



# Escuela Politécnica Nacional

## Facultad de Ingeniería en Sistemas

**ASIGNATURA:** Construcción y evolución de software

**GRUPO:** GR5

**INTEGRANTES:**

- Daniel Mera
- George Quishpe
- Christian Agila

### Análisis de cambio:

1. Planificación:

a. Historias de usuario.

Se agrego dos historias de usuario en las cuales se representa la necesidad de las funcionalidad de “cambio” y “ganancia/perdida”.

Historia de Usuario	
Numero: 4	Usuario: Cliente
Nombre historia: Calculo automático del cambio.	
Descripción:	
Como usuario, deseo que el sistema incluya una columna que muestre la diferencia entre el precio actual y el precio de compra original de cada acción, o cambio, en porcentaje, para cada registro de acciones en la tabla.	
<b>Criterios de aceptación:</b>	
1. El sistema debe calcular automáticamente el cambio para cada registro de acciones en la tabla.	
2. El cambio se calcula como la diferencia entre el precio de compra actual y el precio de compra original por acción.	
3. El sistema debe realizar el cálculo del cambio y la ganancia/pérdida en tiempo real, es decir, tan pronto como haya cambios en los precios de las acciones u otros factores relevantes.	

Historia de Usuario	
Numero: 5	Usuario: Cliente
Nombre historia: Calculo automático del cambio.	
Descripción:	
Como usuario, deseo que el sistema incluya columnas que muestren la diferencia entre el valor actual de las acciones y el costo total original de la transacción, es decir la ganancia o pérdida.	
<b>Criterios de aceptación:</b>	
1. El sistema debe calcular automáticamente la ganancia o pérdida para cada registro de acciones en la tabla.	
2. La ganancia se calcula como la diferencia entre el valor actual de las acciones y el costo total original de la transacción.	
3. La pérdida se calcula como la diferencia entre el costo total original de la transacción y el valor actual de las acciones.	

## 2. Desarrollo:

### a. Codificación:

Se agregaron dos métodos `getCambio()` y `setCambio($cambio)`. Estos métodos permiten obtener y establecer el valor de la propiedad `cambio`.

Se agregaron dos métodos `getGananciaPerdida()` y `setGananciaPerdida($gananciaPerdida)`. Estos métodos permiten obtener y establecer el valor de la propiedad `gananciaPerdida`.

Se agregó un método privado `calcularCambio()` que calcula el cambio porcentual en función de la ganancia o pérdida y el costo total. Este método devuelve el cambio porcentual.

Se agregó un método privado `getPrecioMercadoFromAPI($nombre)` que obtiene el precio de mercado de un activo utilizando una API externa. Este método toma el nombre del activo como parámetro, llama a la API de Finnhub con el símbolo del activo y devuelve el precio actual del mercado. Si hay un error en la solicitud o la respuesta no contiene el precio ('c'), devuelve -1.

```
public function getCambio() {
    return $this->cambio;
}

public function setCambio($cambio) {
    $this->cambio = $cambio;
}

public function getGananciaPerdida() {
    return $this->gananciaPerdida;
}

public function setGananciaPerdida($gananciaPerdida) {
    $this->gananciaPerdida = $gananciaPerdida;
}

private function calcularCambio() {
    if ($this->gananciaPerdida !== null) {
        $cambioPorcentaje = (($this->gananciaPerdida - $this->costoTotal) / $this->costoTotal) * 100;
        return $cambioPorcentaje;
    } else {
        return null; // Maneja el caso en el que no se pueda obtener la ganancia/pérdida
    }
}

private function getPrecioMercadoFromAPI($nombre) {
    $api_key = "c4b4d9e01qks5iv03pgcnb4dj9r01qks5iv03q0";
    $url = "https://finnhub.io/api/v1/quote?symbol=$nombre&token=$api_key";

    $response = file_get_contents($url);

    $data = json_decode($response, true);

    if ($data && isset($data['c'])) {
        return $data['c'];
    } else {
        return -1;
    }
}
```

Se agregó dos nuevos campos (`cambio` y `gananciaPerdida`) a la consulta SQL y ajustan la preparación de la consulta y la vinculación de parámetros para incluir estos dos nuevos valores en la inserción de datos en la tabla de acciones.

```

39 39 // Prepara una consulta SQL para insertar una nueva fila en la tabla 'acciones'.
40 - $result = parent::conectar()->prepare("INSERT INTO acciones (nombre, fechaCompra, precioUnitario, cantidad, costoTotal) VALUES (?, ?, ?, ?, ?)");
41 + $result = parent::conectar()->prepare("INSERT INTO acciones (nombre, fechaCompra, precioUnitario, cantidad, costoTotal, cambio, gananciaPerdida) VALUES (?, ?, ?, ?, ?, ?, ?)");
42
43 // Obtiene los atributos de la instancia de Accion para la inserción.
44 $nombre = $accion->getNombre();
45 $fechaCompra = $accion->getFechaCompra();
46 $precioUnitario = $accion->getPrecioUnitario();
47 $cantidad = $accion->getCantidad();
48 $costoTotal = $accion->getCostoTotal();
49 + $cambio = $accion->getCambio();
50 + $gananciaPerdida = $accion->getGananciaPerdida();
51
52 // Vincula los parámetros de la consulta con los valores obtenidos de la instancia de Accion.
53 $result->bindParam(1, $nombre, PDO::PARAM_STR);
54 $result->bindParam(2, $fechaCompra, PDO::PARAM_STR);
55 $result->bindParam(3, $precioUnitario, PDO::PARAM_STR);
56 $result->bindParam(4, $cantidad, PDO::PARAM_STR);
57 $result->bindParam(5, $costoTotal, PDO::PARAM_STR);
58 + $result->bindParam(6, $cambio, PDO::PARAM_STR);
59 + $result->bindParam(7, $gananciaPerdida, PDO::PARAM_STR);

```

### 3. Pruebas:

#### a. Pruebas de la API Finnhub

Se realizaron las pruebas unitarias de todos los métodos introducidos que hacen uso de la API Finnhub

```

private function getPrecioMercadoFromAPI($nombre) {
    $api_key = "cnb4dj9r01qks5iv03pgcnb4dj9r01qks5iv03q0";
    $url = "https://finnhub.io/api/v1/quote?symbol=$nombre&token=$api_key";

    $response = file_get_contents($url);

    $data = json_decode($response, true);

    if ($data && isset($data['c'])) {
        return $data['c'];
    } else {
        return -1;
    }
}

```

```

private function calcularCambio() {
    if ($this->gananciaPerdida != null) {
        $cambioPorcentaje = (($this->gananciaPerdida - $this->costoTotal) / $this->costoTotal) * 100;
        return $cambioPorcentaje;
    } else {
        return null; // Maneja el caso en el que no se pueda obtener la ganancia/pérdida
    }
}

```

```

public function test_given_oneAction_when_consultPrice_then_true() {
    $accion = new Accion("AAPL", "2024-02-25", 200, 7);
    $precioMercado = $accion->getGananciaPerdida();
    $cambio = $accion->getCambio();
    $this->assertNotEquals(-1, $precioMercado);
    $this->assertNotNull($cambio);
}

```