



UNIVERSIDAD

Juan Daniel Bogotá Fuentes

Desarrollo orientado por objetos
DOPO

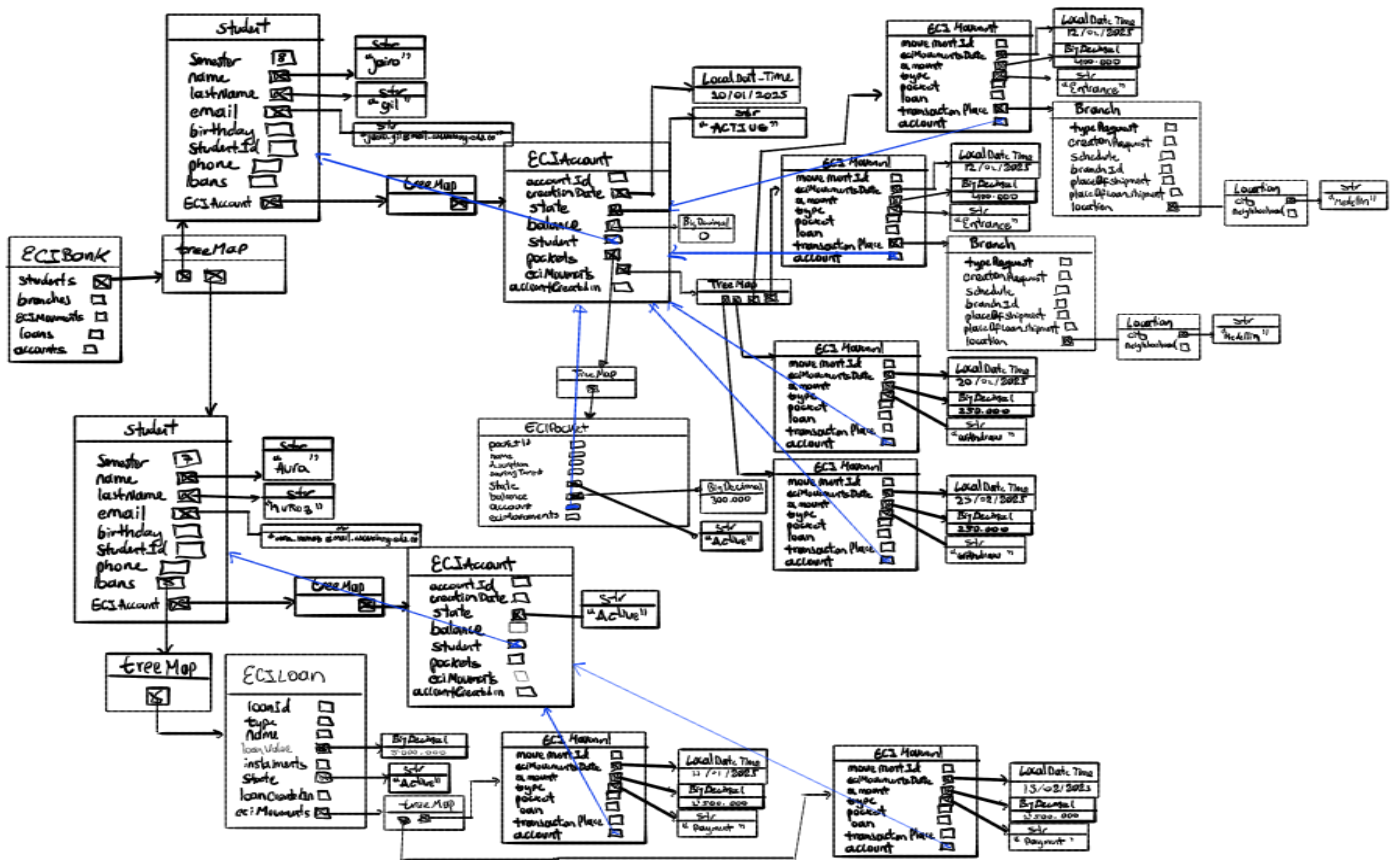
PRE-PARCIAL 1er TERCIO
15/09/2025

PROFESOR: ANGIE TATIANA MEDINA GIL

Contenido

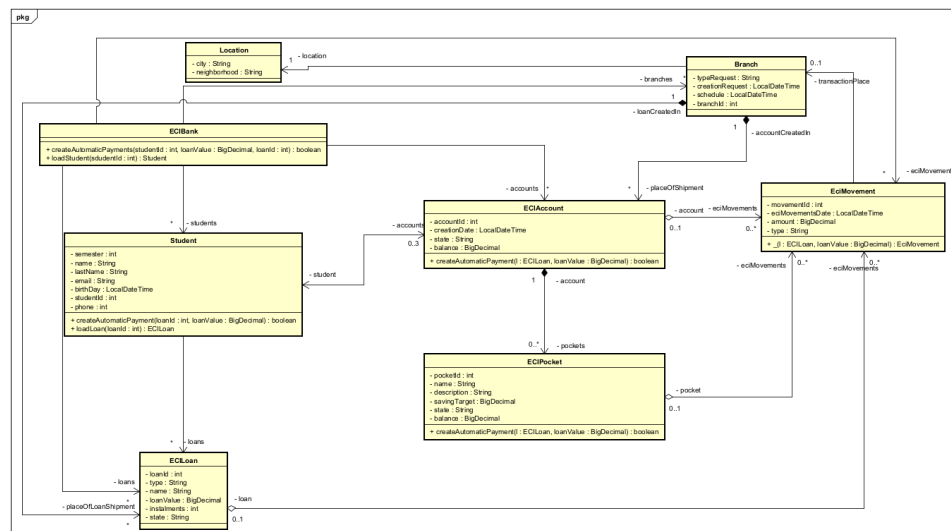
I.	MEMORIA:	3
II.	CÓDIGO:.....	3
III.	DISEÑO:.....	5
IV.	CONCEPTOS:	6

I. MEMORIA:



II. CÓDIGO:

2. Diagrama de clases actualizado.



3. Encabezado y atributos de ECIBank, además de la documentación de la invariante.

```
import java.time.LocalDateTime;
import java.util.TreeMap;
/*
 * Class representing a bank with students, branches, accounts, and loans.
 * Each entity is stored in a TreeMap for efficient access and management.
 * @author Juan Daniel Bogotá
 * @version 1.0
 * @since 2025-09-15
 */

/* Invariante de clase:
 * 1) students, branches, accounts y loans son no nulos.
 * 2) Las llaves en cada TreeMap son únicas.
 * 3) Toda cuenta referencia a un Student existente.
 * 4) Todo préstamo referencia a un Student existente.
 * 5) No deben existir cuentas con identificadores negativos ni préstamos
con montos negativos.
 * 6) No deben existir estudiantes con identificadores negativos.
 */

public class ECIBank {

    private TreeMap<Integer, Student> students;
    private TreeMap<Integer, Branch> branches;
    private TreeMap<Integer, ECIAccount> accounts;
    private TreeMap<Integer, ECILoan> loans;

    public ECIBank() {
        students = new TreeMap<>();
        branches = new TreeMap<>();
        accounts = new TreeMap<>();
        loans = new TreeMap<>();
    }
}
```

Está en la carpeta .zip, una carpeta llamada eciBank, ahí se encuentra todo el proyecto BlueJ.

```
sequenceDiagram
    actor Actor0 as : Actor0
    participant ECIBank as ECIBank
    participant I as I : ECILoan
    participant e as e : ECIAccount
    participant p as p : ECIPocket

    Actor0->>ECIBank: procesarPagosAutomaticos()
    activate ECIBank
    ECIBank->>I: movimiento = getLastMovement()
    activate I
    I->>ECIBank: 
    deactivate I
    loop [i in ECILoan]
        alt [movimiento < 2 meses]
            ECIBank->>I: calculatePayment()
            activate I
            I->>ECIBank: 
            deactivate I
            ECIBank->>I: getStudent()
            activate I
            I->>ECIBank: 
            deactivate I
            ECIBank->>e: checkAvailableBalance(studentId, loanValue)
            activate e
            e->>p: getBalance()
            activate p
            p->>e: 
            deactivate p
            e->>ECIBank: 
            deactivate e
            ECIBank->>e: pagoCuota = payFee(studentId, loanValue)()
            activate e
            e->>p: deductPocketBalance(loanValue)
            activate p
            p->>e: 
            deactivate p
            e->>ECIBank: 
            deactivate e
        else [pagoCuota == true]
            ECIBank->>I: <<create>>  
ECIMovement(loanValue, type)
            activate I
            I->>ECIBank: 
            deactivate I
            ECIBank->>I: updateState(state)
            activate I
            I->>ECIBank: 
            deactivate I
        end
    end
    deactivate ECIBank
```

IV. CONCEPTOS:

- Mencione algunos mecanismos o buenas prácticas para la implementación del código en JAVA.
 - Encapsulamiento: Mantener los atributos como private, así proteger la integridad de los datos y tener acceso controlado.
 - camelCase: para las variables y método, ejemplo: sumarNumeros(int a, int b).
 - Pruebas unitarias: validar que los métodos funcionan según lo esperado y asegurar mantenibilidad del código.
 - Respetar principios SOLID, por ejemplo, el principio de responsabilidad única, cada clase debe tener un único propósito, por ejemplo, ECIMovement solo registra movimientos, no debería calcular préstamos.
- ¿Por qué se dice que Java posee una arquitectura neutral y cómo influye esto en su portabilidad?
 - Java usa una maquina virtual (JVM), esta disponible para muchas plataformas dando compatibilidad, por ejemplo, para Windows, macOS, Android, etc.
 - Es una maquina virtual porque compila el mismo tipo de archivo independiente de la máquina.