

# 1. Descripción General

## 1.1 Objetivo

Desarrollar una aplicación de escritorio en Java (Calculadora V.1) con interfaz gráfica que permita realizar operaciones básicas (suma, resta, multiplicación, división), manejar números decimales con precisión a dos cifras, mantener un historial de operaciones y disponer de botones de borrado (C y AC). El proyecto se gestionará con Git y se alojará en GitHub.

## 1.2 Alcance

- Incluye:
  - Operaciones: +, -, \*, /.
  - Entrada de números con punto decimal.
  - Precisión de salida a 2 decimales.
  - Historial de operaciones con posibilidad de limpiar.
  - Botones: C (borrar entrada actual/último dígito) y AC (borrar todo).
  - Aplicación de escritorio en Java con GUI (JavaFX).
  - Repositorio en GitHub con control de versiones básico.
- No incluye (V.1):
  - Funciones científicas (trigonométricas, potencias avanzadas).
  - Soporte para expresiones complejas con paréntesis múltiples.
  - Internacionalización (i18n), accesibilidad avanzada o temas visuales.

# 2. Requerimientos

## 2.1 Requerimientos Funcionales

**RF-01. Operaciones básicas** La aplicación debe permitir sumar, restar, multiplicar y dividir con operandos ingresados vía interfaz.

**RF-02. Punto decimal** La calculadora debe admitir la entrada de números con punto decimal y mostrar resultados con dos decimales por defecto (configurable).

**RF-03. Historial** La aplicación debe mostrar un historial de operaciones ejecutadas (ej.:  $12.00 + 3.50 = 15.50$ ). Debe permitir limpiar el historial.

**RF-04. Botón C (Clear)** Debe borrar el último dígito o la entrada actual sin afectar la operación previa.

**RF-05. Botón AC (All Clear)** Debe restablecer la calculadora a su estado inicial (pantalla, buffer y operación pendientes, e historial opcional según diseño).

**RF-06. División por cero** La calculadora debe manejar la división por cero mostrando un mensaje de error y evitando el bloqueo de la UI.

**RF-07. Teclado** Permitir interacción con teclado numérico y teclas básicas (0-9, ., +, -, \*, /, Enter, Backspace, Esc).

**RF-08. Persistencia de historial** Permitir que el historial persista entre sesiones en un archivo local.

## 2.2 Requerimientos No Funcionales

**RNF-01. Plataforma** Compatible con Windows y Linux (Java 11+).

**RNF-02. Rendimiento** Operaciones y actualizaciones de UI deben responder en < 100 ms en hardware estándar.

**RNF-03. Usabilidad** Interfaz clara, botones de tamaño adecuado, lectura legible y flujo intuitivo.

**RNF-04. Confiabilidad** Manejo de errores de entrada (múltiples puntos, entradas vacías) sin crashear.

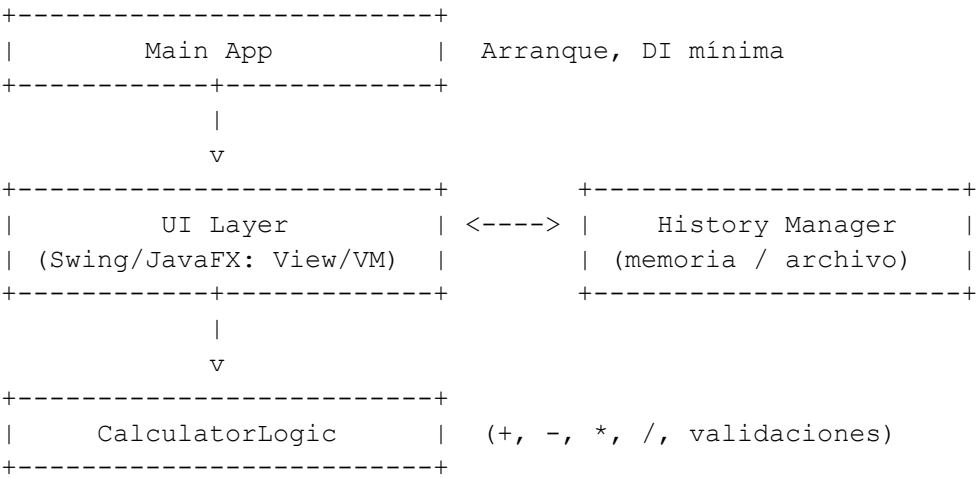
**RNF-05. Mantenibilidad** Separación clara entre lógica de negocio (cálculo) y capa de UI. Cobertura mínima de pruebas unitarias del 70% para la lógica.

**RNF-06. Seguridad** No requiere datos sensibles. Evitar ejecución de código no confiable o acceso a red.

**RNF-07. Portabilidad** Entrega como .jar ejecutable; sin dependencias nativas.

### 3. Arquitectura y Diseño

#### 3.1 Arquitectura (vista lógica)



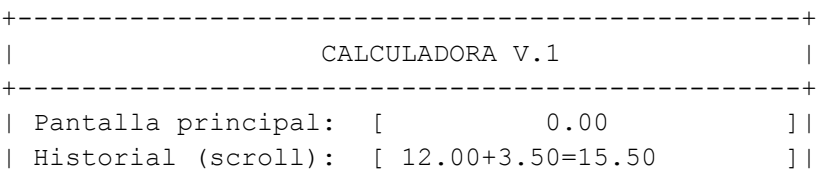
#### 3.2 Estructura de Paquetes (propuesta)

```
src/
└─ com.calcv1/
    ├── app/           (Main)
    ├── ui/            (ventanas, controladores, listeners)
    ├── core/          (CalculatorLogic, operaciones, validación)
    ├── history/        (modelo, persistencia opcional)
    └─ util/           (formatos numéricos, helpers)
```

#### 3.3 Clases principales (borrador)

- Main (app): Inicializa UI.
- CalculatorView (ui): Componentes gráficos (pantalla, botones, historial).
- CalculatorController (ui): Orquesta eventos, comunica UI con lógica.
- CalculatorLogic (core): Implementa operaciones y validaciones.
- HistoryService (history): Maneja el historial en memoria/archivo.
- DecimalFormatter (util): Formatea a 2 decimales (configurable).

### 4. Diseño de Interfaz (Mockup)



|                     |       |         |           |  |  |  |  |  |  |
|---------------------|-------|---------|-----------|--|--|--|--|--|--|
| [ 9.00/0.00=ERROR ] |       |         |           |  |  |  |  |  |  |
| [ AC ]              | [ C ] | [ ÷ ]   | [ × ]     |  |  |  |  |  |  |
| [ 7 ]               | [ 8 ] | [ 9 ]   | [ - ]     |  |  |  |  |  |  |
| [ 4 ]               | [ 5 ] | [ 6 ]   | [ + ]     |  |  |  |  |  |  |
| [ 1 ]               | [ 2 ] | [ 3 ]   | [ = ]     |  |  |  |  |  |  |
| [ 0 ]               | [ . ] | [ +/- ] | [ CLR-H ] |  |  |  |  |  |  |

Leyenda:

- AC: borrar todo
- C: borrar último dígito / entrada actual
- CLR-H: limpiar historial
- +/-: cambia signo (opcional recomendado)

## 5. Casos de Uso

### CU-01: Realizar suma

- Actor: Usuario
- Precondición: App abierta.
- Flujo principal: Ingresar primer número → Presionar + → Ingresar segundo número → =.
- Postcondición: Mostrar resultado con dos decimales y registrar en historial.

### CU-02: Ingresar número con decimales

- Flujo: Ingresar dígitos → . → más dígitos.
- Regla: Un solo . por número.

### CU-03: Borrar entrada (C)

- Flujo: Durante la entrada, presionar C.
- Resultado: Se elimina el último dígito o la entrada actual.

### CU-04: Reiniciar calculadora (AC)

- Resultado: Limpia pantalla, operación en curso y buffers; historial según configuración (no borrar por defecto, salvo que se use CLR-H).

### CU-05: Consultar y limpiar historial

- Flujo: Ver panel de historial; botón CLR-H para limpiar todo.

### CU-06: División por cero

- Flujo: Ingresar a / 0 → =.
- Resultado: Mostrar ERROR sin cerrar la aplicación.

## 6. Reglas de Negocio y Validaciones

- RB-01: Máximo un punto decimal por número.
- RB-02: Redondeo a 2 decimales (modo “half-up” recomendado).
- RB-03: El botón = ejecuta la operación pendiente; una segunda pulsación repite la última operación (opcional).
- RB-04: +/- cambia el signo del número actual (si se implementa).
- RB-05: División por cero produce mensaje de error y no actualiza historial como operación válida (opcional: sí registrar como error).

## 7. Estrategia de Control de Versiones (Git/GitHub)

### 7.1 Flujo de ramas

- main: versión estable.
- develop: integración de features.
- feature/...: ramas por funcionalidad (ej. feature/historial, feature/decimal).

### 7.2 Estructura del repositorio

```

/
├─ README.md
├─ LICENSE
├─ .gitignore
├─ docs/                (este documento, mockups)
├─ src/                 (código fuente)
├─ tests/              (pruebas unitarias)
├─ build/              (artefactos, jar)
├─ .github/
│   └─ ISSUE_TEMPLATE.md
│   └─ workflows/ci.yml  (opcional: build y tests)

```

### 7.3 Convención de commits

- Formato: tipo(scope): resumen
- Tipos: feat, fix, chore, docs, test, refactor.
- Ej.: feat(ui): agregar botón CLR-H para historial.

### 7.4 Versionado Semántico

- V.1 inicial: v1.0.0.
- Parches: v1.0.x (bugs).

- Menores: v1.x.0 (nuevas funciones sin romper compatibilidad).

## 8. Plan de Pruebas

### 8.1 Pruebas Unitarias (lógica)

| ID    | Operacion           | Entrada | Proceso        | Salida esperada | Tipo de salida    |
|-------|---------------------|---------|----------------|-----------------|-------------------|
| PU-01 | Suma simple         | (2,2)   | $2 + 2$        | 4               | Normal            |
| PU-02 | Resta negativa      | (3,5)   | $3 - 5$        | -2              | Numeros negativos |
| PU-03 | Multiplicación      | (1.5,2) | $1.5 \times 2$ | 3               | Normal            |
| PU-04 | División exacta     | (10,2)  | $10 \div 2$    | 5               | Normal            |
| PU-05 | División decimal    | (7,3)   | $7 \div 3$     | 2.33            | Decimales         |
| PU-06 | División por cero   | (5,0)   | $5 \div 0$     | ERROR           | Error             |
| PU-07 | Punto decimal único | (1,2,3) | 1.2.3          | Rechazado       | Error             |

### 8.2 Pruebas Funcionales (UI)

| ID    | Entrada | Entrada | Proceso    | Salida esperada | Tipo de salida |
|-------|---------|---------|------------|-----------------|----------------|
| PF-01 | Tecla 1 | (1)     | Ingresar 1 | 1               | Normal         |
| PF-02 | Tecla 2 | (2)     | Ingresar 2 | 2               | Normal         |
| PF-03 | Tecla 3 | (3)     | Ingresar 3 | 3               | Normal         |
| PF-04 | Tecla 4 | (4)     | Ingresar 4 | 4               | Normal         |

|       |                       |             |   |                           |                  |
|-------|-----------------------|-------------|---|---------------------------|------------------|
| PF-05 | Tecla 5               | (5)         | Ingresar 5                                  | 5                         | Normal           |
| PF-06 | Tecla 6               | (6)         | Ingresar 6                                  | 6                         | Normal           |
| PF-07 | Tecla 7               | (7)         | Ingresar 7                                  | 7                         | Normal           |
| PF-08 | Tecla 8               | (8)         | Ingresar 8                                  | 8                         | Normal           |
| PF-09 | Tecla 9               | (9)         | Ingresar 9                                  | 9                         | Normal           |
| PF-10 | Tecla .               | (.)         | Ingresar .                                  | .                         | Normal           |
| PF-11 | Tecla 0               | (0)         | Ingresar 0                                  | 0                         | Caso con<br>cero |
| PF-12 | Tecla 00              | (00)        | Ingresar 00                                 | 0                         | Caso con<br>cero |
| PF-13 | Tecla 1 y<br>Tecla 00 | (1,00)      | Ingresar 1<br>y 00                          | 100                       | Normal           |
| PF-14 | Tecla Ans             | (100, ANS)  | Ingresar<br>100 = 100<br><br>→ ANS =<br>100 | 100                       | Normal           |
| PF-01 | C tecla               | (C)         | Ingresar<br>123 → C                         | 12                        | Normal           |
| PF-02 | AC                    | (AC)        | Ingresar 45<br>→ AC                         | 0                         | Caso con<br>cero |
| PF-03 | Historial             | (Historial) | 2 + 3 =                                     | Registra<br><br>2 + 3 = 5 | Normal           |
| PF-04 | Decimal               | (1.5,2)     | 1 . 5 + 2 =                                 | 3.50                      | Decimales        |

### 8.3 Criterios de Aceptación

- CA-01: Todas las operaciones básicas funcionan con precisión a 2 decimales.
- CA-02: C y AC operan según definición.
- CA-03: Historial visible, con formato consistente y opción de limpieza.

- CA-04: División por cero manejada sin bloqueo.
- CA-05: Build reproducible y ejecutable .jar generado.

## 9. Cronograma (estimado)

| Fase                  | Duración | Entregables                        |
|-----------------------|----------|------------------------------------|
| Diseño y mockups      | 1 día    | Mockups, estructura de paquetes    |
| Implementación lógica | 1–2 días | CalculatorLogic + tests            |
| UI básica             | 1–2 días | CalculatorView y Controller        |
| Historial y borrados  | 1 día    | HistoryService, botones C/AC/CLR-H |
| Pruebas e integración | 1 día    | Suite de pruebas, correcciones     |
| Empaquetado y README  | 0.5 día  | .jar, documentación                |

## 10. Riesgos y Mitigaciones

| Riesgo  | Impacto | Mitigación   |
|---|---------|--|
| Inconsistencia de formatos decimales por locale | Media   | Fijar Locale.US/formateador propio                 |
| Bloqueo de UI por operaciones                   | Baja    | Ejecutar cálculos ligeros en EDT; validar entradas |
| Falta de pruebas                                | Alta    | Añadir JUnit para core; CI opcional                |
| Manejo de errores incompleto                    | Media   | Casos de prueba para entradas inválidas            |



## 11. Entrega y Empaquetado

Requisitos: Java 11+ (JDK).

Build: mvn package o gradle build (a elegir).

Artefacto: calculadora-v1.jar ejecutable:

```
java -jar build/calculadora-v1.jar
```

## 12. Glosario

- C: Borrar última entrada o dígito.
- AC: Borrar todo (reset).
- Historial: Registro visible de operaciones realizadas.
- Precisión 2 decimales: Formateo/Redondeo a dos cifras decimales (configurable).

## 13. Anexos

### 13.1 Configuración recomendada

- Tecnología GUI: Swing (simplicidad) o JavaFX (más moderno).
- Pruebas: JUnit 5.
- Formateo decimal: DecimalFormat("#0.00") o BigDecimal con RoundingMode.HALF\_UP.

### 13.2 .gitignore (Java/Swing genérico)

```
# Build
/build/
/target/
/out/
*.class

# IDEs
/.idea/
/.vscode/
*.iml

# Sistema
.DS_Store
Thumbs.db
```

```
# Logs
*.log
```

### 13.3 Plantilla de Issue (sugerida)

```
## Descripción
[Explicar el problema o la mejora.]

## Pasos para reproducir (si aplica)
1.
2.
3.

## Resultado esperado
[...]
## Resultado actual
[...]
## Contexto adicional / Capturas
[...]
```

## 14. Resumen Ejecutivo

Calculadora V.3 entregará una app de escritorio en Java con operaciones básicas, manejo de decimales a dos cifras, historial de operaciones y controles de borrado (C y AC). El proyecto se gestionará en GitHub con un flujo de ramas simple, pruebas unitarias para la lógica de cálculo y un empaquetado reproducible en .jar.