cuenta de ahorro o inversiones
}
```

ENTIDAD PRESTAMO

```
package edu.mtisw.payrollbackend.entities;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import jakarta.persistence.*;

@Entity
@Table(name = "prestamo")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class PrestamoEntity {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(unique = true, nullable = false, name = "id")
   private Long id;

   @Column(name = "tipo")// entre "Primera vivienda", "Segunda vivienda", "Propiedades comerciales", "Remodelacion"
   private String tipo;

   @Column(name = "plazo")
   private int plazo;

   @Column(name = "tasa_interes")
```

```
    private double tasaInteres;

    @Column(name = "monto")
    private int monto;

    @Column(name = "estado")// entre "Aprobado", "Rechazado", "En proceso"
    private String estado;
}
```

ENTIDAD USUARIO-COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.entities;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import jakarta.persistence.*;

@Entity
@Table(name = "usuarioComprobanteIngresos")
@Data
@NoArgsConstructor
@AllArgsConstructor

public class UsuarioComprobanteIngresosEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(unique = true, nullable = false, name = "id")
    private Long id;

    @Column(name = "idUsuario")
    private Long idUsuario;

    @Column(name = "idComprobanteIngresos")
    private Long idComprobanteIngresos;
}
```

ENTIDAD USUARIO

```
package edu.mtisw.payrollbackend.entities;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import jakarta.persistence.*;

@Entity
@Table(name = "usuario")
@Data
@NoArgsConstructor
@AllArgsConstructor
public class UsuarioEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(unique = true, nullable = false, name = "id")
    private Long id;

    @Column(name = "rut")
    private String rut;

    @Column(name = "nombre")
    private String nombre;

    @Column(name = "apellido")
    private String apellido;

    @Column(name = "edad")
    private int edad;

    @Column(name = "tipo_empleado")
    private String tipoEmpleado; // "Empleado" o "Independiente"
}
```

ENTIDAD USUARIO-PRESTAMO

```
package edu.mtisw.payrollbackend.entities;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import jakarta.persistence.*;

@Entity
@Table(name = "usuarioPrestamo")
@Data
@NoArgsConstructor
@AllArgsConstructor

public class UsuarioPrestamoEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(unique = true, nullable = false, name = "id")
    private Long id;

    @Column(name = "idUsuario")
    private Long idUsuario;

    @Column(name = "idPrestamo")
    private Long idPrestamo;
}
```

REPOSITORIO BANCO

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.BancoEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

public interface BancoRepository extends JpaRepository<BancoEntity, Integer>{
}
```

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.ComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.entities.EmployeeEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface ComprobanteIngresosRepository extends JpaRepository<ComprobanteIngresosEntity, Integer>{
}
```

REPOSITORIO PRESTAMO

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.PrestamoEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface PrestamoRepository extends JpaRepository<PrestamoEntity, Integer> {
}
```

REPOSITORIO COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.UsuarioComprobanteIngresosEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface UsuarioComprobanteIngresosRepository extends JpaRepository<UsuarioComprobanteIngresosEntity, Integer> {

}
```

REPOSITORIO USUARIO-PRESTAMO

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.UsuarioPrestamoEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface UsuarioPrestamoRepository extends JpaRepository<UsuarioPrestamoEntity, Integer> {
}
```

REPOSITORIO USUARIO

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import java.util.List;

@Repository
public interface UsuarioRepository extends JpaRepository<UsuarioEntity, Integer>{

}
```

REPOSITORIO USUARIO-PRESTAMO

```
package edu.mtisw.payrollbackend.repositories;

import edu.mtisw.payrollbackend.entities.UsuarioPrestamoEntity;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;
```

```
import java.util.List;

@Repository
public interface UsuarioPrestamoRepository extends JpaRepository<UsuarioPrestamoEntity, Integer> {
    List<UsuarioPrestamoEntity> findByIdUsuario(Long idUsuario);
}
```

SERVICIO BANCO

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.BancoEntity;
import edu.mtisw.payrollbackend.repositories.BancoRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service
public class BancoService {
    //-------------------------------------CRUD-----------------------------------------------
    //-------------------------------------PRINCIPALES----------------------------------------
    //simularCredito()(P1)

    //evaluarCredito()(P4)

    //evaluarRelacionCuotaIngreso()(R1)

    //evaluarDeudas()(R2)

    //evaluarAntiguedad()(R3)

    //evaluarRelacionDeudaIngreso()(R4)

    //evaluarMontoMaximoFinanciamiento()(R5)

    //evaluarEdad()(R6)

    //evaluarCapacidadAhorro()(R7)

    //evaluarSaldoMinimo()(R71)

    //evaluarHistorialAhorroConsistente(R72)

    //evaluarDepositosPeriodicos(R73)

    //evaluarRelacionSaldoAntiguedad(R74)

    //evaluarRetiroReciente(R75)

    //calcularCostoTotales()(P6)
}
}
```

SERVICIO COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.ComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.repositories.ComprobanteIngresosRepository;
import edu.mtisw.payrollbackend.repositories.UsuarioRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service
public class ComprobanteIngresosService {
    @Autowired
    ComprobanteIngresosRepository comprobanteIngresosRepository;

    //-------------------------------------CRUD-----------------------------------------------
    // Obtener todos los comprobantes de ingresos
    public ArrayList<ComprobanteIngresosEntity> getComprobanteIngresos(){
        return (ArrayList<ComprobanteIngresosEntity>) comprobanteIngresosRepository.findAll();
    }

    // Obtener un comprobante de ingresos por id
    public ComprobanteIngresosEntity getComprobanteIngresosById(Integer id){
        return comprobanteIngresosRepository.findById(id).get();
    }

    // Guardar un comprobante de ingresos
    public ComprobanteIngresosEntity saveComprobanteIngresos(ComprobanteIngresosEntity comprobanteIngresos){
        return comprobanteIngresosRepository.save(comprobanteIngresos);
    }

    // Actualizar un comprobante de ingresos
    public ComprobanteIngresosEntity updateComprobanteIngresos(ComprobanteIngresosEntity comprobanteIngresos){
        return comprobanteIngresosRepository.save(comprobanteIngresos);
    }

    // Eliminar un comprobante de ingresos
    public boolean deleteComprobanteIngresos(Integer id) throws Exception {
```

```
    try{
      comprobanteIngresosRepository.deleteById(id);
      return true;
    } catch (Exception e) {
      throw new Exception(e.getMessage());
    }
  }

  //-------------------------------------PRINCIPALES-----------------------------------------

}
```

SERVICIO PRESTAMO

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.PrestamoEntity;
import edu.mtisw.payrollbackend.repositories.PrestamoRepository;
import edu.mtisw.payrollbackend.repositories.ExtraHoursRepository;
import edu.mtisw.payrollbackend.repositories.PrestamoRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service
public class PrestamoService {
  @Autowired
  PrestamoRepository prestamoRepository;
  //-------------------------------------CRUD-----------------------------------------------
  // Obtener todos los prestamos
  public ArrayList<PrestamoEntity> getPrestamos(){
    return (ArrayList<PrestamoEntity>) prestamoRepository.findAll();
  }

  // Obtener un prestamo por id
  public PrestamoEntity getPrestamoById(Integer id){
    return prestamoRepository.findById(id).get();
  }

  // Guardar un prestamo
  public PrestamoEntity savePrestamo(PrestamoEntity prestamo){
    return prestamoRepository.save(prestamo);
  }

  // Actualizar un prestamo
  public PrestamoEntity updatePrestamo(PrestamoEntity prestamo){
    return prestamoRepository.save(prestamo);
  }

  // Eliminar un prestamo
  public boolean deletePrestamo(Integer id) throws Exception {
    try{
      prestamoRepository.deleteById(id);
      return true;
    } catch (Exception e) {
      throw new Exception(e.getMessage());
    }
  }

  //-------------------------------------PRINCIPALES-----------------------------------------
}
```

SERVICIO USUARIO-COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.UsuarioComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.repositories.UsuarioComprobanteIngresosRepository;
import edu.mtisw.payrollbackend.repositories.UsuarioRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service
public class UsuarioComprobanteIngresosService {
  @Autowired
  UsuarioComprobanteIngresosRepository usuarioComprobanteIngresosRepository;
  //-------------------------------------CRUD-----------------------------------------------
  // Obtener todos los usuarios comprobante de ingresos
  public ArrayList<UsuarioComprobanteIngresosEntity> getUsuariosComprobanteIngresos(){
    return (ArrayList<UsuarioComprobanteIngresosEntity>) usuarioComprobanteIngresosRepository.findAll();
  }

  // Obtener un usuario comprobante de ingresos por id
  public UsuarioComprobanteIngresosEntity getUsuarioComprobanteIngresosById(Integer id){
    return usuarioComprobanteIngresosRepository.findById(id).get();
  }

  // Guardar un usuario comprobante de ingresos
  public UsuarioComprobanteIngresosEntity saveUsuarioComprobanteIngresos(UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos){
    return usuarioComprobanteIngresosRepository.save(usuarioComprobanteIngresos);
  }
```

```
    // Actualizar un usuario comprobante de ingresos
    public UsuarioComprobanteIngresosEntity updateUsuarioComprobanteIngresos(UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos){
        return usuarioComprobanteIngresosRepository.save(usuarioComprobanteIngresos);
    }

    // Eliminar un usuario comprobante de ingresos
    public boolean deleteUsuarioComprobanteIngresos(Integer id) throws Exception {
        try{
            usuarioComprobanteIngresosRepository.deleteById(id);
            return true;
        } catch (Exception e) {
            throw new Exception(e.getMessage());
        }
    }

    //----------------------------------------PRINCIPALES----------------------------------------
}
```

SERVICIO USUARIO-PRESTAMO

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.UsuarioPrestamoEntity;
import edu.mtisw.payrollbackend.repositories.UsuarioPrestamoRepository;
import edu.mtisw.payrollbackend.repositories.UsuarioRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;

@Service
public class UsuarioPrestamoService {
    @Autowired
    UsuarioPrestamoRepository usuarioPrestamoRepository;
    //----------------------------------------CRUD----------------------------------------
    // Obtener todos los usuarios-prestamos
    public ArrayList<UsuarioPrestamoEntity> getUsuariosPrestamos(){
        return (ArrayList<UsuarioPrestamoEntity>) usuarioPrestamoRepository.findAll();
    }

    // Obtener un usuario-prestamo por id
    public UsuarioPrestamoEntity getUsuarioPrestamoById(Integer id){
        return usuarioPrestamoRepository.findById(id).get();
    }

    // Guardar un usuario-prestamo
    public UsuarioPrestamoEntity saveUsuarioPrestamo(UsuarioPrestamoEntity usuarioPrestamo){
        return usuarioPrestamoRepository.save(usuarioPrestamo);
    }

    // Actualizar un usuario-prestamo
    public UsuarioPrestamoEntity updateUsuarioPrestamo(UsuarioPrestamoEntity usuarioPrestamo){
        return usuarioPrestamoRepository.save(usuarioPrestamo);
    }

    // Eliminar un usuario-prestamo
    public boolean deleteUsuarioPrestamo(Integer id) throws Exception {
        try{
            usuarioPrestamoRepository.deleteById(id);
            return true;
        } catch (Exception e) {
            throw new Exception(e.getMessage());
        }
    }

    //----------------------------------------PRINCIPALES----------------------------------------
}
```

SERVICIO USUARIO

```
package edu.mtisw.payrollbackend.services;

import edu.mtisw.payrollbackend.entities.*;
import edu.mtisw.payrollbackend.repositories.*;
import edu.mtisw.payrollbackend.repositories.UsuarioRepository;
import jakarta.persistence.Id;
import org.hibernate.type.TrueFalseConverter;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

@Service
public class UsuarioService {
    @Autowired
    UsuarioRepository usuarioRepository;
    @Autowired
    ComprobanteIngresosRepository comprobanteIngresosRepository;
    @Autowired
    PrestamoRepository prestamoRepository;
    @Autowired
    UsuarioPrestamoRepository usuarioPrestamoRepository;
    @Autowired
    UsuarioComprobanteIngresosRepository usuarioComprobanteIngresosRepository;
    //----------------------------------------CRUD----------------------------------------
```

```
// Obtener todos los usuarios
public ArrayList<UsuarioEntity> getUsuarios(){
   return (ArrayList<UsuarioEntity>) usuarioRepository.findAll();
}
// Obtener un usuario por id
public UsuarioEntity getUsuarioById(Integer id){
   return usuarioRepository.findById(id).get();
}
// Guardar un usuario
public UsuarioEntity saveUsuario(UsuarioEntity usuario){
   return usuarioRepository.save(usuario);
}
// Actualizar un usuario
public UsuarioEntity updateUsuario(UsuarioEntity usuario){
   return usuarioRepository.save(usuario);
}
// Eliminar un usuario
public boolean deleteUsuario(Integer id) throws Exception {
   try{
      usuarioRepository.deleteById(id);
      return true;
   } catch (Exception e) {
      throw new Exception(e.getMessage());
   }
}
//-------------------------------------PRINCIPALES-----------------------------------------
//simularCredito()(P1)
public Map<String, Object> simularCredito(Long idUsuario, Long idPrestamo) throws Exception {
   // Buscar el usuario por ID
   UsuarioEntity usuario = usuarioRepository.findById(Math.toIntExact(idUsuario))
         .orElseThrow(() -> new Exception("Usuario no encontrado"));

   // Buscar el prestamo por ID
   PrestamoEntity prestamo = prestamoRepository.findById(Math.toIntExact(idPrestamo))
         .orElseThrow(() -> new Exception("Préstamo no encontrado"));

   // Datos del préstamo para la simulación
   double tasaInteresAnual = prestamo.getTasaInteres();
   int plazoEnAnios = prestamo.getPlazo();
   int monto = prestamo.getMonto();

   // Calcular la tasa de interés mensual
   double tasaInteresMensual = (tasaInteresAnual / 12) / 100;

   // Número de pagos (meses)
   int numeroPagos = plazoEnAnios * 12;

   // Cálculo del pago mensual usando la fórmula de amortización
   double pagoMensual = (monto * tasaInteresMensual * Math.pow(1 + tasaInteresMensual, numeroPagos)) /
         (Math.pow(1 + tasaInteresMensual, numeroPagos) - 1);

   // Cálculo del interés total a pagar
   double totalPagos = pagoMensual * numeroPagos;
   double interesesTotales = totalPagos - monto;

   // Crear un mapa con los resultados de la simulación
   Map<String, Object> simulacionResultado = new HashMap<>();
   simulacionResultado.put("monto", monto);
   simulacionResultado.put("plazo", plazoEnAnios);
   simulacionResultado.put("tasaInteres", tasaInteresAnual);
   simulacionResultado.put("pagoMensual", pagoMensual);
   simulacionResultado.put("interesesTotales", interesesTotales);
   simulacionResultado.put("totalPagos", totalPagos);

   return simulacionResultado;
}

// Registrar usuario(implementado en el CRUD)

// Solicitar Credito (P3) (por implementar)
// Consiste en aplicar las funciones saveComprobanteIngresos de la entidad
// y sus id en la tabla intermedia usuario_comprobante_ingresos
// ComprobanteIngresos savePrestamo de la entidad Prestamo
// y sus id en la tabla intermedia usuario_prestamo
public PrestamoEntity solicitarCredito(Long idUsuario, PrestamoEntity prestamo, ComprobanteIngresosEntity comprobanteIngresos) throws Exception {
   // Obtener el usuario por ID
   UsuarioEntity usuario = usuarioRepository.findById(Math.toIntExact(idUsuario)).orElseThrow(() -> new Exception("Usuario no encontrado"));

   // Guardar el comprobante de ingresos
   ComprobanteIngresosEntity comprobanteIngresosGuardado = comprobanteIngresosRepository.save(comprobanteIngresos);

   // Guardar el prestamo
   prestamo.setEstado("En proceso");  // Establecer el estado inicial como "En proceso"
   PrestamoEntity prestamoGuardado = prestamoRepository.save(prestamo);

   // Crear y guardar la relación en UsuarioPrestamo
   UsuarioPrestamoEntity usuarioPrestamo = new UsuarioPrestamoEntity();
   usuarioPrestamo.setIdUsuario(usuario.getId());
   usuarioPrestamo.setIdPrestamo(prestamoGuardado.getId());
   usuarioPrestamoRepository.save(usuarioPrestamo);

   // También podríamos relacionar el comprobante de ingresos si es necesario.
   UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos = new UsuarioComprobanteIngresosEntity();
   usuarioComprobanteIngresos.setIdUsuario(usuario.getId());
   usuarioComprobanteIngresos.setIdComprobanteIngresos(comprobanteIngresosGuardado.getId());
   // Guardar en la tabla intermedia
   usuarioComprobanteIngresosRepository.save(usuarioComprobanteIngresos);

   // Retornar el prestamo guardado
   return prestamoGuardado;
}


// Obtener estado solicitud (P5) (por implementar)
// Método para obtener los estados de los préstamos de un usuario
public List<PrestamoEntity> obtenerEstadoSolicitudes(Long idUsuario) throws Exception {
```

```
        // Verificar si el usuario existe
        UsuarioEntity usuario = usuarioRepository.findById(Math.toIntExact(idUsuario))
                .orElseThrow(() -> new Exception("Usuario no encontrado"));

        // Buscar todas las relaciones usuario-prestamo
        List<UsuarioPrestamoEntity> usuarioPrestamos = usuarioPrestamoRepository.findByIdUsuario(idUsuario);

        // Extraer todos los préstamos asociados
        List<PrestamoEntity> prestamos = new ArrayList<>();
        for (UsuarioPrestamoEntity usuarioPrestamo : usuarioPrestamos) {
            PrestamoEntity prestamo = prestamoRepository.findById(Math.toIntExact(usuarioPrestamo.getIdPrestamo()))
                    .orElseThrow(() -> new Exception("Préstamo no encontrado"));
            prestamos.add(prestamo);
        }

        return prestamos;
    }
}
```

CONTROLADOR BANCO

```
package edu.mtisw.payrollbackend.controllers;

package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.BancoEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.services.BancoService;
import edu.mtisw.payrollbackend.services.UsuarioService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Map;

@RestController
@RequestMapping("/api/v1/bancos")
@CrossOrigin("*")
public class BancoController {
    @Autowired
    UsuarioService usuarioService;
    @Autowired
    BancoService bancoService;
    //-------------------------------------CRUD-------------------------------------------------
    //-------------------------------------PRINCIPALES-------------------------------------------
    //evaluarCredito()(P4)

    //evaluarRelacionCuotaIngreso()(R1)
    @GetMapping("/evaluar-relacion-cuota-ingreso/{idUsuario}/{idPrestamo}")
    public ResponseEntity<Boolean> evaluarRelacionCuotaIngreso(
            @PathVariable Long idUsuario,
            @PathVariable Long idPrestamo) {
        try {
            boolean resultado = bancoService.evaluarRelacionCuotaIngreso(idUsuario, idPrestamo);
            return ResponseEntity.ok(resultado);
        } catch (Exception e) {
            return ResponseEntity.badRequest().body(null);
        }
    }

    //evaluarDeudas()(R2)

    //evaluarAntiguedad()(R3)
    @GetMapping("/evaluar-antiguedad/{idUsuario}")
    public ResponseEntity<Boolean> evaluarAntiguedad(
            @PathVariable Long idUsuario) {
        try {
            boolean resultado = bancoService.evaluarAntiguedad(idUsuario);
            return ResponseEntity.ok(resultado);
        } catch (Exception e) {
            return ResponseEntity.badRequest().body(null);
        }
    }

    //evaluarRelacionDeudaIngreso()(R4)

    //evaluarMontoMaximoFinanciamiento()(R5)

    //evaluarEdad()(R6)

    //evaluarCapacidadAhorro()(R7)

    //evaluarSaldoMinimo()(R71)

    //evaluarHistorialAhorroConsistente(R72)

    //evaluarDepositosPeriodicos(R73)

    //evaluarRelacionSaldoAntiguedad(R74)

    //evaluarRetiroReciente(R75)

    //calcularCostoTotales()(P6)
}
```

CONTROLADOR COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.ComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.services.ComprobanteIngresosService;
import edu.mtisw.payrollbackend.services.UsuarioService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
```

```
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/v1/comprobante-ingresos")
@CrossOrigin("*")
public class ComprobanteIngresosController {
    @Autowired
    ComprobanteIngresosService comprobanteIngresosService;

    //-------------------------------------CRUD-----------------------------------------------
    // Obtener comprobantes de ingresos
    @GetMapping("/")
    public ResponseEntity<List<ComprobanteIngresosEntity>> listComprobanteIngresos(){
        List<ComprobanteIngresosEntity> comprobanteIngresos = comprobanteIngresosService.getComprobanteIngresos();
        return ResponseEntity.ok(comprobanteIngresos);
    }

    // Obtener comprobante de ingreso
    @GetMapping("/{id}")
    public ResponseEntity<ComprobanteIngresosEntity> getComprobanteIngresosById(@PathVariable Integer id){
        ComprobanteIngresosEntity comprobanteIngresos = comprobanteIngresosService.getComprobanteIngresosById(id);
        return ResponseEntity.ok(comprobanteIngresos);
    }

    // Guardar comprobante de ingreso
    @PostMapping("/")
    public ResponseEntity<ComprobanteIngresosEntity> saveComprobanteIngresos(@RequestBody ComprobanteIngresosEntity comprobanteIngresos){
        ComprobanteIngresosEntity comprobanteIngresosNuevo = comprobanteIngresosService.saveComprobanteIngresos(comprobanteIngresos);
        return ResponseEntity.ok(comprobanteIngresosNuevo);
    }

    // Actualizar comprobante de ingreso
    @PutMapping("/")
    public ResponseEntity<ComprobanteIngresosEntity> updateComprobanteIngresos(@RequestBody ComprobanteIngresosEntity comprobanteIngresos){
        ComprobanteIngresosEntity comprobanteIngresosActualizado =  comprobanteIngresosService.updateComprobanteIngresos(comprobanteIngresos);
        return ResponseEntity.ok(comprobanteIngresosActualizado);
    }

    // Eliminar comprobante de ingreso
    @DeleteMapping("/{id}")
    public ResponseEntity<Boolean> deleteComprobanteIngresosById(@PathVariable Integer id) throws Exception {
        var isDeleted = comprobanteIngresosService.deleteComprobanteIngresos(id);
        return ResponseEntity.noContent().build();
    }

    //-------------------------------------PRINCIPALES-----------------------------------------

}
```

CONTROLADOR PRESTAMO

```
package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.PrestamoEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.services.PrestamoService;
import edu.mtisw.payrollbackend.services.UsuarioService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/v1/prestamos")
@CrossOrigin("*")
public class PrestamoController {
    @Autowired
    PrestamoService prestamoService;

    //-------------------------------------CRUD-----------------------------------------------
    // Obtener prestamos
    @GetMapping("/")
    public ResponseEntity<List<PrestamoEntity>> listPrestamos(){
        List<PrestamoEntity> prestamos = prestamoService.getPrestamos();
        return ResponseEntity.ok(prestamos);
    }

    // Obtener prestamo
    @GetMapping("/{id}")
    public ResponseEntity<PrestamoEntity> getPrestamoById(@PathVariable Integer id){
        PrestamoEntity prestamo = prestamoService.getPrestamoById(id);
        return ResponseEntity.ok(prestamo);
    }

    // Guardar prestamo
    @PostMapping("/")
    public ResponseEntity<PrestamoEntity> savePrestamo(@RequestBody PrestamoEntity prestamo){
        PrestamoEntity prestamoNuevo = prestamoService.savePrestamo(prestamo);
        return ResponseEntity.ok(prestamoNuevo);
    }

    // Actualizar prestamo
    @PutMapping("/")
    public ResponseEntity<PrestamoEntity> updatePrestamo(@RequestBody PrestamoEntity prestamo){
        PrestamoEntity prestamoActualizado =  prestamoService.updatePrestamo(prestamo);
        return ResponseEntity.ok(prestamoActualizado);
    }

    // Eliminar prestamo
    @DeleteMapping("/{id}")
    public ResponseEntity<Boolean> deletePrestamoById(@PathVariable Integer id) throws Exception {
        var isDeleted = prestamoService.deletePrestamo(id);
```

```
        return ResponseEntity.noContent().build();
    }

    //-------------------------------------PRINCIPALES-----------------------------------------
}
```

CONTROLADOR USUARIO-COMPROBANTE-INGRESOS

```
package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.UsuarioComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.services.UsuarioComprobanteIngresosService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/v1/usuarios-comprobantes-ingresos")
@CrossOrigin("*")
public class UsuarioComprobanteIngresosController {
    @Autowired
    UsuarioComprobanteIngresosService usuarioComprobanteIngresosService;
    //-------------------------------------CRUD-----------------------------------------------
    // Obtener usuarios comprobantes ingresos
    @GetMapping("/")
    public ResponseEntity<List<UsuarioComprobanteIngresosEntity>> listUsuariosComprobantesIngresos(){
        List<UsuarioComprobanteIngresosEntity> usuariosComprobantesIngresos = usuarioComprobanteIngresosService.getUsuariosComprobanteIngresos();
        return ResponseEntity.ok(usuariosComprobantesIngresos);
    }

    // Obtener un usuario comprobante de ingresos por id
    @GetMapping("/{id}")
    public ResponseEntity<UsuarioComprobanteIngresosEntity> getUsuarioComprobanteIngresosById(@PathVariable Integer id){
        UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos = usuarioComprobanteIngresosService.getUsuarioComprobanteIngresosById(id);
        return ResponseEntity.ok(usuarioComprobanteIngresos);
    }

    // Guardar un usuario comprobante de ingresos
    @PostMapping("/")
    public ResponseEntity<UsuarioComprobanteIngresosEntity> saveUsuarioComprobanteIngresos(@RequestBody UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos){
        UsuarioComprobanteIngresosEntity usuarioComprobanteIngresosSaved = usuarioComprobanteIngresosService.saveUsuarioComprobanteIngresos(usuarioComprobanteIngresos);
        return ResponseEntity.ok(usuarioComprobanteIngresosSaved);
    }

    // Actualizar un usuario comprobante de ingresos
    @PutMapping("/")
    public ResponseEntity<UsuarioComprobanteIngresosEntity> updateUsuarioComprobanteIngresos(@RequestBody UsuarioComprobanteIngresosEntity usuarioComprobanteIngresos){
        UsuarioComprobanteIngresosEntity usuarioComprobanteIngresosUpdated = usuarioComprobanteIngresosService.updateUsuarioComprobanteIngresos(usuarioComprobanteIngresos);
        return ResponseEntity.ok(usuarioComprobanteIngresosUpdated);
    }

    // Eliminar un usuario comprobante de ingresos
    @DeleteMapping("/{id}")
    public ResponseEntity<Boolean> deleteUsuarioComprobanteIngresos(@PathVariable Integer id){
        try {
            boolean isDeleted = usuarioComprobanteIngresosService.deleteUsuarioComprobanteIngresos(id);
            return ResponseEntity.ok(isDeleted);
        } catch (Exception e) {
            return ResponseEntity.badRequest().build();
        }
    }

    //-------------------------------------PRINCIPALES-----------------------------------------
}
```

CONTROLADOR USUARIOS

```
package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.ComprobanteIngresosEntity;
import edu.mtisw.payrollbackend.entities.PrestamoEntity;
import edu.mtisw.payrollbackend.entities.UsuarioEntity;
import edu.mtisw.payrollbackend.services.UsuarioService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

@RestController
@RequestMapping("/api/v1/usuarios")
@CrossOrigin("*")
public class UsuarioController {
    @Autowired
    UsuarioService usuarioService;
    //-------------------------------------CRUD-----------------------------------------------
    // Obtener usuarios
    @GetMapping("/")
    public ResponseEntity<List<UsuarioEntity>> listUsuarios(){
        List<UsuarioEntity> usuarios = usuarioService.getUsuarios();
        return ResponseEntity.ok(usuarios);
    }

    // Obtener usuario
    @GetMapping("/{id}")
    public ResponseEntity<UsuarioEntity> getUsuarioById(@PathVariable Integer id){
        UsuarioEntity usuario = usuarioService.getUsuarioById(id);
        return ResponseEntity.ok(usuario);
    }
```

```
// Guardar usuario
@PostMapping("/")
public ResponseEntity<UsuarioEntity> saveUsuario(@RequestBody UsuarioEntity usuario){
   UsuarioEntity usuarioNuevo = usuarioService.saveUsuario(usuario);
   return ResponseEntity.ok(usuarioNuevo);
}

// Actualizar usuario
@PutMapping("/")
public ResponseEntity<UsuarioEntity> updateUsuario(@RequestBody UsuarioEntity usuario){
   UsuarioEntity usuarioActualizado =  usuarioService.updateUsuario(usuario);
   return ResponseEntity.ok(usuarioActualizado);
}

// Eliminar usuario
@DeleteMapping("/{id}")
public ResponseEntity<Boolean> deleteUsuarioById(@PathVariable Integer id) throws Exception {
   var isDeleted = usuarioService.deleteUsuario(id);
   return ResponseEntity.noContent().build();
}

//-------------------------------------PRINCIPALES-------------------------------------
//simularCredito()(P1)
@GetMapping("/{id}/simular-credito/{idPrestamo}")
public ResponseEntity<Map<String, Object>> simularCredito(
      @PathVariable Long id,
      @PathVariable Long idPrestamo) {
   try {
      // Llamar al servicio para simular el crédito
      Map<String, Object> resultadoSimulacion = usuarioService.simularCredito(id, idPrestamo);
      return ResponseEntity.ok(resultadoSimulacion);
   } catch (Exception e) {
      return ResponseEntity.badRequest().body(null);
   }
}

// Registrar usuario(implementado en el CRUD)

// Solicitar Credito (P3) (por implementar)
// Registrar usuario (P2) (implementado en el CRUD)
@PostMapping("/{id}/solicitar-credito")
public ResponseEntity<PrestamoEntity> solicitarCredito(
      @PathVariable Long id,
      @RequestBody Map<String, Object> requestBody) {

   try {
      // Extracción segura de datos
      String tipo = (String) requestBody.get("tipo");
      int plazo = ((Number) requestBody.get("plazo")).intValue();
      double tasaInteres = ((Number) requestBody.get("tasaInteres")).doubleValue();
      int monto = ((Number) requestBody.get("monto")).intValue();
      int antiguedadLaboral = ((Number) requestBody.get("antiguedadLaboral")).intValue();
      int ingresoMensual = ((Number) requestBody.get("ingresoMensual")).intValue();
      int saldo = ((Number) requestBody.get("saldo")).intValue();

      // Manejo de listas
      List<Number> gastosNumbers = (List<Number>) requestBody.get("gastosUltimos24Meses");
      List<Integer> gastosUltimos24Meses = gastosNumbers.stream()
            .map(Number::intValue)
            .collect(Collectors.toList());

      List<Number> ingresosNumbers = (List<Number>) requestBody.get("ingresosUltimos24Meses");
      List<Integer> ingresosUltimos24Meses = ingresosNumbers.stream()
            .map(Number::intValue)
            .collect(Collectors.toList());

      // Crear los objetos PrestamoEntity y ComprobanteIngresosEntity
      PrestamoEntity prestamo = new PrestamoEntity();
      prestamo.setTipo(tipo);
      prestamo.setPlazo(plazo);
      prestamo.setTasaInteres(tasaInteres);
      prestamo.setMonto(monto);
      prestamo.setEstado("En proceso");

      ComprobanteIngresosEntity comprobanteIngresos = new ComprobanteIngresosEntity();
      comprobanteIngresos.setAntiguedadLaboral(antiguedadLaboral);
      comprobanteIngresos.setIngresoMensual(ingresoMensual);
      comprobanteIngresos.setSaldo(saldo);

      // Convertir listas a cadenas
      String gastosString = gastosUltimos24Meses.stream()
            .map(String::valueOf)
            .collect(Collectors.joining(","));
      comprobanteIngresos.setGastosUltimos24Meses(gastosString);

      String ingresosString = ingresosUltimos24Meses.stream()
            .map(String::valueOf)
            .collect(Collectors.joining(","));
      comprobanteIngresos.setIngresosUltimos24Meses(ingresosString);

      // Llamar al servicio para procesar la solicitud de crédito
      PrestamoEntity prestamoSolicitado = usuarioService.solicitarCredito(id, prestamo, comprobanteIngresos);

      return ResponseEntity.ok(prestamoSolicitado);
      //analizar si se retorna el prestamo o todos los datos ingresados
   } catch (Exception e) {
      e.printStackTrace();
      return ResponseEntity.badRequest().body(null);
   }
}

// Obtener estado solicitud (P5) (por implementar)
@GetMapping("/{id}/estado-solicitudes")
public ResponseEntity<List<PrestamoEntity>> obtenerEstadoSolicitudes(@PathVariable Long id) {
   try {
      List<PrestamoEntity> estadosPrestamos = usuarioService.obtenerEstadoSolicitudes(id);
```

```
            return ResponseEntity.ok(estadosPrestamos);
        } catch (Exception e) {
            return ResponseEntity.badRequest().body(null);
        }
    }
}
```

CONTROLADOR USUARIO-PRESTAMOS

```
package edu.mtisw.payrollbackend.controllers;

import edu.mtisw.payrollbackend.entities.UsuarioPrestamoEntity;
import edu.mtisw.payrollbackend.services.UsuarioPrestamoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.List;

@RestController
@RequestMapping("/api/v1/usuarios-prestamos")
@CrossOrigin("*")
public class UsuarioPrestamoController {
    @Autowired
    UsuarioPrestamoService usuarioPrestamoService;
    //-------------------------------------CRUD-----------------------------------------------
    // Obtener usuarios prestamos
    @GetMapping("/")
    public ResponseEntity<List<UsuarioPrestamoEntity>> listUsuarios(){
        List<UsuarioPrestamoEntity> usuarios = usuarioPrestamoService.getUsuariosPrestamos();
        return ResponseEntity.ok(usuarios);
    }

    // Obtener usuario prestamo
    @GetMapping("/{id}")
    public ResponseEntity<UsuarioPrestamoEntity> getUsuarioById(@PathVariable Integer id){
        UsuarioPrestamoEntity usuario = usuarioPrestamoService.getUsuarioPrestamoById(id);
        return ResponseEntity.ok(usuario);
    }

    // Guardar usuario prestamo
    @PostMapping("/")
    public ResponseEntity<UsuarioPrestamoEntity> saveUsuario(@RequestBody UsuarioPrestamoEntity usuario){
        UsuarioPrestamoEntity usuarioNuevo = usuarioPrestamoService.saveUsuarioPrestamo(usuario);
        return ResponseEntity.ok(usuarioNuevo);
    }

    // Actualizar usuario prestamo
    @PutMapping("/")
    public ResponseEntity<UsuarioPrestamoEntity> updateUsuario(@RequestBody UsuarioPrestamoEntity usuario){
        UsuarioPrestamoEntity usuarioActualizado =  usuarioPrestamoService.updateUsuarioPrestamo(usuario);
        return ResponseEntity.ok(usuarioActualizado);
    }

    // Eliminar usuario prestamo
    @DeleteMapping("/{id}")
    public ResponseEntity<Boolean> deleteUsuarioById(@PathVariable Integer id) throws Exception {
        var isDeleted = usuarioPrestamoService.deleteUsuarioPrestamo(id);
        return ResponseEntity.noContent().build();
    }

    //-------------------------------------PRINCIPALES-----------------------------------------
}
```