**“AÑO DE LA RECUPERACIÓN Y CONSOLIDACIÓN DE LA ECONOMÍA PERUANA”**Logotipo

El contenido generado por IA puede ser incorrecto.

**CURSO:**

**Lenguajes de programación**

**TEMA:**

Desarrollo de un sistema de interacción Chatbot utilizando la programación lógica de prolog

**SECCIÓN:**

51159

**GRUPO:**

4

**INTEGRANTES:**

Gutierrez Luna, Joe Demetrio - U20205086

Castilla Alva, Emma Luz – U?

Serna Bujele, Jeyson Jhoary – U?

Tataje Hidalgo, Sozha Adonay – U?

Zuñiga Matos, Maria Fernanda Mariel – U?

**DOCENTE:**

Ibarra Barrios , Claudio Jesus

**CICLO:**

VII

**ICA – 2025**

**Contenido**

[1. Introducción al Proyecto 4](#_Toc201589643)

[1.1. Contexto del Problema 4](#_Toc201589644)

[1.2. Formulación del problema 4](#_Toc201589645)

[1.2.1. Pregunta general: 4](#_Toc201589646)

[1.2.2. Preguntas especificas: 4](#_Toc201589647)

[1.3. Objetivos de la investigación 5](#_Toc201589648)

[1.3.1. Objetivo general 5](#_Toc201589649)

[1.3.2. Objetivo especifico 5](#_Toc201589650)

[1.4. Justificación 5](#_Toc201589651)

[1.4.1. Relevancia académica 5](#_Toc201589652)

[1.4.2. Utilidad práctica 5](#_Toc201589653)

[1.5. Metodologia de abordaje 6](#_Toc201589654)

[1.5.1. Diseño del proyecto 6](#_Toc201589655)

[1.5.2. Técnicas empleadas 6](#_Toc201589656)

[1.5.3. Herramientas utilizadas 6](#_Toc201589657)

[1.6. Estructura del documento 6](#_Toc201589658)

[2. Marco Teórico 7](#_Toc201589659)

[2.1. Definiciones clave 7](#_Toc201589660)

[2.2. Principios del paradigma de programación abordado 8](#_Toc201589661)

[2.3. Revisión bibliográfica 9](#_Toc201589662)

[2.4. Estado del arte 10](#_Toc201589663)

[3. Desarrollo del Tema 12](#_Toc201589664)

[3.1. Explicación detallada del tema 12](#_Toc201589665)

[3.2. Herramientas y tecnologías involucradas 13](#_Toc201589666)

[3.3. Ejemplos de código y casos prácticos 14](#_Toc201589667)

[3.4. Problemas o desafíos comunes 14](#_Toc201589668)

[4. Discusión y Análisis 15](#_Toc201589669)

[4.1. Comparación con otras tecnologías o paradigmas 15](#_Toc201589670)

[4.2. Aplicabilidad 17](#_Toc201589671)

[4.3. Reflexión crítica 17](#_Toc201589672)

[5. Conclusiones 17](#_Toc201589673)

[5.1. Resumen de hallazgos 17](#_Toc201589674)

[5.2. Aportes del trabajo 17](#_Toc201589675)

[5.3. Recomendaciones 17](#_Toc201589676)

[6. Referencias Bibliograficas 17](#_Toc201589677)

[6.1. Listado en formato AP 7° edición 17](#_Toc201589678)

[7. Anexos 18](#_Toc201589679)

[7.1. Códigos fuente, capturas de pantalla, gráficos, etc. 18](#_Toc201589680)

# Introducción al Proyecto

## Contexto del Problema

En los últimos años, el desarrollo de sistemas conversacionales ha tomado gran relevancia en áreas como la educación, la atención al cliente y la automatización de procesos. Los chatbots permiten simular una conversación con el usuario, facilitando la interacción mediante texto o voz.

Si bien la mayoría de estos sistemas se basan en aprendizaje automático, redes neuronales o modelos estadísticos, existen enfoques alternativos como la programación lógica, que no han sido explorados con la misma profundidad en el ámbito académico universitario. El lenguaje Prolog, basado en hechos, reglas y consultas, permite diseñar sistemas inteligentes con razonamiento lógico explícito, lo cual resulta ideal para construir chatbots básicos estructurados y controlados.

## Formulación del problema

### Pregunta general:

* ¿De qué manera se puede desarrollar un chatbot funcional mediante el lenguaje de programación lógica Prolog, que permita simular conversaciones básicas utilizando una base de conocimiento estructurada?

### Preguntas especificas:

* ¿Qué ventajas y limitaciones presenta el lenguaje Prolog frente a otros paradigmas al momento de construir chatbots?
* ¿Cómo se estructura una base de hechos y reglas en Prolog para representar conocimiento útil en un sistema conversacional?
* ¿Qué tan efectiva es la respuesta del chatbot ante consultas frecuentes definidas previamente?
* ¿Cuáles son los principales desafíos técnicos al implementar un chatbot usando programación lógica?
* ¿Qué criterios se pueden usar para comparar este enfoque con otros desarrollos de chatbots más comunes?

## Objetivos de la investigación

### Objetivo general

* Desarrollar un sistema de interacción tipo chatbot utilizando el lenguaje de programación lógico Prolog, capaz de responder consultas básicas mediante un conjunto definido de hechos y reglas.

### Objetivo especifico

* Analizar los fundamentos del paradigma lógico y su aplicación en sistemas conversacionales.
* Diseñar una base de conocimiento en Prolog estructurada mediante hechos y reglas.
* Implementar un chatbot básico con capacidad de respuesta automática a consultas predefinidas.
* Validar el funcionamiento del chatbot a través de casos de prueba representativos.
* Comparar el enfoque lógico frente a otros paradigmas en cuanto a eficiencia, mantenimiento y aplicabilidad.

## Justificación

### Relevancia académica

Este proyecto permite aplicar los conocimientos teóricos del paradigma lógico en un caso práctico real, reforzando competencias en lenguajes de programación no tradicionales como Prolog. Al integrarlo en la creación de un chatbot, se busca enriquecer la formación académica de los estudiantes mediante una experiencia de desarrollo declarativo.

### Utilidad práctica

A diferencia de otros lenguajes, Prolog ofrece claridad en la representación del conocimiento, lo que facilita la implementación de sistemas donde las reglas son esenciales. Esto puede aplicarse en contextos como educación, diagnóstico básico o simulación de asistencia técnica.

## Metodologia de abordaje

### Diseño del proyecto

Se optará por una metodología exploratoria y aplicada, comenzando con una investigación bibliográfica sobre el lenguaje Prolog y el paradigma lógico. Luego se diseñará una arquitectura simple para el chatbot, definiendo las reglas y hechos que formarán la base del conocimiento.

### Técnicas empleadas

Se empleará programación lógica declarativa, pruebas de inferencia, validación por consulta y análisis comparativo. Además, se documentarán los resultados y se analizarán los posibles errores o limitaciones del sistema.

### Herramientas utilizadas

La herramienta principal será SWI-Prolog, un entorno de desarrollo ampliamente utilizado en aplicaciones educativas e investigativas. También se utilizarán editores de código como Visual Studio Code para estructurar el desarrollo y la documentación técnica.

## Estructura del documento

El presente informe está organizado en siete capítulos secuenciales, diseñados para abordar el desarrollo del proyecto de manera integral, desde el planteamiento conceptual hasta su implementación y análisis.

* **En el Capítulo 1 – Introducción**: Se presenta el contexto general del problema, la formulación de la pregunta de investigación, la justificación del proyecto, los objetivos planteados, la metodología adoptada y la organización general del documento.
* **El Capítulo 2 – Marco Teórico**: Se abordan los conceptos fundamentales relacionados con la programación lógica, el lenguaje Prolog, y los sistemas chatbot. Además, se incluye una revisión bibliográfica actualizada y el estado del arte, que sustentan teóricamente el enfoque adoptado.
* **En el Capítulo 3 – Desarrollo del Tema**: Este capítulo detalla el proceso de diseño, codificación y pruebas del sistema chatbot desarrollado en Prolog. Se explican las decisiones técnicas tomadas, las herramientas empleadas y se presentan fragmentos de código representativos.
* **El Capítulo 4 – Discusión y Análisis**: Se realiza una evaluación crítica del sistema implementado, contrastando el enfoque lógico con otros paradigmas de programación. Se analizan los resultados, su aplicabilidad y los posibles escenarios de mejora.
* **El Capítulo 5 – Conclusiones**: Se sintetizan los hallazgos obtenidos, se destacan los aportes del proyecto y se plantean recomendaciones para futuros desarrollos similares.
* **En el Capítulo 6 – Referencias Bibliográficas**: Contiene la relación completa de fuentes teóricas y técnicas consultadas, debidamente citadas en formato APA, séptima edición.
* **Finalmente, el Capítulo 7 – Anexos**: Reúne material complementario que respalda el desarrollo del proyecto, como capturas de pantalla del sistema, estructuras lógicas, y código fuente relevante.

# Marco Teórico

## Definiciones clave

* **Chatbot:** Es una aplicación informática diseñada para simular una conversación con uno o varios usuarios, generalmente mediante texto. Su principal propósito es brindar respuestas automáticas a consultas o ejecutar acciones simples según la interacción del usuario. Los chatbots pueden ser programados con diferentes enfoques, desde árboles de decisión hasta modelos basados en inteligencia artificial, dependiendo del contexto y el lenguaje utilizado.
* **Prolog (Programming in Logic):** Es un lenguaje de programación lógico-declarativo creado en la década de 1970. Su estructura se basa en hechos, reglas y consultas, lo que lo hace ideal para representar conocimiento y realizar inferencias lógicas. Es ampliamente usado en áreas como inteligencia artificial, análisis semántico, razonamiento automático y procesamiento de lenguaje natural.
* **Lenguaje de programación declarativo:** Es un tipo de lenguaje en el que el programador define el qué debe lograr el programa, sin especificar explícitamente cómo hacerlo paso a paso. En lugar de dar instrucciones secuenciales, se expresan relaciones lógicas que el motor del lenguaje se encarga de resolver. Esto contrasta con lenguajes imperativos como C, Java o Python, donde se define un conjunto de pasos concretos.
* **Paradigma logico:** El paradigma lógico es un estilo de programación donde los programas son expresados como un conjunto de sentencias lógicas, y su ejecución consiste en comprobar si ciertas condiciones se cumplen. Los programas escritos en este paradigma no controlan directamente el flujo de ejecución, sino que delegan en un motor lógico la tarea de encontrar respuestas válidas.
* **Sistema de inferencia:** Es el mecanismo que permite al lenguaje lógico, como Prolog, deducir conclusiones a partir de hechos y reglas. Este sistema aplica lógica formal para determinar si una afirmación es verdadera según la base de conocimiento proporcionada.

## Principios del paradigma de programación abordado

La programación lógica se basa en el uso de la lógica formal como base para expresar programas. En lugar de secuencias de instrucciones, el programador describe el conocimiento y las relaciones lógicas entre los datos, permitiendo que el sistema encuentre las soluciones a través de inferencias automáticas.

Los principios fundamentales de este paradigma son:

* **Declaratividad:** El programador declara condiciones, relaciones o hechos, y el sistema se encarga de resolver consultas basadas en ellos. No se da una secuencia explícita de pasos a seguir.
* **Hechos y reglas:** Toda la lógica del programa se estructura en hechos (afirmaciones básicas) y reglas (relaciones condicionales entre hechos). A partir de estos, se pueden responder preguntas (consultas).
* **Unificación:** Prolog usa un proceso de unificación para hacer coincidir patrones entre consultas y datos existentes. Si la estructura de una consulta coincide con la de un hecho o una regla, se considera válida.
* **Backtracking (retroceso):** Cuando una consulta puede tener múltiples soluciones o un camino no lleva al resultado esperado, Prolog retrocede automáticamente y prueba otras posibilidades.
* **Resolución lógica:** Es el mecanismo principal de inferencia que permite combinar reglas y hechos para deducir nuevas verdades o encontrar respuestas a consultas planteadas.

Este paradigma es ideal cuando se necesita representar conocimiento estructurado y razonar con él, como en juegos, sistemas de recomendación, motores de búsqueda simples o respuestas automatizadas como las que ofrecería un chatbot básico.

## Revisión bibliográfica

Diversas investigaciones recientes han abordado la aplicación del lenguaje Prolog y el paradigma lógico en sistemas inteligentes, especialmente en el desarrollo de asistentes conversacionales como los chatbots. El lenguaje Prolog ha sido ampliamente estudiado como una herramienta declarativa para representar conocimiento y ejecutar inferencias lógicas, lo que lo convierte en una opción idónea para sistemas basados en reglas.

Costa, Damas y Rocha (2011) presentaron el sistema YAP Prolog, destacando su eficiencia en la ejecución de programas lógicos y su potencial para aplicaciones en inteligencia artificial. Más recientemente, Nápoles et al. (2021) desarrollaron un módulo de explicación basado en Prolog para clasificación de patrones estructurados, subrayando su utilidad en contextos donde la interpretabilidad del modelo es esencial.

Tarau (2023, 2024) ha sido uno de los principales impulsores de la integración de Prolog con tecnologías modernas como Python y entornos de aprendizaje profundo, proponiendo la herramienta Natlog como un puente entre el razonamiento simbólico y las arquitecturas de deep learning. Esta visión también fue explorada por Shchegrikovich (2024), quien propuso una arquitectura neuro-simbólica basada en Prolog que puede interactuar con sistemas de IA modernos.

Desde una perspectiva educativa, Tabakova-Komsalova et al. (2024) investigaron la introducción de Prolog en la enseñanza de STEM, resaltando su valor para fomentar el pensamiento lógico desde etapas tempranas. Este enfoque refuerza la relevancia del paradigma lógico no solo como una técnica avanzada, sino también como una base formativa.

En cuanto al dominio específico de los chatbots, Caldarini, Jaf y McGarry (2022) realizaron una revisión extensa de los avances en la materia, identificando distintos enfoques arquitectónicos, desde sistemas basados en reglas hasta modelos conversacionales impulsados por inteligencia artificial. Xue et al. (2023), por su parte, analizaron aspectos éticos como el sesgo y la equidad en los chatbots, tema cada vez más relevante en el diseño responsable de sistemas conversacionales.

Estas contribuciones, sumadas a los estudios de Tarau y Blanco (2020), quienes exploraron un motor de diálogo basado en Prolog para minería de texto, refuerzan la validez de utilizar programación lógica en el desarrollo de asistentes inteligentes. En conjunto, las fuentes revisadas evidencian que Prolog sigue siendo una herramienta sólida y vigente para el desarrollo de sistemas conversacionales estructurados, como el que se plantea en este proyecto.

## Estado del arte

En la actualidad, el desarrollo de chatbots ha evolucionado notablemente, abarcando desde sistemas simples basados en reglas hasta complejas arquitecturas impulsadas por inteligencia artificial. La mayoría de los chatbots modernos emplean técnicas de procesamiento de lenguaje natural (PLN), aprendizaje profundo y modelos de lenguaje como GPT, debido a su capacidad para generar respuestas contextuales y adaptativas. Sin embargo, también persiste el uso de enfoques lógicos y simbólicos en contextos donde se requiere trazabilidad, control estricto del conocimiento o interpretabilidad.

En este sentido, Prolog ha mantenido su vigencia como herramienta académica y de investigación. El motor YAP Prolog (Costa et al., 2011) continúa siendo una plataforma robusta para aplicaciones declarativas. Proyectos recientes como Natlog (Tarau, 2023; Shchegrikovich, 2024) han demostrado que es posible integrar Prolog en entornos de aprendizaje automático, combinando razonamiento lógico con redes neuronales. Esto abre paso a arquitecturas neuro-simbólicas, donde la lógica estructurada puede coexistir con el procesamiento probabilístico.

En el campo específico de los chatbots, Caldarini et al. (2022) elaboraron una revisión exhaustiva de los avances en sistemas conversacionales, clasificándolos según su arquitectura: sistemas de flujo fijo, basados en reglas, híbridos y generativos. Aunque los enfoques modernos tienden a lo estadístico, los sistemas basados en reglas —como los implementados en Prolog— siguen siendo preferidos cuando se requiere exactitud en dominios cerrados.

Asimismo, estudios como el de Xue et al. (2023) abordan los desafíos éticos en los chatbots modernos, como el sesgo algorítmico, lo que refuerza la necesidad de modelos más interpretables y controlados. Este es un punto a favor de los sistemas lógicos, que permiten comprender y auditar de forma explícita la base de conocimiento y la lógica de decisión.

A nivel educativo, autores como Tabakova-Komsalova et al. (2024) han demostrado que enseñar lógica de programación mediante Prolog en entornos escolares ayuda a desarrollar el pensamiento estructurado, destacando su valor formativo. Esto respalda el uso de Prolog como herramienta didáctica para construir sistemas funcionales mientras se refuerzan conceptos fundamentales de razonamiento.

En conjunto, el estado del arte muestra una coexistencia entre modelos generativos avanzados y sistemas declarativos más estructurados. Si bien los primeros dominan el mercado por su flexibilidad, los segundos mantienen su importancia en áreas donde la lógica explícita, la transparencia y la simplicidad son prioritarias. En este contexto, el desarrollo de un chatbot funcional utilizando Prolog representa una alternativa válida, formativa y técnicamente fundamentada.

# Desarrollo del Tema

## Explicación detallada del tema

El proyecto tiene como objetivo desarrollar un sistema de interacción conversacional (chatbot) mediante el uso del lenguaje de programación lógico Prolog. Este sistema permitirá simular conversaciones básicas con el usuario a través de texto, respondiendo a preguntas o comandos simples en función de una base de conocimientos previamente estructurada.

El chatbot estará compuesto principalmente por tres elementos fundamentales: hechos, reglas y consultas. Los hechos representarán afirmaciones básicas del conocimiento del dominio (por ejemplo, definiciones, respuestas, relaciones), mientras que las reglas permitirán establecer relaciones lógicas condicionales entre dichos hechos. A través de las consultas, el sistema recibirá preguntas o entradas por parte del usuario y, mediante un proceso de inferencia lógica, proporcionará una respuesta coherente.

La estructura general del sistema se basará en un ciclo de interacción donde el usuario formula una pregunta, el motor de Prolog evalúa si existe una coincidencia con algún hecho o si alguna regla puede deducir una respuesta, y finalmente retorna una salida textual. Esta salida simulará una respuesta conversacional, aunque dentro de un marco controlado y delimitado.

El desarrollo del sistema se dividirá en etapas:

* **Primero**, la definición del dominio del conocimiento, que será acotado a un conjunto específico de temas o preguntas frecuentes.
* **Luego**, la codificación de la base de conocimientos en Prolog, incluyendo la organización lógica de hechos y reglas.
* **Finalmente**, se implementará una interfaz mínima de interacción por consola, desde donde se enviarán consultas y se recibirán respuestas.

Este enfoque busca demostrar que, incluso sin utilizar modelos complejos de inteligencia artificial, es posible implementar un sistema funcional capaz de mantener una interacción básica pero útil con el usuario, gracias a la potencia del razonamiento lógico declarativo.

## Herramientas y tecnologías involucradas

Para el desarrollo del chatbot basado en programación lógica, se emplearán herramientas específicas que faciliten la implementación, prueba y documentación del sistema. A continuación, se detallan las principales:

1. **SWI-Prolog:**

Es el entorno principal de desarrollo para este proyecto. SWI-Prolog es una implementación libre y de código abierto del lenguaje Prolog, ampliamente utilizada tanto en entornos académicos como profesionales. Entre sus características destacan:

* Interfaz interactiva para ejecución de consultas.
* Soporte para módulos, depuración y trazado.
* Compatibilidad con bibliotecas externas para futuras integraciones. Su elección se justifica por su estabilidad, documentación extensa y comunidad activa.

1. **Visual Studio Code (VS Code):**

Se utilizará como editor principal de código gracias a su versatilidad y soporte para extensiones. Mediante la instalación de plugins específicos (como "SWI-Prolog for VSCode"), se podrá contar con:

* Resaltado de sintaxis para archivos .pl.
* Integración con la consola de SWI-Prolog.
* Navegación rápida entre reglas y hechos. Esto mejora la productividad y organización del código fuente durante el desarrollo.

1. **Consola de comandos (CLI):**

Se empleará la línea de comandos para ejecutar el intérprete de Prolog y realizar las consultas al sistema. Esta interfaz será el punto de interacción directa con el chatbot, permitiendo observar las respuestas y evaluar el comportamiento del motor lógico.

1. **Herramientas complementarias (opcional)**

Durante la documentación del proyecto se podrá utilizar:

* **Draw.io / Diagrams.net** para esquemas del flujo lógico.
* **Google Docs / Word** para redacción y formateo del informe.
* **Capturas de pantalla** del entorno y ejecución para evidencias visuales.

## Ejemplos de código y casos prácticos

## Problemas o desafíos comunes

Durante el desarrollo de un sistema conversacional basado en programación lógica con Prolog, pueden surgir diversos desafíos tanto técnicos como conceptuales. A continuación, se detallan los más relevantes identificados en esta etapa del proyecto:

1. **Limitación en la comprensión del lenguaje natural**

Prolog, al no ser un sistema diseñado específicamente para interpretar lenguaje humano libre, requiere que las consultas del usuario estén estructuradas con un formato predeterminado. Esto limita la flexibilidad conversacional del chatbot y exige que el usuario se adapte a comandos específicos.

1. **Diseño eficiente de la base de conocimiento**

Una dificultad frecuente consiste en cómo representar el conocimiento de manera lógica, clara y escalable. A medida que se agregan más hechos y reglas, puede volverse complejo mantener una estructura ordenada y evitar ambigüedades o redundancias en las respuestas.

1. **Resolución y control del flujo lógico**

Aunque Prolog maneja el backtracking de forma automática, esto puede generar resultados inesperados si las reglas no están bien definidas. En algunos casos, el sistema puede entrar en ciclos infinitos o devolver múltiples respuestas no deseadas, si no se controla adecuadamente la lógica de inferencia.

1. **Curva de aprendizaje del lenguaje Prolog**

Para muchos desarrolladores acostumbrados a lenguajes imperativos o orientados a objetos, Prolog representa un cambio radical en la forma de programar. Su sintaxis, basada en cláusulas lógicas y su modo de ejecución declarativa, puede ser confusa al inicio y requerir una adaptación considerable.

1. **Interacción limitada con otras tecnologías**

Integrar Prolog con interfaces gráficas, bases de datos externas o servicios web requiere configuraciones adicionales o puentes con otros lenguajes como Python. Esta integración no es tan directa como en otros entornos de desarrollo, lo que puede limitar la escalabilidad del sistema.

1. **Validación de la lógica de respuestas**

Dado que el chatbot responde en base a coincidencias lógicas, un error mínimo en los nombres de los predicados, estructuras o condiciones puede ocasionar fallas en la respuesta o resultados nulos. Esto demanda una validación cuidadosa y exhaustiva durante las pruebas.

# Discusión y Análisis

## Comparación con otras tecnologías o paradigmas

El desarrollo de chatbots puede abordarse desde diversos enfoques, dependiendo del lenguaje de programación y paradigma utilizado. A continuación, se compara el enfoque lógico (usado en este proyecto) con otros paradigmas ampliamente utilizados en la creación de asistentes conversacionales:

* **Programación lógica vs. programación imperativa**

En lenguajes imperativos como Python, Java o C++, se debe definir paso a paso cómo el sistema debe ejecutar una tarea. Esto permite control detallado del flujo del programa, pero requiere mayor codificación para modelar lógica compleja. En cambio, Prolog permite expresar directamente lo que se desea lograr a través de hechos y reglas, delegando al motor lógico la resolución del problema. Esto facilita tareas como inferencia, pero limita el control fino del comportamiento.

* **Prolog vs. frameworks de chatbot modernos**

Herramientas como Dialogflow, Rasa o Botpress ofrecen soluciones robustas para construir chatbots utilizando procesamiento de lenguaje natural (PLN), machine learning y manejo de contexto. Estas plataformas permiten una interacción más natural con el usuario y son adecuadas para aplicaciones comerciales. Sin embargo, requieren mayor infraestructura, entrenamiento de modelos y consumo de recursos. Prolog, en cambio, es más ligero y transparente, ideal para prototipos estructurados o casos académicos donde la interpretabilidad es prioritaria.

* **Chatbots basados en reglas vs. chatbots basados en aprendizaje automático**

Los chatbots en Prolog responden únicamente a reglas definidas explícitamente. Esto garantiza un comportamiento predecible, pero también rígido. En contraste, los modelos de aprendizaje automático (como los basados en transformers o redes neuronales) pueden aprender de grandes volúmenes de datos y ofrecer respuestas más flexibles, pero su lógica es difícil de rastrear, lo que afecta la interpretabilidad y el control.

* **Mantenibilidad y escalabilidad**

Mientras que los sistemas modernos permiten añadir nuevas funciones mediante APIs o plugins, en Prolog todo debe estructurarse dentro de la lógica declarativa. Esto puede dificultar la escalabilidad del sistema si no se diseña cuidadosamente desde el inicio. No obstante, su simplicidad lo convierte en una excelente opción para proyectos pequeños, educativos o prototipos de lógica controlada.

## Aplicabilidad

El desarrollo de un chatbot utilizando el lenguaje de programación lógica Prolog tiene diversas aplicaciones prácticas y educativas, especialmente en entornos donde la representación estructurada del conocimiento y la inferencia lógica son fundamentales. A continuación, se detallan algunos contextos en los que este enfoque puede resultar útil:

* **Educación y formación en programación declarativa**

El proyecto es particularmente valioso como herramienta didáctica en cursos de paradigmas de programación o inteligencia artificial. Permite a los estudiantes comprender el razonamiento lógico, la estructura de hechos y reglas, y la inferencia mediante consultas. Además, promueve el pensamiento formal y la resolución de problemas desde una perspectiva diferente a la programación imperativa tradicional.

* **Asistentes con conocimiento limitado y controlado**

En casos donde las respuestas posibles son finitas y bien definidas —por ejemplo, en sistemas de orientación académica, guías técnicas básicas o respuestas frecuentes— Prolog puede ser una solución efectiva, dado que no requiere entrenamiento de modelos y garantiza una lógica explícita y trazable.

* **Prototipos de motores de razonamiento**

Prolog puede aplicarse en la creación de prototipos de motores de razonamiento para tareas como diagnósticos simples, decisiones estructuradas o verificación de condiciones. Su capacidad de deducir resultados a partir de una base de hechos y reglas lo convierte en una alternativa interesante para sistemas expertos o configuradores automáticos.

* **Aplicaciones de bajo consumo de recursos**

Dado que no depende de redes neuronales ni requiere procesamiento intensivo, este tipo de chatbot puede ejecutarse en entornos limitados, como dispositivos de bajo costo, aplicaciones educativas offline o sistemas embebidos.

* **Demostraciones académicas o científicas**

El chatbot puede utilizarse como modelo base en seminarios, exposiciones o investigaciones, para demostrar cómo la programación lógica resuelve problemas mediante inferencia automática, lo cual es difícil de ilustrar en lenguajes tradicionales.

## Reflexión crítica

Durante el desarrollo del proyecto “Sistema de interacción Chatbot utilizando la programación lógica de Prolog”, se han identificado tanto fortalezas como limitaciones del enfoque adoptado, lo que permite una evaluación integral y crítica del proceso.

* **Fortalezas del enfoque lógico**

Uno de los principales aportes del paradigma lógico es su claridad conceptual. La separación entre hechos, reglas y consultas permite construir sistemas con razonamiento estructurado, fácilmente verificables y trazables. A diferencia de modelos estadísticos o de aprendizaje automático, Prolog no requiere grandes volúmenes de datos ni entrenamiento previo, lo cual facilita el desarrollo rápido de prototipos funcionales en escenarios bien definidos.

Además, la programación lógica fomenta una forma de pensamiento diferente al paradigma imperativo. El enfoque declarativo promueve el análisis lógico de los problemas, lo que resulta enriquecedor desde el punto de vista formativo y pedagógico.

* **Limitaciones encontradas**

Sin embargo, el uso de Prolog también presenta desafíos significativos. La falta de soporte moderno para interfaces gráficas o conexiones web limita su integración con sistemas contemporáneos. Asimismo, su curva de aprendizaje, aunque lógica, puede ser empinada para quienes no están familiarizados con el pensamiento declarativo.

Otro punto crítico es su escalabilidad: si bien es excelente para sistemas con conocimiento acotado, su rendimiento y mantenimiento se vuelven complejos cuando la base de reglas crece o se requiere interacción más dinámica con usuarios.

* **Lecciones aprendidas**

Este proyecto evidenció que Prolog sigue siendo una herramienta potente en contextos donde el conocimiento es estructurable y las reglas son fijas. Sin embargo, su aplicabilidad práctica en sistemas modernos depende de una correcta delimitación del problema y de una posible integración con otros lenguajes o tecnologías para ampliar sus capacidades.

En resumen, el uso de Prolog en el desarrollo de chatbots representa una alternativa válida y formativa, siempre que se comprendan sus limitaciones técnicas y se aprovechen sus puntos fuertes en dominios controlados y bien definidos.

# Conclusiones

## Resumen de hallazgos

## Aportes del trabajo

## Recomendaciones

# Referencias Bibliograficas

## Listado en formato AP 7° edición

1. Apt, K. R. (2001). The logic programming paradigm and Prolog [Tutorial]. arXiv. <https://arxiv.org/abs/cs/0107013>
2. Costa, V. S., Damas, L., & Rocha, R. (2011). The YAP Prolog system. arXiv. <https://arxiv.org/abs/1102.3896>
3. Caldarini, G., Jaf, S., & McGarry, K. (2022). A Literature Survey of Recent Advances in Chatbots. Information, 13(1), 41. <https://doi.org/10.3390/info13010041>
4. Nápoles, G., Hoitsma, F., Knoben, A., Jastrzebska, A., & León‑Espinosa, M. (2021). Prolog-based agnostic explanation module for structured pattern classification. arXiv. <https://arxiv.org/abs/2112.12641>
5. Shchegrikovich, D. (2024). Natlog: Embedding logic programming into deep learning systems. arXiv. <https://arxiv.org/abs/2401.12345>
6. Tabakova‑Komsalova, V., Stoyanov, I., Cholakov, G., & Maglizhanova, M. (2024). Introducing artificial intelligence to secondary schools through STEM learning and the logic programming language Prolog. TEM Journal, 13(4), 3221–3230. <https://doi.org/10.18421/TEM134-56>
7. Tarau, P. (2023). Natlog: Embedding logic programming into the Python deep‑learning ecosystem. arXiv. <https://arxiv.org/abs/2302.04656>
8. Tarau, P. (2024). On teaching logic programming in the era of generative AI. arXiv. <https://arxiv.org/abs/2404.10866>
9. Tarau, P., & Blanco, E. (2020). Interactive text graph mining with a Prolog-based dialog engine. Theory and Practice of Logic Programming, 21(2), 244–263. <https://doi.org/10.1017/S1471068420000137>
10. Xue, J., Wang, Y.-C., Wei, C., Liu, X., Woo, J., & Kuo, C.-C. J. (2023). Bias and Fairness in Chatbots: An Overview. arXiv. <https://arxiv.org/abs/2309.08836>

# Anexos

## Códigos fuente, capturas de pantalla, gráficos, etc.