INSTITUTO SUPERIOR TECNOLÓGICO “SAN GABRIEL”

CONDICIÓN UNIVERSITARIO

INFORME DE PRÁCTICAS LABORALES II

PREVIO A LA OBTENCIÓN DEL TÍTULO DE

TECNÓLOGO SUPERIOR UNIVERSATORIO EN DESARROLLO DE SOFTWARE

LUGAR DE REALIZACIÓN

_____

PRACTICANTE

_____

MES - AÑO

RIOBAMBA – ECUADOR

ÍNDICE

1. Introducción 5

2. Información general 6

2.1 Institución donde se realizaron las prácticas 6

2.1.1 Razón Social: 6

2.1.2 Reseña Histórica: 6

2.1.3 Misión 6

2.1.4 Visión 6

2.2 Duración de las prácticas 6

2.2.1 Fecha de Inicio: 6

2.2.2 Fecha de finalización: 6

2.2.3 Número total de horas: 6

2.3 Área del conocimiento en la que se realizaron las prácticas: 6

2.4 Asignaturas articuladas a prácticas pre profesionales laborales 7

2.5 Tutor institución Sede de las Prácticas 7

2.5.1 Nombres y Apellidos: 7

2.5.2 Cédula de Ciudadanía: 7

2.5.3 Teléfono: 7

2.5.4 Correo electrónico: 7

2.6 Docente tutor de prácticas Pre – Profesionales del Instituto 7

2.6.1 Nombres y Apellidos: 7

2.6.2 Cédula de Ciudadanía: 7

2.6.3 Teléfono: 7

2.6.4 Correo electrónico: 7

3. Desarrollo de las Actividades 8

3.1 Actividades desarrolladas durante las prácticas laborales 8

3.1.1 Conocer los conceptos fundamentales de programación orientada a objetos (POO) y comprender la sintaxis básica de Java 8

3.1.2 Describir las características clave de Java, como la portabilidad, orientación a objetos, y la gestión

automática de memoria _____8

3.1.3 Crear programas simples en Java siguiendo la estructura básica, incluyendo la declaración de paquetes, importaciones, función principal, y aplicar buenas prácticas de codificación para organizar y comentar programas de manera efectiva. _____9

3.1.4 Identificar y utilizar correctamente los tipos de datos primitivos en Java, como enteros, decimales, caracteres y booleanos y selecciona el tipo de dato apropiado según la naturaleza de la información que se manejan en el programa._____9

3.1.5 Utilizar correctamente operadores aritméticos, de asignación, de comparación y lógicos en Java para realizar operaciones matemáticas y manipulación de datos, explica la precedencia de los operadores y utilizar paréntesis para controlar el orden de evaluación._____10

3.1.6 Manipular cadenas de texto, incluyendo la concatenación, extracción de subcadenas y búsqueda de caracteres, utilizar métodos de la clase String para realizar operaciones comunes, como la conversión de mayúsculas a minúsculas y viceversa._____10

3.1.7 Utilizar diferentes métodos para leer datos desde la entrada estándar (teclado) en programas Java, como la clase Scanner o BufferedReader, y maneja excepciones relacionadas con la entrada de datos, como la entrada de datos no válidos. _____11

3.1.8 Utilizar correctamente las sentencias condicionales (if, else if, else) para controlar el flujo de ejecución de un programa, anida sentencias condicionales para manejar situaciones más complejas y comprenderán la importancia de las estructuras de control de flujo en la toma de decisiones en un programa. _____11

3.1.9 Implementar bucles (for, while, do-while) para realizar iteraciones y repetir bloques de código, identificar situaciones en las que el uso de bucles es apropiado y selecciona el tipo de bucle más adecuado para una tarea específica. _____12

3.1.10 Definir y utilizar clases en Java, comprendiendo la diferencia entre una clase y un objeto, crean (objetos) de una clase, inicializa sus atributos y utiliza métodos asociados, conoce conceptos clave como encapsulamiento, herencia y polimorfismo en el contexto de las clases y los objetos._____13

3.1.11 Declarar métodos para modular y organizar código, entender la importancia de los parámetros y el valor de retorno en la creación y llamada de métodos, aplica conceptos como la descarga sobre métodos que contribuyen a la modularidad y reutilización del código. _____13

3.1.12 Aplicar y comprender los modificadores de acceso (públicos, privados, protegidos) para controlar el acceso a los campos y métodos de una clase. _____14

3.1.13 Comprender la diferencia entre sobrecarga y sobreescritura de métodos, y es capaz de implementar ambos conceptos correctamente en sus programas._____14

3.1.14 Diseñar y utilizar la herencia en Java, creando superclases y subclases para aprovechar la reutilización del código y la creación de jerarquías de clases. _____15

3.1.15 Entender el concepto de polimorfismo y ser capaces de implementarlo a través de la sobre escritura de métodos y la referencia a objetos de clases superiores. 15

3.1.16 Diseñar y utilizar clases abstractas para proporcionar una base común para clases derivadas, así como entender cómo implementar métodos abstractos. 16

3.1.17 Organizar su código en paquetes para modularizar y estructurar proyectos de manera eficiente, además de entender cómo importar clases de otros paquetes. 16

3.1.18 Manejar excepciones, utilizando bloques try-catch, y entiende la jerarquía de excepciones en Java para manejarlas de manera efectiva. 16

3.1.19 Diseñar interfaces gráficas de usuario (GUI) utilizando la biblioteca Swing de Java. 17

3.1.20 Comprender el modelo de programación basado en componentes de JSF y desarrollar aplicaciones web Java utilizando JSF, diseña páginas de forma declarativa y gestionar eventos del lado del servidor. 17

3.1.21 Conectar a bases de datos utilizando JDBC, realizar consultas SQL desde Java y gestionar transacciones y entiende los conceptos de ResultSet, Statement y Connection en el contexto de JDBC. 18

3.2 Recursos 18

4. Habilidades Desarrolladas 19

4.1 Impacto de las prácticas pre profesionales laborales en la sociedad 20

5. Marco Teórico 20

5.1 Programación Orientada a Objetos 20

6. Conclusiones 21

7. Recomendaciones 22

8. Referencias Web - Bibliográficas 23

9. Anexos 24

9.1 Anexo 1 25

Hoja de Evaluación 25

9.2 Anexo 2 26

Certificado de prácticas 26

9.3 Anexo 3 27

Registro de asistencia 27

1. Introducción

En el ámbito del desarrollo de software, las prácticas laborales representan una oportunidad única para aplicar y perfeccionar las habilidades adquiridas en las aulas. Durante este periodo, tendrán la oportunidad de sumergirse en entornos profesionales, colaborar con equipos multidisciplinarios y enfrentarse a desafíos del mundo real.

El desarrollo de software es una disciplina dinámica que requiere no solo habilidades técnicas sólidas, sino también la capacidad de adaptarse a cambios constantes y resolver problemas de manera eficiente. A lo largo de las prácticas, deberá explorar nuevas tecnologías, participar activamente en proyectos desafiantes y aprovechar cada oportunidad para aprender de sus compañeros y mentores.

Según el Art. 89 del Reglamento de Régimen Académico del Consejo de Educación Superior, las Prácticas pre profesionales son: *"actividades de aprendizaje orientadas a la aplicación de conocimientos y al desarrollo de destrezas y habilidades específicas, que un estudiante debe adquirir para un adecuado desempeño en su futura profesión. Estas prácticas deberán ser de investigación-acción y se realizarán en el entorno institucional, empresarial o comunitario, público o privado, adecuado para el fortalecimiento del aprendizaje. Las prácticas pre profesionales o pasantías son parte fundamental del currículo conforme se regula en el presente Reglamento".* (pág. 35)

Por lo mencionado, se realiza este primer bloque de prácticas, gracias a la aceptación de la empresa: _____, gerente propietario _____, está ubicada en las calles _____.

El período de prácticas inicia el día _____ de _____ y finaliza el _____ del año _____, cumpliendo con los siguientes horarios, desde las __h00 am hasta las ___H00 pm, y de _____H00 hasta las ____H00, de lunes a viernes, cumpliendo con un total de 120 horas, las prácticas son

supervisadas por el (Cargo y Nombre:). _____.

En las practicas 2 se tiene como materia articulada la asignatura Programación Java. Finalmente se indica que el presente informe tiene la siguiente estructura: Introducción, Objetivos, Desarrollo del tema, Conclusiones, Recomendaciones y Referencias bibliográficas.

2. Información general

2.1 Institución donde se realizaron las prácticas

2.1.1 Razón Social:

2.1.2 Reseña Histórica:

2.1.3 Misión

2.1.4 Visión

2.2 Duración de las prácticas

2.2.1 Fecha de Inicio:

- _____

2.2.2 Fecha de finalización:

- _____

2.2.3 Número total de horas:

- 120 horas

2.3 Área del conocimiento en la que se realizaron las prácticas:

- Desarrollo de software

2.4 Asignaturas articuladas a prácticas pre profesionales laborales

- Lenguaje de programación Java

2.5 Tutor institución Sede de las Prácticas

2.5.1 Nombres y Apellidos:

- _____

2.5.2 Cédula de Ciudadanía:

- _____

2.5.3 Teléfono:

- _____

2.5.4 Correo electrónico:

- _____

2.6 Docente tutor de prácticas Pre – Profesionales del Instituto

2.6.1 Nombres y Apellidos:

- _____

2.6.2 Cédula de Ciudadanía:

- _____

2.6.3 Teléfono:

- _____

2.6.4 Correo electrónico:

- _____

3. Desarrollo de las Actividades

3.1 Actividades desarrolladas durante las prácticas laborales

3.1.1 Conocer los conceptos fundamentales de programación orientada a objetos (POO) y comprender la sintaxis básica de Java

Se practicaba la declaración de variables en Java, se asignaba nombres descriptivos y se elegia tipos de datos apropiados, se proporcionaron ejemplos específicos, como la creación de variables para representar nombres, cargos y clasificación. Además, Se exploraban diferentes operadores en Java, desde los aritméticos hasta los relacionales y lógicos.

Se realizaban ejercicios prácticos para consolidar la comprensión de cómo utilizar estos operadores en situaciones concretas.

Tiempo dedicado a esta actividad: ____

Foto 1: Imagen referencial (colocar una propia)

3.1.2 Describir las características clave de Java, como la portabilidad, orientación a objetos, y la gestión automática de memoria

Se realizaba una explicación detallada de la portabilidad de Java y cómo se lograba a través del concepto "Write Once, Run Anywhere" (Escribe una vez, ejecuta en cualquier lugar). Para lo cual en un ejercicio práctico "Hola Mundo" se modificaba y ejecutaba el mismo código Java en diferentes entornos y plataformas.

Tiempo dedicado a esta actividad: ____

Foto 2: Imagen referencial (d colocar una propia)

3.1.3 Crear programas simples en Java siguiendo la estructura básica, incluyendo la declaración de paquetes, importaciones, función principal, y aplicar buenas prácticas de codificación para organizar y comentar programas de manera efectiva.

Se desarrollaba ejemplos prácticos donde se crearon programas simples con funciones main que realizaban tareas específicas, se aplicaban buenas prácticas, donde se elegia nombres de variables descriptivos, el uso adecuado de sangrías y la aplicación consistente de convenciones de codificación.

Tiempo dedicado a esta actividad: ____

Foto 3: Imagen referencial (colocar una propia)

3.1.4 Identificar y utilizar correctamente los tipos de datos primitivos en Java, como enteros, decimales, caracteres y booleanos y selecciona el tipo de dato apropiado según la naturaleza de la información que se manejan en el programa.

Se creaban ejercicios donde se aplicaban los tipos de datos primitivos en Java, incluyendo enteros (int), decimales (double), caracteres (char) y booleanos (boolean), se analizaba sobre la importancia de elegir el tipo de dato adecuado que optimizaba el uso de memoria y garantizaba la precisión de los datos. Durante esta actividad, se demostró como se realizaba la declaración y manipulación efectiva de variables en Java. Se enfocaba en asignar valores, se seguían las reglas de nomenclatura y se comprendieron conceptos clave como el alcance, la inmutabilidad y el uso del modificador final en relación con las variables. A través de ejemplos prácticos, se fortaleció su comprensión y se aplicaban estos conceptos en situaciones del mundo real.

Tiempo dedicado a esta actividad: ____

Foto X: Imagen referencial (colocar una propia)

3.1.5 Utilizar correctamente operadores aritméticos, de asignación, de comparación y lógicos en Java para realizar operaciones matemáticas y manipulación de datos, explica la precedencia de los operadores y utilizar paréntesis para controlar el orden de evaluación.

En esta actividad, se exploraba el correcto uso de diversos operadores en Java, incluyendo aritméticos, de asignación, de comparación y lógicos. Se hizo hincapié en ejercicios donde se aplicaba operaciones matemáticas y manipulación de datos. Además, se aplicaba correctamente la precedencia de los operadores y la importancia de utilizar paréntesis para controlar el orden de evaluación. Con ejercicios prácticos del mundo real, se fortalecieron las habilidades en la manipulación efectiva de datos.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.6 Manipular cadenas de texto, incluyendo la concatenación, extracción de subcadenas y búsqueda de caracteres, utilizar métodos de la clase String para realizar operaciones comunes, como la conversión de mayúsculas a minúsculas y viceversa.

La programación se centraba en técnicas como la concatenación, extracción de subcadenas y búsqueda de caracteres. También se exploraban métodos de la clase String para realizar operaciones comunes, incluyendo la conversión

entre mayúsculas y minúsculas. A través de ejercicios prácticos, se adquirían conocimientos sobre cadena y subcadenas.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.7 Utilizar diferentes métodos para leer datos desde la entrada estándar (teclado) en programas Java, como la clase Scanner o BufferedReader, y maneja excepciones relacionadas con la entrada de datos, como la entrada de datos no válidos.

Se analizaba el manejo efectivo de la entrada de datos desde la entrada estándar (teclado) en programas Java. Se exploraron diferentes métodos, como la clase Scanner y BufferedReader, y se puso énfasis en el manejo de excepciones relacionadas con la entrada de datos, incluyendo situaciones de entrada no válida. A través de casos prácticos se mejoraron las habilidades para leer datos de manera segura.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.8 Utilizar correctamente las sentencias condicionales (if, else if, else) para controlar el flujo de ejecución de un programa, anida sentencias condicionales para manejar situaciones más complejas y comprenderán la importancia de las estructuras de control de flujo en la toma de decisiones en un programa.

Inicialmente se aplicaban las sentencias condicionales en Java para controlar

el flujo de ejecución en el código que se programó. Se abordaron conceptos como if, else if, y else, se tomaba en cuenta el anidamiento de sentencias condicionales para manejar situaciones más complejas. Se analizaba como estas estructuras controlaban el flujo de ejecución según condiciones específicas con el uso adecuado de comparadores y expresiones booleanas.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.9 Implementar bucles (for, while, do-while) para realizar iteraciones y repetir bloques de código, identificar situaciones en las que el uso de bucles es apropiado y selecciona el tipo de bucle más adecuado para una tarea específica.

Se analizaba en qué situación realmente se debería utilizar un bucle for, while, y do-while, se enfatizó la identificación de situaciones apropiadas para el uso, así como la selección del tipo más adecuado según la tarea específica. De acuerdo al código se determinaba cuando se utilizaba bucles for para iteraciones con una cantidad específica de repeticiones, a través de variables de control y expresiones de iteración. Se exploraba cuando usar bucles while para situaciones donde la cantidad de iteraciones no es conocida de antemano. Posteriormente se analizaba bucles do-while y su aplicación en situaciones donde se debería ejecutar el bloque de código al menos una vez.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (debe colocar una propia)

3.1.10 Definir y utilizar clases en Java, comprendiendo la diferencia entre una clase y un objeto, crean (objetos) de una clase, inicializa sus atributos y utiliza métodos asociados, conoce conceptos clave como encapsulamiento, herencia y polimorfismo en el contexto de las clases y los objetos.

En esta sección se enfocaban en definir y utilizar clases, comprender la diferencia entre una clase y un objeto, crear instancias de una clase, inicializar sus atributos y utilizar métodos asociados. Así como se ponía en práctica conceptos clave como encapsulamiento, herencia y polimorfismo, esto resultaba crucial en la creación de instancias de estas clases para representar objetos en el código.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.11 Declarar métodos para modular y organizar código, entender la importancia de los parámetros y el valor de retorno en la creación y llamada de métodos, aplica conceptos como la descarga sobre métodos que contribuyen a la modularidad y reutilización del código.

Se declaraba la aplicación de métodos en Java como una herramienta esencial para modularizar y organizar código. Se exploró la importancia de los parámetros y del valor de retorno en la creación y llamada de métodos, y se aplicaban conceptos como la sobrecarga de métodos que contribuiría a la modularidad y reutilización del código. Se analizaba cómo los métodos ayudan a dividir tareas complejas en partes más manejables.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.12 Aplicar y comprender los modificadores de acceso (públicos, privados, protegidos) para controlar el acceso a los campos y métodos de una clase.

En esta parte se exploraba y aplicaba los modificadores de acceso en Java para controlar la visibilidad y acceso a campos y métodos en una clase. Se enfocaban en los modificadores public, private, y protected, se analizaba cómo estos afectaban la encapsulación y la seguridad en el diseño de clases.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.13 Comprender la diferencia entre sobrecarga y sobreescritura de métodos, y es capaz de implementar ambos conceptos correctamente en sus programas.

Se analizaban los conceptos de sobrecarga y sobreescritura de métodos en el contexto de la programación orientada a objetos en Java. Se estudiaba la diferencia entre estos dos conceptos para tener la capacidad de implementarlos correctamente en cualquier código.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.14 Diseñar y utilizar la herencia en Java, creando superclases y subclases para aprovechar la reutilización del código y la creación de jerarquías de clases.

Se diseñaba y aplicaba la herencia en Java, se aprovechaba la creación de superclases y subclases para potenciar la reutilización del código y construir jerarquías de clases. Mediante líneas del código se lograba entender y poner en práctica cómo la herencia facilitaba la creación de estructuras de código más flexibles y adaptables.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.15 Entender el concepto de polimorfismo y ser capaces de implementarlo a través de la sobre escritura de métodos y la referencia a objetos de clases superiores.

Se realizaba un análisis sobre la comprensión de polimorfismo para lograr implementarlo mediante la sobreescritura de métodos y el uso de referencias a objetos de clases superiores. Se realizaba este análisis que permitía tratar objetos de diferentes clases de manera uniforme, para lograr crear código más flexible y adaptable.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.16 Diseñar y utilizar clases abstractas para proporcionar una base común para clases derivadas, así como entender cómo implementar métodos abstractos.

Se analizaban los conceptos que permitan comprender cómo estas clases proporcionaban una base común para las clases derivadas para así adquirir la habilidad de implementar métodos abstractos. El objetivo fue fomentar una jerarquía de clases más estructurada y flexible.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.17 Organizar su código en paquetes para modularizar y estructurar proyectos de manera eficiente, además de entender cómo importar clases de otros paquetes.

Se analizaba el código con el objetivo principal de modularizar y estructurar el código de manera eficiente, con esto se fomentaba una gestión más clara y sostenible del desarrollo del software. Además, se abordaba los conceptos de importación de clases desde otros paquetes.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.18 Manejar excepciones, utilizando bloques try-catch, y entiende la jerarquía de excepciones en Java para manejarlas de manera efectiva.

Se enfocaba en la utilización de bloques try-catch, se implementaban bloques

para rodear segmentos de código propensos a excepciones además se gestionaba situaciones y se comprendió la jerarquía de excepciones en Java para manejarlas de manera efectiva.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.19 Diseñar interfaces gráficas de usuario (GUI) utilizando la biblioteca Swing de Java.

Se diseñaba Interfaces Gráficas de Usuario (GUI) utilizando la biblioteca Swing de Java. Se procedía con un prediseño o bosquejo previo se centraba en la creación de ventanas, componentes, paneles, se exploraba elementos nuevos como la exploración de la disposición de componentes como BorderLayout y FlowLayout, además de la gestión de eventos para ofrecer experiencias interactivas a los usuarios, la interfaz se realizaba en base a los requerimientos y peticiones de los usuarios, por lo general se diseñaba interfaces fáciles de usar simples y planas.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.20 Comprender el modelo de programación basado en componentes de JSF y desarrollar aplicaciones web Java utilizando JSF, diseña páginas de forma declarativa y gestionar eventos del lado del servidor.

Se desarrollo aplicaciones web utilizando JavaServer Faces (JSF).

Previamente se comprendió el modelo de programación basado en componentes de JSF, se diseñaba páginas de manera declarativa y se gestionaba eventos del lado del servidor, de forma declarativa se utilizaba Facelets, la tecnología de vista en JSF. Finalmente se evaluaba de cómo el modularidad de componentes simplificaba el desarrollo y mantenimiento del código.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.1.21 Conectar a bases de datos utilizando JDBC, realizar consultas SQL desde Java y gestionar transacciones y entiende los conceptos de ResultSet, Statement y Connection en el contexto de JDBC.

Previamente se diseñaba la base de datos que incluía las tablas y las relacione, se exploraba la conectividad a bases de datos desde aplicaciones Java utilizando JDBC (Java Database Connectivity). El enfoque se basaba en realizar consultas SQL, gestionar transacciones y se comprendio los conceptos clave como ResultSet, Statement y Connection en el contexto de JDBC.

Tiempo dedicado a esta actividad: ____

Foto x: Imagen referencial (colocar una propia)

3.2 Recursos

Los recursos utilizados para el desarrollo de las actividades fueron:

- Un equipo con conexión a internet
- IDE NetBeans para desarrollo.

4. Habilidades Desarrolladas

PERSONAL	PROFESIONAL
Durante las actividades de desarrollo de software, se enfrentaron a diversos desafíos, desde errores en el código hasta la resolución de problemas lógicos en consultas SQL. Las habilidades desarrolladas fueron: analizar, identificar y resolver problemas de manera efectiva	Profesionalmente se desarrollaron habilidades para comprender y aplicar conceptos de POO en Java, como encapsulamiento, herencia y polimorfismo, es esencial para el desarrollo de software. Estas habilidades permiten la creación de código modular, reutilizable y fácil de mantener.
Al diseñar interfaces gráficas de usuario con Swing o JSF, se necesitaron habilidades creativas para estructurar y diseñar interfaces intuitivas y atractivas. La capacidad para pensar de manera creativa mejoró la experiencia del usuario.	La experiencia en el manejo de bases de datos con JDBC y el diseño de consultas SQL proporciona habilidades fundamentales para el desarrollo de aplicaciones empresariales. La capacidad para interactuar eficientemente con bases de datos mejora la funcionalidad de las aplicaciones.
Trabajar en proyectos de desarrollo de software a menudo implica la colaboración en equipos por lo que se debe comunicarse de manera efectiva para entender los requisitos, discutir soluciones y compartir conocimientos, demostrando habilidades de colaboración y	La capacidad para diseñar y desarrollar aplicaciones web con tecnologías como JSF muestra habilidades en el desarrollo de soluciones web. Esto implica comprender la arquitectura web, gestionar sesiones y eventos del lado del servidor, y crear interfaces interactivas.

comunicación.	

4.1 Impacto de las prácticas pre profesionales laborales en la sociedad

La programación orientada a objetos (POO) y el manejo de bases de datos tienen un impacto significativo en la sociedad al contribuir a la automatización de procesos empresariales. Las aplicaciones desarrolladas pueden mejorar la eficiencia, reducir errores y acelerar tareas que afectan directamente a la productividad y rentabilidad de las empresas.

_____

_____

_____

_____

5. Marco Teórico

5.1 Programación Orientada a Objetos

6. Conclusiones

- Las actividades enfocadas en programación orientada a objetos (POO), manejo de bases de datos con JDBC, desarrollo de interfaces gráficas y aplicaciones web con JSF proporcionaron a los programadores un conjunto integral de habilidades técnicas. Estas habilidades abarcan desde la lógica de programación hasta la creación de experiencias de usuario efectivas y la gestión eficiente de datos.
- Las actividades realizadas son del mundo real en el desarrollo de software. Desde la conexión a bases de datos empresariales hasta el diseño de interfaces amigables, Esto proporciona una base sólida para enfrentar desafíos en la industria y contribuir a proyectos significativos.
- La resolución de problemas es constante en todas las actividades. Se aprendió a aplicar conceptos teóricos, también a enfrentar y superar desafíos prácticos. Esto empodera para abordar problemas del mundo real en el desarrollo de software, fomentando la confianza en la capacidad para analizar, diseñar e implementar soluciones efectivas.
- En lo profesional, se logró

  _____

  _____

  _____.
- Referente al aprendizaje personal, se reforzó

  I_____

  _____

  _____.

7. Recomendaciones

- A los estudiantes de la carrera de desarrollo de software, recuerden que su tiempo en el Instituto San Gabriel es una oportunidad valiosa para construir una base sólida. Aprovechen al máximo cada experiencia, busquen asesoramiento cuando sea posible y sigan persiguiendo su pasión por el desarrollo de software.
- A la carrera en lo posible vincular a estudiantes a empresas que tengan requerimientos específicos en proyectos java, que posean tutores que guíen y asesoren a los practicantes y que puedan completar proyectos.

8. Referencias Web - Bibliográficas

9. Anexos

9.1 Anexo 1

Hoja de Evaluación

9.2 Anexo 2

Certificado de prácticas

9.3 Anexo 3

Registro de asistencia