**LOAN APPLICATION EXPERT SYSTEM**

**SHAKTIMAN  
  
 CS-514**

**Applied Artificial Intelligence**

**Project 2**

CS 514

Applied Artificial Intelligence

Project 2

1

**INDEX**

|  |  |
| --- | --- |
| **TOPIC** | **PAGE NO** |
| **Abstract** | **3** |
|  |  |
| **Features of Loan Application Expert System** | **3** |
|  |  |
| **Rules and Description** | **4** |
|  |  |
| **Usage Manual** | **6** |
|  |  |
| **Sample Runs** | **7** |
|  |  |
| **Test Cases** | **10** |
|  |  |



2

**ABSTRACT:**

Loan Application Expert system is a rule based expert system designed using FUZZY JESS wherein information of a loan applicant is asserted by the user of the Bank, and finally concludes whether applicant is eligible to apply for a loan or not. System also provides suggestions to the user of the system to help him/her to decide in a very convenient manner. The expert system checks for all the boolean and Fuzzy parameters asserted by the user and finally claims the decision depending upon its interpretation and a few important rules set in the bank’s policy.

**FEATURES:**

1. The system asks the bank’s user admin to input all the required details of an applicant applying for a business loan; such as name, age, number of businesses, amount owed, education, experience, credit score etc.
2. The system first checks the age of an applicant whether if its above the minimum requirement.
3. System provides the feature to check for the limit of total amount an applicant can owe.
4. The system analyzes an applicant’s history like *years of education*, and *experience* in the relevant business.
5. An applicant’s chance of approval increases if he/she have cleared all the previously borrowed loans.
6. The system checks for the minimum number of businesses an applicant should possess as an asset to apply for the business loan.
7. Bank always checks for the amount of investment an applicant has already made, or willing to make thereby making an interpretation if an investment made is above the threshold set or not.
8. At the end, the system provides features to input the behavioral aspects of an applicant based on the information collected like, repayment ability and management capability.
9. Finally, an important aspect called *Credit Score* is verified i.e. if he/she has a good or a poor credit score. If poor, the application is rejected straightaway.
10. The rules of the system are sequenced in such a way that if any/all of the top 4 parameters (Legal age, Amount owed, previous loans paid, and credit score) do not meet the required values, then the applicant’s application is rejected.
11. System provides suggestions to the bank employee related to every parameter that is not satisfied to provide employee with the in-depth reasoning about client's information and application status.

3!

**RULES & DESCRIPTION:**

Template 1 mentioned in the code describe variables used for storing applicant’s information

(deftemplate Business\_loan\_applicant (slot applicant\_name)

* + Applicant's age in number(Integer VALUE only) (slot age (type INTEGER))
    - Amount currently owed by applicant in Dollars(Integer VALUE

only)

(slot current\_owe (type INTEGER))

* Any pending payment of previously sanctioned loans?

(slot paid (allowed-values Yes No))

* Applicant's Years of experience in related business (slot exp (type FlOAT))
* Total number of businesses owned by applicant

(slot bus (type INTEGER))

* Years of education of an applicant (slot edu (type INTEGER))
* Total amount invested so far by the applicant in the business (slot inv (type INTEGER))
* Ability to repay loan

(slot repay (allowed-values Good Poor))

; Management Ability of an applicant

(slot mgmt (allowed-values Good Average Poor))

; Credit History of an applicant

(slot credit (allowed-values Good Poor))

)

Template 2 mentioned in the code describes Bank certain parameters.

(deftemplate Bank

(slot bank\_name)

; policy for Repayment ability

(slot repay\_ability (allowed-values Good Poor))

; policy for Management ability

(slot mgmt\_ability (allowed-values Good Average Poor))

; policy for credit history

(slot credit\_history (allowed-values Good Poor))

)

!4

Fuzzy Templates

(deftemplate current\_owe

"Auto-generated"

(declare (ordered TRUE)))

(deftemplate experience

"Auto-generated"

(declare (ordered TRUE)))

(deftemplate business\_owned

"Auto-generated"

(declare (ordered TRUE)))

(deftemplate education

"Auto-generated"

(declare (ordered TRUE)))

(deftemplate amount\_invested

"Auto-generated"

(declare (ordered TRUE)))

In general, system uses 14 rules in total to make a decision for a loan application:

**Rule 0:** Fuzzifies the information of few parameters which have subtle differences intheir value, according to the overall requirement. [check rule 0 in Shaktiman.clp]

1. **initial:** Outputs applicant information.
2. **ageCheck:** Checks if applicants age satisfies the legal age set by loan.
3. **currentOwedAmountCheck:** Checks if applicant’s currently owed amount to bankis higher than the threshold limit.
4. **borrowedLoansCheck:** Checks if applicant has paid previously borrowed loans.
5. **experienceCheck:** Checks for the number of years of experience applicantpossess.
6. **businessesOwnedCheck:** The bank needs minimum of businesses as an asset forthe loan by applicant.
7. **educationCheck:** Checks for the minimum education qualification.
8. **investmentCheck:** Checks for the investment made by an applicant so far.
9. **repayAbilityCheck:** Analyses the ability of an applicant to repay the loan amount.
10. **managementAbilityCheck:** Analyses the ability of an applicant to manage the loanamount (paying interest and EMI’s).
11. **creditScoreCheck:** Checks for the credit score rating of an applicant.
12. **finalDecision\_Impparameters:** Final rule to make a decision of approving a loanapplication or not on the basis of satisfiability of the top 4 parameters set.
13. **printFacts:** To print all the facts.

!5

FUZZIFICATION OF THE DATA:

The MAIN: init-FuzzyVariables rule is used divides the currentOwedAmount, experience, businessesOwned and, education into desirable fuzzy categories with the following template:

Example:

(call ?\*currentOweVar\* addTerm "low" (new ZFuzzySet 50000 100000))

(?\*currentOweVar\* addTerm "medium" (new TrapezoidFuzzySet 100001 140000 160000 180000))

(?\*currentOweVar\* addTerm "High" (new SFuzzySet 180000 200000))

Once the fuzzification is complete, the user defined values for the applicant template are passed into the rule so that they are categorized accordingly and used further

ahead. This is done using:

(assert (current\_owe (new FuzzyValue ?\*currentOweVar\* (new

SingletonFuzzySet ?Business\_loan\_applicant.current\_owe))))

Finally, the rule that was defined for Boolean logic in project 1 can now be divided into all possible subcategories that deal with all the fuzzy sets accordingly.

—Modifiers Used for Fuzzy Variables: low, moderate/medium, high

**USAGE MANUAL:**

1. Create a new Java project in eclipse. Make sure you include the JAR file “fuzzyJ-2.0.jar” under New Project > Libraries.
2. Add Shaktiman.clp to the src folder of the project or create a new file and copy the contents of Shaktiman.clp into the blank file. Make sure you save the file with .clp extension.In the run configurations of the file, change “jess.Main” to “nrc.fuzzy.jess.FuzzyMain”.
3. Run the project. In case you run into any errors, make sure that the run configs is pointed to the FuzzyMain as by default it is shifted back to jess.Main.

- In case the grader wants to tweak the input, he has to make changes to the init rule at the end of the program. The assertion there looks like:

(assert (Business\_loan\_applicant (applicant\_name "Shaktiman")

(age 18) (current\_owe 99999) (paid Yes)

(exp 2.5) (bus 2) (edu 5) (inv 30000)

(repay Good) (mgmt Poor) (credit Good)))

!6

4. And the allowed input values can be referred from through the template.

**SAMPLE OUTPUTS:**

***NOTE:* (Minimum, Maximum) value that can be entered for every fuzzy variableare as follow:**

**current\_owe: (0, 200000) ; exp :(2.0, 12.0) ; bus:(1, 10) ; edu(0, 15); inv(0, 500000)**

****

1. **SAMPLE 1**

(assert (Business\_loan\_applicant (applicant\_name "Shaktiman")

(age 18) (current\_owe 99999) (paid Yes)

(exp 2.5) (bus 2) (edu 5) (inv 30000)

(repay Good) (mgmt Poor) (credit Good)))

Output:

Please see images in the images folder of the project

1. **SAMPLE 2**

(assert (Business\_loan\_applicant (applicant\_name "Shaktiman 1")

(age 18) (current\_owe 200000) (paid Yes)

(exp 12.0) (bus 10) (edu 2) (inv 90000)

(repay Good) (mgmt Good) (credit Good)))

Output:

Please see images in the images folder of the project

**OTHER TEST CASES: Please use only one test case at a time in the Shaktiman.clp file**

(assert (Business\_loan\_applicant (applicant\_name "Shaktiman 3")

(age 25) (current\_owe 50000) (paid Yes)

(exp 12.0) (bus 10) (edu 15) (inv 500000)

(repay Good) (mgmt Poor) (credit Good)))

(assert (Business\_loan\_applicant (applicant\_name "Shaktiman 4")

(age 16) (current\_owe 110000) (paid No)

(exp 1.0) (bus 1) (edu 5) (inv 4000)

(repay Poor) (mgmt Average) (credit Poor)))

10