

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD: INFORMÁTICA Y ELECTRÓNICA

**ESCUELA DE INGENIERÍA EN SISTEMAS**

**CARRERA: SOFTWARE**

# GUÍA DE LABORATORIO DE APLICACIONES INFORMÁTICAS II

## PRÁCTICA No

1. **DATOS GENERALES:**

**NOMBRE DEL ESTUDIANTE CODIGO DEL ESTUDIANTE**

Katherin Narváez 6779

**FECHA DE REALIZACIÓN: FECHA DE ENTREGA:**

**2025/07/18 2025/07/18**

1. **OBJETIVO(S):**

**2.1.GENERAL**

Desarrollar la fase de documentación adjunto al proceso de ingeniería inversa sore una aplicación de software ya existente la cual fue analizada y valorada, con el fin de comprender sus componentes, arquitectura, su funcionamiento y el análisis de consideraciones para que faciliten su mantenimiento.

**2.2.ESPECÍFÍCOS**

-Recopilar información del análisis anteriormente realizado y la aplicación de ingeniería inversa.

- Documentar la aplicación de ingeniería inversa sobre el aplicativo informático elegido en base a lo visto en clases.

1. **METODOLOGÍA**

En el presente documento se realizara la documentación de la aplicación de la técnica de ingeniería inversa aplicada anteriormente sobre un software existente. Mediante el análisis estático y dinámico.

1. **EQUIPOS Y MATERIALES:**

* Computador
* Entorno integrado de desarrollo (IDE)
* Aula virtual
* Acceso a internet
* Bibliografía

1. **MARCO TEORICO:**

**Proceso de ingeniería inversa:**

La ingeniería inversa puede ser un proceso complejo, pero generalmente sigue algunos pasos clave:

1. **Preparación**

Se debe recopilar mucha información antes de poder realizar la ingeniería inversa. Esto se logra familiarizándose con el producto o software que se va a revertir, leyendo libros si están disponibles y adquiriendo las herramientas necesarias.

En el caso del hardware, esto podría consistir simplemente en desmontar el dispositivo con mucho cuidado, y en el caso del software, generalmente se empieza observando la composición del programa.

**2. Desmontaje o des compilación**

En la ingeniería inversa de software, el siguiente paso suele consistir en desensamblar o descompilar el código para ver su estructura básica. En hardware, este paso implica analizar cuidadosamente los componentes del dispositivo, como sus placas de circuito o chips, para comprender cómo interactúan.

**3. Análisis**

Aquí es donde se produce la mayor parte de la ingeniería inversa. En software, se realiza mediante el análisis de código para comprender la lógica del programa, sus vulnerabilidades y cómo mejorarlo o modificarlo. En hardware, se puede realizar mediante el estudio de cableado y la comprensión de cómo interactúan los componentes entre sí.

**4. Documentación e informes**

Una vez finalizado el análisis, conviene registrar los resultados. Esto podría consistir en crear un informe completo sobre el diseño y la funcionalidad del producto o en registrar las vulnerabilidades descubiertas en un único programa de software. La documentación contribuye a profundizar en el conocimiento de los elementos descubiertos y proporciona orientación para futuros desarrollos o trabajos correctivos.

**5. Explotación o mejora**

Tras el proceso de análisis, el siguiente paso consiste en explotar las vulnerabilidades descubiertas, si se es investigador de seguridad, o bien mejorar el diseño de un producto existente. Por ejemplo, tras revertir un análisis de malware, se podría desarrollar un parche o contramedidas. De lo contrario, tras el análisis de un producto, se puede proceder al diseño de uno mejor o diferente.

1. **PROCEDIMIENTO DOCUMENTACION:**

**Información del Proyecto**

|  |  |
| --- | --- |
| Empresa / Organización | ESPOCH |
| Proyecto | **Sistema de Facturación(Veterinaria)** |
| Fecha de preparación | 07/18/2025 |

1. **HISTORIAL DE CAMBIOS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Organización** | **Descripción** |
| 07/18/2025 | 1.0 | ESPOCH | Análisis del proyecto existente |
| 07/18/2025 | 1.0 | ESPOCH | Aplicación de ingeniería inversa, análisis estático y dinámico. |
| 07/18/2025 | 1.2 | ESPOCH | Elaboración de diagramas |
| 07/18/2025 | 1.0 | ESPOCH | Elaboración del proceso de documentación |
| 07/18/2025 | 1.0 | ESPOCH | Retroalimentación de información, análisis y resultados |

**6.1 Proceso de Análisis**

**6.1.1 Reconstrucción de la Arquitectura**

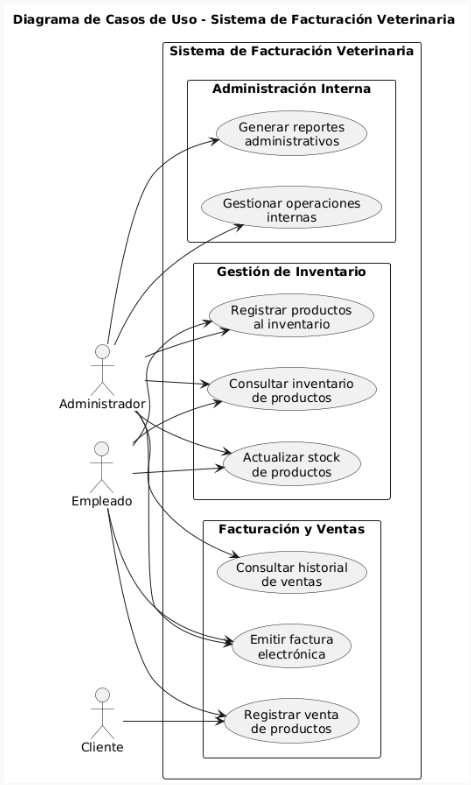
1. **CONFIGURACIONES CRITICAS.**

**Detalles de entorno:**

|  |  |
| --- | --- |
| Elemento | Descripción |
| Servidor | **Localhost (localhost) usando servidor embebido de PHP o XAMPP** |
| IDE | **Visual Studio Code (VS Code) con extensiones para PHP, HTML, CSS** |
| Lenguaje/Framework | **PHP 7.x o superior, sin frameworks externos (estructura personalizada)** |
| Base de Datos | **MySQL, definida en archivo conexion.php** |
| Servicios | **Navegador (Chrome/Firefox), ejecución local con conexión a base de datos** |
| Observaciones | **Entorno usado para probar validaciones de formularios, lógica de negocio y flujo básico de facturación e inventario** |

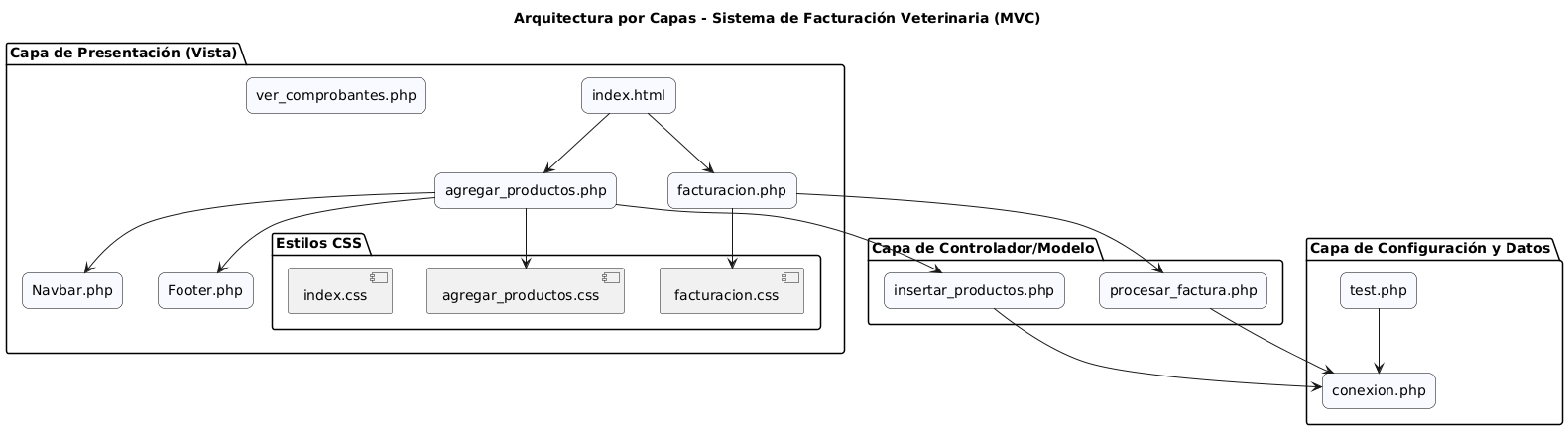
1. **DIAGRAMAS DE FLUJO**

* **Diagrama de Casos de Uso :**

****

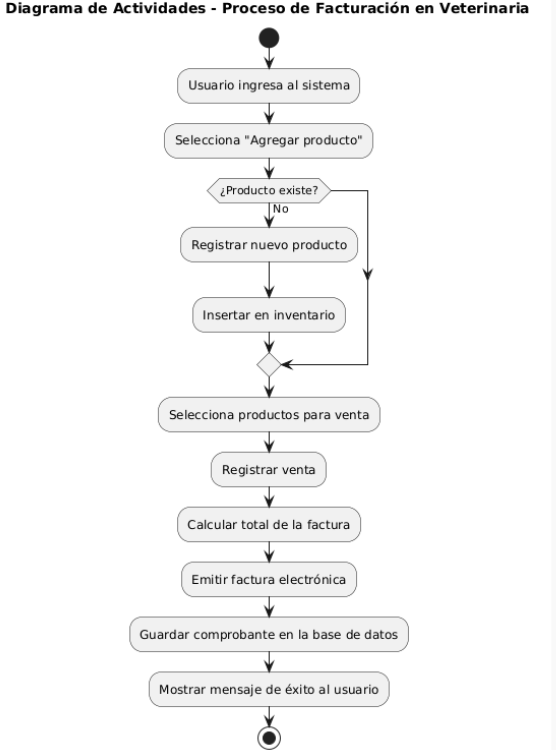
Este diagrama muestra las principales funcionalidades que el usuario puede realizar al interactuar con la aplicación. En este caso, representa cómo el usuario. Este diagrama es útil para entender las necesidades del usuario y los servicios que el sistema debe ofrecer desde su perspectiva.

**Diagrama de la Arquitectura**

****

* **Diagrama de Procesos del Sistema**

Estos diagramas describen el flujo funcional del sistema, mostrando paso a paso las acciones que se ejecutan desde que el usuario inicia una operación hasta que se presenta el resultado. Incluye la entrada de datos, selección de la operación, validaciones y la visualización del resultado.

****

1. **GLOSARIO DE TÉRMINOS**
2. **PROCEDIMIENTO DE DESPLIEGUE Y RECUPERACIÓN**
3. **CONCLUSIONES Y RECOMENDACIONES:**

**Conclusiones:**

El análisis mediante ingeniería inversa permitió identificar con claridad los componentes clave del sistema, su arquitectura en capas (presentación, lógica y datos) y las dependencias entre ellos. Sin embargo, el proceso reveló limitaciones en la documentación interna, lo que dificultó la comprensión de ciertos módulos complejos, como el manejo de transacciones en la pasarela de pagos. Además, la falta de comentarios descriptivos en el código y la escasa trazabilidad entre los requisitos y su implementación complicaron el mapeo completo de funcionalidades.

Para finalizar podemos concluir que estas mejoras abordan seguridad, rendimiento, usabilidad y escalabilidad, basadas en el análisis y aplicación de la ingeniería inversa. Se recomienda comenzar con las de alta prioridad para reducir riesgos críticos.

**Recomendaciones:**

Para optimizar futuros ejercicios de ingeniería inversa, se recomienda implementar una documentación técnica detallada que incluya diagramas de secuencia y flujo de datos críticos.Estas mejoras no solo agilizarían el proceso de ingeniería inversa, sino que también facilitarían el mantenimiento y la evolución del sistema, asegurando una base más sólida para posteriores actualizaciones.

1. **BIBLIOGRAFÍA:**

* Eilam, E. (2005). *Reversing: Secrets of reverse engineering*. Wiley. <https://doi.org/10.1002/0471756978>
* Pressman, R. S. (2010). Ingeniería del Software: Un Enfoque Práctico (7ª ed.). McGraw-Hill. (Capítulo 29: "Reengineering and Reverse Engineering")
* Sommerville, I. (2011). Software Engineering (9ª ed.). Pearson. (Sección 27.4: "Reverse Engineering")
* Chikofsky, E. J. y Cross, J. H. (1990). Ingeniería inversa y recuperación de diseño: Una taxonomía. *IEEE Software, 7*(1), 13-17. <https://doi.org/10.1109/52.43044>