University of Asia Pacific

Department of Computer Science & Engineering

Artificial Intelligence and Expert Systems Lab (CSE - 404)

Title **:** Library Book Recommendation System using Prolog

Date of Submission **:** 02 August 2025

>>Submitted by :

Name : **Shanjida Islam**

Student ID : **22101138**

Section : C-2

Year-Semester : 4th year – 1st semester

Session : Spring 2025

>>Submitted to:

Bidita Sarkar Diba

Lecturer, Department of Computer Science and Engineering, University of Asia Pacific.

Table of Contents

Library Book Recommendation System using Prolog

Problem Description:

In libraries, people often don't know what book to pick, especially if they're not familiar with all the options. This system is made to solve that problem. It's a simple book recommendation tool built using Prolog. It suggests books based on what kind of stories a person likes (like fantasy or mystery) and how experienced they are as a reader (beginner, intermediate, or advanced). It can also go deeper and find books from related subgenres, kind of like giving extra suggestions if it finds something similar to what the person already likes.

Tools and Languages Used:

- **Language: Prolog**

The project was developed using Prolog, a logic programming language ideal for building rule-based knowledge systems.

- **Interpreter: SWI-Prolog**

SWI-Prolog was used to run and test the knowledge base. It provides a powerful and user-friendly environment for executing Prolog programs.

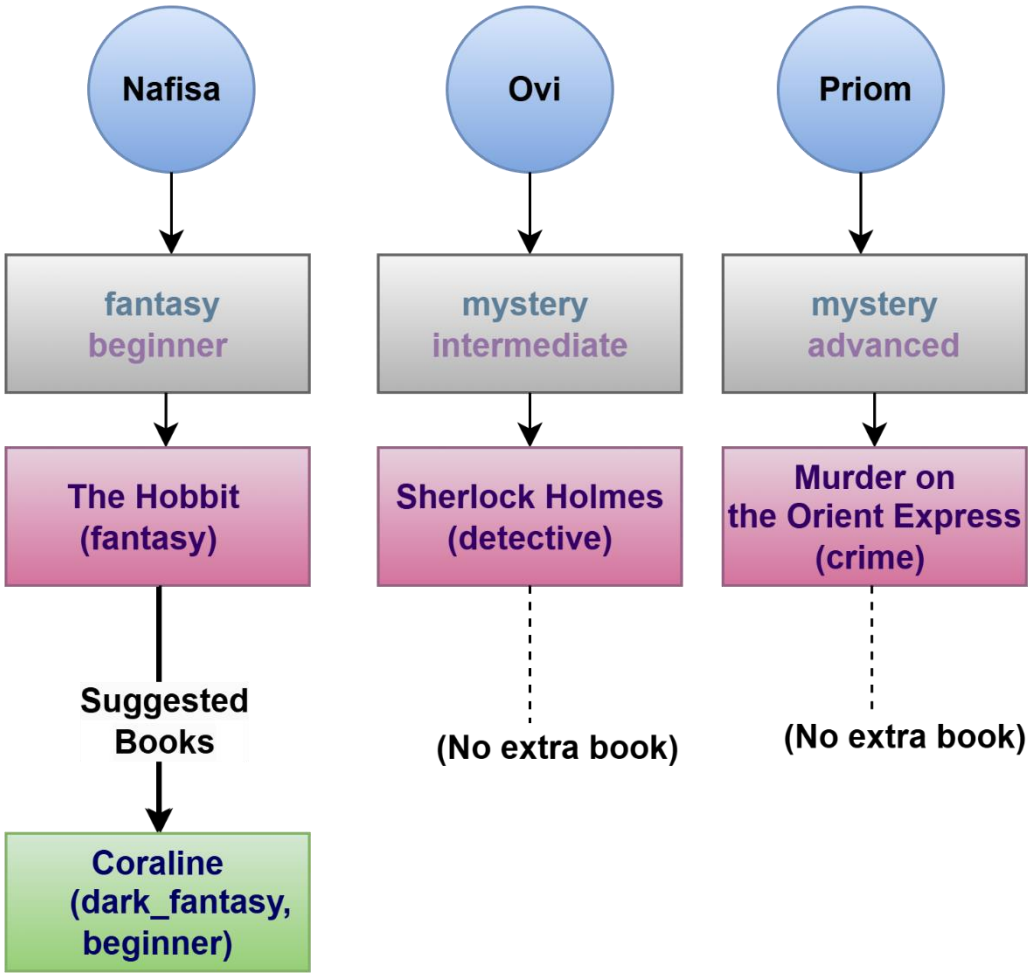
- **Editor: Notepad**

The source code (`.pl` file) was written and saved using Notepad, a simple yet effective text editor for quick development.

- **Diagram Tool**: **draw.io**

The structure of the knowledge base, including user preferences, genres, subgenres, and recommendation flow, was illustrated using draw.io. This visual tool made it easy to design and present the recursive relationships between genres and books.

Diagram/Figure



This diagram shows how books are recommended based on each person's favorite genre and reading level.

Nafisa likes fantasy and is a beginner. The system first recommends The Hobbit based on a direct match. Then, by exploring subgenres like dark fantasy, it also suggests Coraline. So, she gets an extra suggested book through recursion.

Ovi prefers mystery and is at an intermediate level. He is matched with Sherlock Holmes, which is a detective story and a subgenre of mystery. Since there are no deeper matching subgenres, no extra book is suggested.

Priom also likes mystery but is at an advanced level. He gets Murder on the Orient Express, which is in the crime subgenre of mystery. Like Ovi, no further books are suggested.

This diagram helps visualize how Prolog explores genres and subgenres to find personalized book matches.

Sample Input/output

```
?-  recommend_recursive(nafisa, Book).
Book = 'The Hobbit' ;
Book = 'Coraline' .

?- recommend_recursive(ovi, Book).
Book = 'Sherlock Holmes' .

?-  recommend_recursive(priom, Book).
Book = 'Murder on the Orient Express'
```

When we run the recommend_recursive(User, Book) query, the system looks at what genre the user likes and their level.
It also goes deeper to check related subgenres using recursion.

- For Nafisa, who likes fantasy and is a beginner, the system finds The Hobbit directly. Then it also finds Coraline, because it's part of dark fantasy, which is a subgenre of fantasy.
- For Ovi, who likes mystery and is intermediate, it returns Sherlock Holmes, which falls under detective, a subgenre of mystery.
- For Priom, who likes mystery and is advanced, it gives Murder on the Orient Express, which is from the crime subgenre.

This output shows that the recursive rule successfully handles both direct and subgenre-based recommendations.

Conclusion and Challenges:

In this project, I built a knowledge base in Prolog that recommends books based on a person's preferred genre and reading level. The system uses both direct matching and recursion to explore subgenres, which makes the recommendations smarter and more personalized.

Through this, I learned how Prolog handles logic, facts, rules, and recursive queries. It was interesting to see how a small set of facts and rules can lead to useful results just through logical reasoning.

Challenges:

- Making sure the recursion works properly without looping or missing any matches.
- Designing the genre and subgenre relationships in a way that makes sense.
- Debugging when Prolog throws errors like “unknown predicate” was tricky at first.

Overall, it was a good learning experience in using logic programming for real-world-style applications like recommendation systems.

GitHub Repository Link: https://github.com/Shanjida-Islam138/Library-Book-Recommendation-System-using-Prolog.git