**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERIA**

**Escuela Profesional de Ingeniería de Sistemas**

****

**INFORME DE PROYECTO UNIDAD III**

**“API OPERACIONES BANCARIAS”**

Curso: Patrones de software

Docente: Mag. Patrick Cuadros Quiroga

**Integrantes:**

**Romero Roque, Angelica Beatriz (2019063327)**

**Chambilla Maquera, Aroceli Noemi (2018060897)**

**Arhuata Turpo, Alex (2019063634)**

**Lima Ccosi, Jeackory Geilly (2019063326)**

**Rojas Bedregal Brian Erik (2018060904)**

**Vilca Condori Erlang Fernando (2019064024)**

**Tacna – Perú**

***2023***

*Sistema API Operaciones bancarias*

Documento de Informe Proyecto Final

Versión *1.0*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| CONTROL DE VERSIONES | | | | | |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | ACHM BRB | ACHM | PCQ | 07/07/2023 11/07/2023 | Versión Original |

INDICE GENERAL

[1. Antecedentes 4](#_Toc105945615)

[2. Planteamiento del Problema 4](#_Toc105945616)

[a. Problema 4](#_Toc105945617)

[b. Justificación 4](#_Toc105945618)

[c. Alcance 4](#_Toc105945619)

[3. Objetivos 5](#_Toc105945620)

[3.1. Objetivo General: 5](#_Toc105945621)

[3.2. Objetivo Especifico: 5](#_Toc105945622)

[4. Marco Teórico 5](#_Toc105945623)

[5. Desarrollo de la Solución 6](#_Toc105945624)

[a. Análisis de Factibilidad (técnico, económica, operativa, social, legal, ambiental) 6](#_Toc105945625)

[b. Tecnología de Desarrollo 6](#_Toc105945626)

[c. Metodología de implementación (Documento de VISION, SRS, SAD) 6](#_Toc105945627)

[6. Cronograma 6](#_Toc105945628)

[7. Presupuesto 7](#_Toc105945629)

[8. Conclusiones 7](#_Toc105945630)

[Recomendaciones 7](#_Toc105945631)

[Bibliografía 7](#_Toc105945632)

[Anexos 7](#_Toc105945633)

[Anexo 01 Informe de Factibilidad 7](#_Toc105945634)

[Anex0 02 Documento de Visión 7](#_Toc105945635)

[Anexo 03 Documento SRS 7](#_Toc105945636)

[Anexo 04 Documento SAD 7](#_Toc105945637)

[Anexo 05 Manuales y otros documentos 7](#_Toc105945638)

**INFORME FINAL**

1. **Antecedentes**

Se considera una operación bancaria a toda actividad que una entidad financiera desarrollada con el fin de ofrecer sus servicios a un cliente. Por esta razón, cualquier operación que conlleve la ejecución de un servicio por parte de una entidad financiera para un cliente es una operación.

La aplicación de patrones de software permite mejorar la arquitectura del software y promover buenas prácticas de diseño y programación. Estos patrones ayudan a estructurar el código de manera modular, promoviendo la cohesión y el bajo acoplamiento entre los componentes del sistema. Además, los patrones de software facilitan la reutilización del código, lo que ahorra tiempo y esfuerzo en el desarrollo de futuros proyectos.

Al utilizar patrones de software, los desarrolladores pueden comunicarse y colaborar de manera más efectiva. Los patrones son soluciones reconocidas y estandarizadas que se encuentran ampliamente documentadas, lo que facilita la comprensión y el intercambio de ideas entre los miembros del equipo.

1. **Planteamiento del Problema**
   1. **Problema**

Las operaciones bancarias en donde las actividades financieras son habituales, prestación de servicios al cliente, una gran cantidad de datos al sistema por tal motivo, se necesita tener un Gestor de Datos.

* 1. **Justificación**

La API de operaciones bancarias en Java se justifica por la necesidad de ofrecer servicios financieros eficientes a los clientes de un banco. Además, una API bien diseñada puede mejorar la experiencia del usuario y aumentar la satisfacción del cliente.

Aplicar patrones de software en una cuenta bancaria brinda beneficios importantes como la seguridad de la información, el manejo eficiente de transacciones complejas, la facilidad de mantenimiento y evolución del sistema, la reutilización de código y el cumplimiento normativo. Estas justificaciones respaldan la importancia de utilizar patrones de software en el desarrollo de soluciones para cuentas bancarias.

* 1. **Alcance**

El alcance del proyecto es la creación de una API de operaciones bancarias en Java que permita a los clientes realizar transacciones financieras eficiente. La API debe ser capaz de manejar transacciones como pagos de tarjetas de crédito, consultas de saldo, depósitos, retiros y transferencias, y debe incluir medidas de seguridad sólidas para proteger la información de los clientes.

De igual manera un patrón de software en una cuenta bancaria es especialmente importante debido a la naturaleza crítica y sensible de la información financiera y los riesgos asociados.

1. **Objetivos**
   1. **Objetivo General:**

Desarrollar una API de operaciones bancarias en Java que proporcione servicios financieros eficientes y seguros a los clientes de un banco.  
De igual manera nuestro objetivo patrones de software que aplicaremos en nuestro proyecto nos proporcionara soluciones probadas y efectivas a problemas comunes en el diseño y desarrollo de software.

* 1. **Objetivo Especifico:**

Diseñar la arquitectura de la API, incluyendo los métodos y funcionalidades necesarias para realizar transacciones financieras, como pagos de tarjetas de crédito, consultas de saldo, depósitos, retiros y transferencias. Los patrones de software son soluciones que han demostrado su eficacia y han sido validadas por la comunidad de desarrollo de software.

Estas soluciones han sido utilizadas en numerosos proyectos y han sido refinadas y optimizadas a lo largo del tiempo. Al aplicar un patrón de software, se puede confiar en que se está utilizando una solución probada y confiable para el problema específico que se está abordando.

1. **Marco Teórico**

* Sistema Web: **T**ambién conocido como «aplicaciones Web» son aquellos que están creados e instalados no sobre una plataforma o sistemas operativos (Windows, Linux). Sino que se alojan en un servidor en Internet o sobre una intranet (red local). se pueden utilizar en cualquier navegador Web (chrome, firefox, Internet Explorer, etc.) sin importar el sistema operativo. Para utilizar las aplicaciones Web, no es necesario instalarlas en cada computadora ya que los usuarios se conectan a un servidor donde se aloja el sistema.
* Java: Por lo general Java sirve para crear aplicaciones y procesos en una gran diversidad de dispositivos. Se basa en una programación orientada a objetos, permite ejecutar un mismo programa en múltiples sistemas operativos y hace posible ejecutar el código en sistemas remotos de manera segura.
* API: Significa “interfaz de programación de aplicaciones”. En el contexto de las API, la palabra aplicación se refiere a cualquier software con una función distinta. La interfaz puede considerarse como un contrato de servicio entre dos aplicaciones.
* Patrones de diseño:Los patrones de diseño son unas técnicas para resolver problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño.
* Patrones de creación (o creacionales): Definen cómo puede crearse un objeto. Habitualmente esto incluye aislar los detalles de la creación del objeto, de forma que su código no dependa de los tipos de objeto que hay y, por lo tanto, no deba se modificado al añadir un nuevo tipo de objeto.
* Patrones estructurales: Tratan la manera en que los objetos se conectan con otros objetos, para asegurar que los cambios del sistema no requieren cambiar esas conexiones.
* Patrones de comportamiento: Tratan a los objetos que manejan tipos particulares de acciones dentro de un programa. Éstos encapsulan procesos debe ejecutarse dentro de la funcionalidad de la aplicación, como interpretar un lenguaje, completar una petición, moverse a través de una secuencia o implementar un algoritmo.
* Patrones de arquitectura de microservicios: Se refiere a tipos de diseño o características comunes para el manejo de estos sistemas, dependiendo de sus necesidades y requerimientos. Algunos de estos patrones pueden trabajarse de forma continua para obtener mejores resultados.
* Proceso Unificado de Rational (RUP): es una metodología que tiene como objetivo ordenar y estructurar el desarrollo de software, en la cual se tienen un conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema

1. **Desarrollo de la Solución**
   1. **Análisis de Factibilidad**
2. **Análisis Técnico:**

La siguiente lista se considerará el hardware y software necesario para la implementación del proyecto.

Tabla Nº1. Recursos Necesarios

|  |  |
| --- | --- |
| Recursos Necesarios | |
| Hardware y Software | Especificación |
| Sistema Operativo | Windows 10 Professional 64 bits |
| Editor de código | Visual Studio Code |
| Internet | Claro 80 Mbps |
| Kit de Arduino | Componentes electrónicos |
| Editor de documentos | Microsoft Office 365 |

Fuente: Elaboración propia del equipo de trabajo

1. **Análisis Económico:**

* **Costo General**

Los costos de implementación incluyen comúnmente el costo hardware y software los costos de operación del sistema y los costos de mano de obra materiales, energía, reparaciones y mantenimiento entre otros recursos adicionales.

Tabla Nº2. Costos Generales

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Descripción** | **Cantidad** | **Costos**  **Mensual** | **Tiempo**  **(Meses)** | **Costo Total** |
| Papeles | 1 (500) Hojas | S/ 10.00 | 3 | S/. 10.00 |
| Lapiceros | 3 | S/.7.60 | 3 | S/. 15.20 |
| Folder | 15 | S/.7.50 | 3 | S/. 15.00 |
| Tinta Impresora | 2 | S/.36.0 | 3 | S/. 72.00 |
| **Total** |  | S/. 61.10 |  | S/.122.20 |

Fuente: Elaboración propia del equipo de trabajo

* **Costos operativos durante el desarrollo**

Tabla Nº3. Costos Operativos

|  |  |  |  |
| --- | --- | --- | --- |
| **Descripción** | **Costo Mensual** | **Tiempo** | **Subtotal** |
| Luz | S/.25.00 | 3 meses | S/. 75.00 |
| Agua | S/.10.00 | 3 meses | S/. 30.00 |
| **Total** | | | S/. 105.00 |

Fuente: Elaboración propia del equipo de trabajo

* **Costos del ambiente**

Materiales y herramientas para la implementación del proyecto

Tabla Nº4. Costos del Ambiente

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Hardware -**  **Software** | **Descripción** | **Costo Mensual** | **Duración** | **Cantidad** | **Costo**  **Total** |
| Licencias  Windows | Windows 10  Profesional | S/.57.70 | 1 año | 2 | S/.115.4 |
| Internet | Claro 80 Mbps | S/.90.00 | 3 meses | 1 | S/.270.00 |
| Antivirus | Malware bytes | S/.27.00 | 3 meses | 1 (5 dispositivos) | S/.81.00 |
|  |  | S/.174.91 |  | Total | S/.466.40 |

Fuente: Elaboración propia del equipo de trabajo

* **Costos de personal**

Dentro de los gastos presupuestados que se necesita para el desarrollo del proyecto los cuales están considerados los pagos mensuales a cada uno de los integrantes del equipo de desarrollo y horarios de trabajos de entrada 8:00 am y salida.4:00pm.

Tabla Nº5. Costos del Personal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CARGO** | **Horario** | **Cantidad Horas** | **Costo Mensual** | **Tiempo** | **Subtotal** |
| Jefe de Proyecto | Lunes a viernes | 8 | S/.2.200 | 3 meses | S/.6.600 |
| Analista | Lunes a viernes | 8 | S/.2.000 | 3 meses | S/.6.000 |
| Programador | Lunes a viernes | 8 | S/.1.800 | 3 meses | S/.5.400 |
| Programador | Lunes a viernes | 8 | S/.1.800 | 3 meses | S/.5.400 |
| Documentador | Lunes a viernes | 8 | S/.1.700 | 3 meses | S/.5.100 |
| Total | | | **S/. 11.300** |  | **S/. 28.500** |

Fuente: Elaboración propia del equipo de trabajo

1. **Análisis Operativo**

Nuestro sistema de API operaciones bancarias aplicando patrones de Software es una herramienta útil y factible donde permite optimizar que nuestro API es capaz de integrase de manera efectiva con los sistemas existentes en el entorno bancario, como sistema de Pagar tarjetas de crédito, Realizar consulta del saldo, Gestionar depósitos, Gestionar retiros, Gestionar transferencias.

1. **Análisis Legal**

* **Ley 29733 – Ley de Protección de Datos Personales y su Reglamento**

Regula las exigencias legales que debe cumplir una empresa que recibe, recopila, usa, solicita, almacena o suministra información sobre una persona natural.

1. **Análisis Social**

El objetivo es ayudar en el proceso para que se desarrolle con agilidad, eficiencia, más que nada API es establecer una interfaz y conjunto de reglas que permitan la comunicación y la interacción entre diferentes componentes de software.

Las API son utilizadas para definir la forma en que un software o servicio expone sus funcionalidades y datos, permitiendo a otros programas o sistemas interactuar con ellos de manera controlada.

1. **Análisis Ambiental**

El proyecto no tiene influencias y asuntos de índole ambiental como el impacto y repercusión en el medio ambiente.

* 1. **Tecnología de Desarrollo**

|  |  |
| --- | --- |
| **Software** | **Especificaciones** |
| Sistema Operativo | Windows 10 (64-bits) |
| Editor de Código | * Visual Code |
| * Net.6 |
| Lenguaje de programación | * Java |
| Control de Versiones | Github |
| Diseño de Diagrama UML | Diagrams.net |
| Documento | Google Docs |
| Colaboración | * Google Meets |
| * One Drive |
| Patrones | * Repository |
| * Data Transfer Object |
| * Service |
| * Fabrica |
| * Builder |

* 1. Metodología de implementación
* Nivel de análisis:

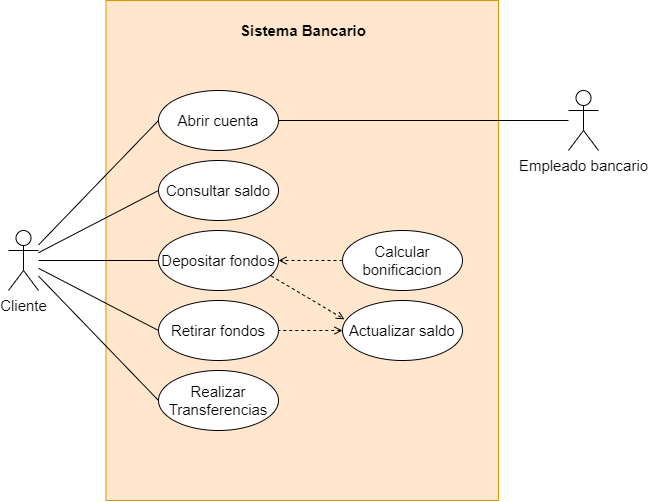
RUP y UML: RUP es una metodología de desarrollo de software y UML es un Lenguaje Unificado de Modelado para especificar, construir y documentar los artefactos de nuestro sistema.

* Artefactos:
  1. Diagrama de Componentes

Diagrama

Descripción generada automáticamente

* 1. Diagrama de casos de uso



* 1. Diagrama De Clases

Diagrama

Descripción generada automáticamente

**6. Cronograma**

*Tabla Nº1: cronograma del proyecto*

|  |  |
| --- | --- |
| **Fases** | **Duración** |
| **INICIO** | 08 /04/2023 hasta 22/04/2023 |
| **ELABORACIÓN** | 06/05/2023 hasta 27/05/2023 |
| **CONSTRUCCIÓN** | 03/06/2023 hasta24/06/2023 |
| **TRANSICIÓN** | 01/07/2023 hasta 08/07/2023 |

*Fuente: elaboración propia*

**7. Presupuesto**

Costos totales del desarrollo del sistema

|  |  |
| --- | --- |
| **Descripción** | **Costo Aproximado** |
| Costos Generales | **S/.122.20** |
| Costo de Ambiente | **S/. 466.40** |
| Costo de Personal | **S/. 28,500.00** |
| Costo Operativo | **S/. 105.00** |
| **Total** | **S/.29,193.60** |

**8. Conclusiones**

Se concluye que el presente proyecto puede ofrecer numerosas ventajas y funcionalidades para los usuarios y las instituciones financieras, puede proporcionar automatización, acceso en tiempo real a información financiera, seguridad, mejor experiencia del usuario, integración con otros servicios y adaptabilidad. Estas características contribuyen a una gestión más eficiente y efectiva de las transacciones y servicios bancarios en línea.

# **Recomendaciones**

Mantén actualizadas las versiones de la API: Asegurarse de estar al tanto de las actualizaciones y cambios en la API bancaria utilizada. Mantén tu sistema web actualizado con las últimas versiones de la API para aprovechar nuevas características, solucionar problemas conocidos y garantizar la compatibilidad con los cambios futuros.

Capacitación y soporte al usuario: Brinda capacitación y soporte adecuados a los usuarios del sistema web para que comprendan cómo utilizar la funcionalidad relacionada con la API bancaria. Proporciona canales de comunicación claros para que los usuarios puedan solicitar asistencia en caso de problemas o consultas.

Pruebas exhaustivas: Realiza pruebas exhaustivas en todas las funcionalidades del sistema, especialmente en las interacciones con la API bancaria. Verifica la integridad de los datos, la precisión de los cálculos y la fiabilidad de las transacciones. Esto garantizará que el sistema funcione correctamente y proporcione una experiencia de usuario satisfactoria.

# **Bibliografía**

1. Overview (java SE 16 & JDK 16). (2021, julio 20). Oracle.com. <https://docs.oracle.com/en/java/javase/16/docs/api/index.html>

Spring.io. Recuperado de <https://spring.io/docs>

1. Lista de API de JAVA. (n.d.). Oracle.com. Recuperado de <https://docs.oracle.com/applications/help/es/enterprise-performance-management/11.2/FDMAD/java_api_list.htm>
2. KarlErickson. (n.d.). Protección de la API REST mediante Spring Cloud Azure. Microsoft.com. Recuperado de <https://learn.microsoft.com/es-es/azure/developer/java/spring-framework/secure-your-restful-api-using-spring-cloud-azure>
3. PhpMyAdmin 5.2.2. (n.d.). Localhosts.Mobi. Recuperado de <https://localhosts.mobi/phpmyadmin>
4. Baeldung.com. Recuperado de <https://www.baeldung.com/postman-testing-collections>

# **Anexos**

**ANEXO 01- PRINCIPIO DE DISEÑO**

**ANEXO 02- PATRONES DE DISEÑO ESTRUCTURALES DE COMPORTAMIENTO, CREACIONALES.**

**ANEXO 03- PATRONES DE SIEÑO DE PRESENTACION O DE MICROSERVICIOS**