Expert System : JEE Branch and College Predictor

Software Requirement Specification

Submitted by : Group 9

Naman Jain	**Parth Jain**	**Ashutosh Kaushik**	**Pratham Prakash Gupta**
ISM2017001	ICM2017006	ICM2017002	IHM2017003
Jai Khurana	**Ayush Shukla**	**Siddharth Gupta**	**Animesh Jha**
ISM2017006	IIT2017106	IRM2017002	ICM2017001

INTRODUCTION

Purpose

The purpose of Software Requirements Specification (SRS) document is to provide the complete description about what can be expected from the software. The software in our case is an expert system concerning JEE branch and college predictor. It lists the functional and non-functional requirements. It also provides the details about the expected schedule and the user interface.

Project Scope and Overview

The aim of this expert system is to aid lakhs of aspiring students that sit for one of the most popular exams of India : JEE Mains. The users would be able to see the best colleges and branches on the basis of their JEE ranks. They will have to input their JEE Ranks, category and home state and they will get a list of the most suitable options. Our expert system will save a lot of time and efforts of the users that they would have wasted had they manually scanned the statistics of various colleges. Our expert system deploys various levels of comparisons with different priorities to come to a very realistics list of colleges and branches.

In a nutshell, JEE Branch and College predictor will be able to do the following tasks :

1. Prompt the user to provide necessary yet minimal information like JEE rank, category and home state.
2. Predict a list of most suitable branches of suitable colleges along with the opening and closing ranks.

COMPLETE OVERVIEW

Basically the Expert system consists of three parts

1. User Interface.
   a. Taking rank, caste and home state as input from the user.
   b. List of colleges based on the input with corresponding starting and closing rank.

2. Knowledge base

3. Inference Engine

4. Database

1. Knowledge base

This will contain the rules which will then use the database facts to infer the result
These rules are based on the priorities given by the user and the attributes
colleges which are present in our database.

For Example (in prolog) :

IsBranchAvailable:-
%Code to check the availability of the branch in the college.

IsDeemedUniversity:-
%Code to check the availability of the branch in the college.

2. Inference engine

The inference engine is the part which uses the rules and the known facts (Knowledge base) to infer things. Here it will use the facts and derived facts about colleges to compare and decide the right college to be shown to the user.

For Example(in prolog) :

CompareAccordingToRank :-
%Code to compare college on the basis of ranks using database facts.

CompareAcordingToBranch :-
%Code to compare college on the basis of branch using database facts.

3. Database

It contains all the attributes of all the colleges that are required to allot the college to particular student according to his priorities.

For Example :

%CollegeName

%Opening Rank
%Closing Rank
%Category
%College state
%NIRF ranking of college
%Branch priority (based on previous trends)

The document will contain all the functional and non-functional requirements.It explains the purpose and features of the expert system, and the requirements so that when it is deployed on a large user base then it could be scaled easily thus solving the problem to the maximum extent. Each specification has been discussed in detail regarding this Expert System.

NON-FUNCTIONAL REQUIREMENTS

1. Performance

1. The database must be normalised and free of redundancy
2. The system should generate a response as early as possible.
3. The expert system must be able to notify the user whenever he/she provides an invalid input or information in an inappropriate format.

2. Reliability and Consistency

1. The input provided by the user must be preserved as long as his/her session in active.
2. The mean time between failures(MTBF) must be long.
3. Same output should be produced when the same input is provided to the expert system.

3. Maintainability

1. The expert system must remain relevant over the years. After every year when new seat allotment statistics are there, they should be easily incorporated with the existing database.
2. If new and better rankings appear, the system administrator must be in a position to add them to allow a better prediction.
3. A well maintained documentation should be there so that a new software engineer can make needed changes.

EXPERT SYSTEM DEVELOPMENT CYCLE

1. Identification of the problem

More than a million students appear for JEE Mains each year and most of them join engineering colleges. After the results, the students become anxious to find out about the best institutes available to them; and for that, they use several college predictors.
But these predictors can often be misleading and might not even present all our best options.
Many of the free predictors are funded by a set of specific colleges and are biased towards them; remember, if it is free, then you are the product.

Most predictors sort the colleges on the basis of their rankings, overlooking the fact that those rankings skip many factors while judging the colleges that could matter highly to the students
As a result of this, many students get stuck in mediocre colleges, which leads to a negative slope in their careers. To help those young pre-engineers, we have designed a college predictor which is based on an expert system. It gives experts' suggestions on the basis of the students' rank, category preferred branches, and state; providing the students with the most competent set of colleges to choose from.

2. Decisions about the mode of development

We would be using Ubuntu operating system because of its security and support.The language used will be python because of its great library ecosystem, flexibility and readability.

1) **Tinker** is used to build GUI applications in Python.

2) **Experta** introduces a form of logical programming in python. Experta integrates with Python allowing you to invoke Experta from Python and intermingle Python statements and expressions within your expert system rules.

3) **CSVReader** is used to read the CSV file of our database.

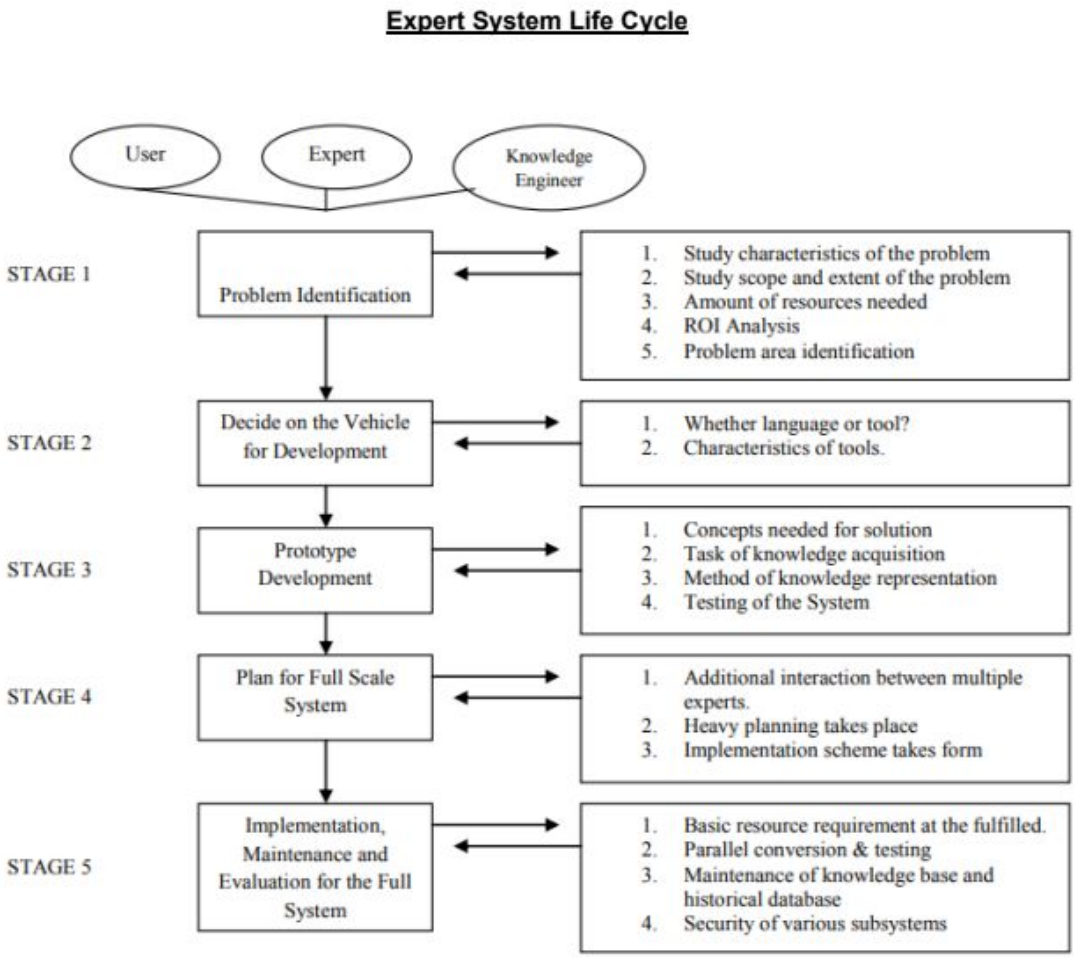

Figure: Expert System Life Cycle

3. Development of the prototype

The Prototype which we will develop will have a GUI with input fields as Rank, Caste and College preference (NIT and IIIT). The user will need to enter his rank, then the category to which he/she belongs. He/she will be shown all the possible options on the basis of the rank and category. The user can also filter the colleges from all the available options by entering the preference of his type of college like NIT and IIIT. The dataset which we would be using would include the cut-off ranks of the colleges for the past 3-4 years. The options would be shown to the user based on this dataset.

The prototype is tested because of which the knowledge of fine granularity emerges which makes the system more efficient.

4. Planning for a full scale system

Once a prototype has been constructed, developing the system into a full-fledged one is truly a challenge. While the prototype intends to produce outputs for a limited range of cases, the full-scale system implements the complete functioning by the utilities of multiple subsystems that skillfully eliminate the boundary cases and take into consideration the various parameters.
In this JEE college predictor, the full-scale system will accurately predict the suitable college for the user based upon their rank, state, branch preference, and the reservation or quota to be applied.

By maintaining a dynamically updated knowledge base, the full-scale college predictor will be able to make precise predictions that will definitely aid the user in making the right choice.

5. Final implementation, maintenance and evolution

The final implementation of a full-scale system involves rigorous testing, eliminating the boundary cases and responsibly making sure that the product delivered to the users stands by their expectations. We intend to fulfill the resource requirements at the site and henceforth adopt the suitable testing techniques. These test cases will be designed such that they are capable of taking into consideration any ambiguous queries and boundary conditions that might occur thus aiding us in comprehending and producing the correct output for every single query. Our system will aim to successfully not only predict college based on the users' ranks but also their preferences and reservations.

The maintenance of a full-scale system involves developing a dynamic knowledge base that is adaptive to the change in trends, times and knowledge. Also, keeping the system secure is mandatory. The knowledge base that we will build for the JEE Rank predictor will be constantly updated based upon the latest college and branch allotments as per the rank, state as well as the reserved and quota seats. It is absolutely necessary for the knowledge base to be with the latest trends so that the user gets the most accurate prediction of their possible college that is in agreement with the latest allotments. To do so, a historic database is built which is then flooded with the required modifications that are kept track of.

Evaluation of an AI system is a huge challenge since a system aims to surpass the accuracy of outputs produced by a human expert. This is mandatory so that the intelligence of an AI system is measured and ensured. The results produced by the site are compared to the conclusions derived by experts and hence the system gets evaluated. The JEE rank predictor will play a major role in shaping the future of the youth and will in some form, affect their choices and decisions. In such an influential system, it becomes mandatory for the developer to make sure

that the outputs produced is the most relevant result that can be produced as per the inputs provided by the user, ie. rank, state, branch preference and quota.

ACTIVITY TIME SCHEDULE

The plan for implementation based on the proposed methodology is formulated as a milestone approach with associated deadlines.

Milestone	**Objective**	**Deadline**
M1	Obtaining the Dataset (Database)	October 1
M2	Creating the Knowledge Base	October 9
M3	Developing the Inference Engine	October 18
M4	Finding out all the exceptions and corner cases	October 24
M5	Handling the exceptions and corner cases	October 28
M6	Final Testing	November 3
M7	Creating the layout of GUI	November 7
M8	Converting our code into GUI format	November 15
M9	Testing after completion of GUI	November 18

TOOLS AND TECHNIQUES

1. Hardware Interface

Processor	Intel Core i5-4210U
RAM	4 GB or 12 GB, DDR4
Hard disk	256GB

2. Software Interface

Operating system	Ubuntu
Language	Python
Libraries	Tinker, experta, CSVReader

Experta is used for implementing logical programming in python.pyke was integrated to python. Experta rules to direct the configure and combine the python functions. The reason behind choosing Experta is it increase code performance, code adaptability and make code reusable. Tinker is used to make GUI in python which is cross platform and easy to implement.

IMPLEMENTATION

1. **Knowledge base**

College And Branch Predictor's (CABP) knowledge of the college and branch selection is stored in the knowledge base. Unlike conventional programming, where the knowledge is represented implicitly within the structure of a program, the knowledge base in expert systems is explicitly separated from the control module, or inference engine. This makes it much easier to add new knowledge either during program development or in the light of experience during the program's lifetime. This incremental updating of the knowledge base can be done relatively easily; this is particularly

important for such a variety among different colleges and branches.

In CABP, the knowledge is encoded in the form of symbolic IF-THEN rules, which is the standard form of knowledge presentation in expert systems. For example, a rule which checks the criteria for a user with rank to be applicable for a college or not is-

For example :

IF(rank>=opening rank)

THEN

Criteria = satisfied

This knowledge base have rules applied on the net ranks calculated from different years with the help of confidence factors which helps setting the approximate rank according to the preferences of the years. Years has a preference here because of the fact that the most recent year's information will give the closest answer to the actual rank. It also has some rules which will help inference engine to choose among different colleges and branches according to user's preferences.

2. **Inference techniques**

The inference engine is the part of an expert system that has the ability to reason with a knowledge base of rules. It determines what 'reasoning' strategy is used and how the uncertainty is processed. The inference engine is, in effect, the intelligence that allows the expert system to make conclusions based on the 'expertise' or 'knowledge' relevant to the problem domain.

CABP's inference engine uses the standard expert system techniques of chaining (both forward and backward), along with sequential and parallel combination of confidence factors. Briefly, forward chaining occurs when the conclusion (or THEN part) of one rule matches the input to another rule and so on. In this way, evidence can propagate through the knowledge base to allow conclusions to be drawn that are not directly related to the evidence by a single rule. Backward chaining works in a similar fashion from conclusions to causes. The CABP's inference engine uses confidence factor to calculate the checking threshold for the rank for different years.

3. **Database**

The main function of database is to store the information about something which will be used by the inference engine to reach on a rule and to find an answer. In our College And Branch Predictor (CABP), the database stores information about colleges which will help our CABP to find the best college for a user according to his current rank and his preferences.

Firstly the database of our CABP stores the name and address of the college which is a primary information about a college. Then it contains the number of branches the college has and all the necessary details about the college which will help the user to decide whether to take this college or not like its reviews and ratings according to different sources. It also contains the contact details of the colleges. Then it contains information about all the branches the college has and all the information about different branches like the opening rank and closing rank of each branch.

The database of our CABP stores the variable information like opening rank, closing rank for more than one year so that the inference engine can predict the college with all the possible variations and it also contains the precedence for different years.