Smart Banking - Advisor Bot

(Artificial Intelligent CPSC 6820)

(Project Report)

Unmesh Kanchan
Department of Computer Science
Clemson University
Clemson, USA
ukancha@g.clemson.edu

Ashwini Aras
Department of Computer Science
Clemson University
Clemson, USA
aaras@g.clemson.edu

***Abstract* — In today's world, Artificial Intelligence has gained immense importance and has become one of the key fields in science and technology. Artificial Intelligence has multiple applications in fields like robotics to high level scientific computations. Banking and finance is a core area for implementation of such an intelligent agent. Artificial Intelligence has various applications in banking and financial field, some of which are organizing operations, maintaining book-keeping, investing in stocks, managing properties. People require advice on taking financial decisions and financial planning. We have developed an advisor bot that will analyze a customer's financial history and help him/her in taking financial decisions. It will also answer customer's banking related queries. For developing this 'Smart Banking - Advisor Bot' system, we consider the use of Artificial Intelligence in banking and financial field.**

***Index Terms* — AI (Artificial Intelligence), Chatbot, SIML (Synthetic Intelligence Markup Language)**

I. Introduction

A. Motivation

Majority of the people find themselves in a fix when it comes to financial matters. Not everyone is capable of making wise decisions when it comes to managing their finances. In such cases, a system that provides guidance for managing finances and bank accounts can prove very helpful. AI has proved to have various applications in the Banking and Finance sector. The main motivation of this project is applying AI concepts to help people take smart financial decisions

B. Proposed System

The system developed provides user with 4 main services viz. 1. Smart Financial Plan, which generates intelligent financial plan by analyzing user's priorities; 2. Financial Decision Help, which helps users in making wise financial decisions by analyzing the user's financial data; 3. Smart Financial Reports, in which we develop intelligent financial report generation system based on the smart analysis of financial data; and 4. Chatbot, which provides answers to users after analyzing their queries.

II. System Design

A. System Architecture

Figure 1 depicts the architecture of the system for smrt banking - advisor bot. The general architecture is the three - tier architecture of ASP.NET.

- *Data* - It is the main SQL database server.
- *Data layer* - It is the tier which contains all the stored SQL procedures.
- *Business layer* - It contains all the C# classes of business logic development
- *Presentation layer* - It contains ASP.NET web forms, web user controls, master pages, etc.

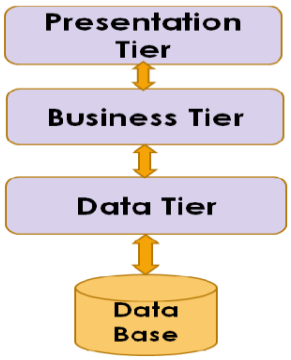


Fig. 1. Smart Banking - Advisor Bot System Architecture

System Design:

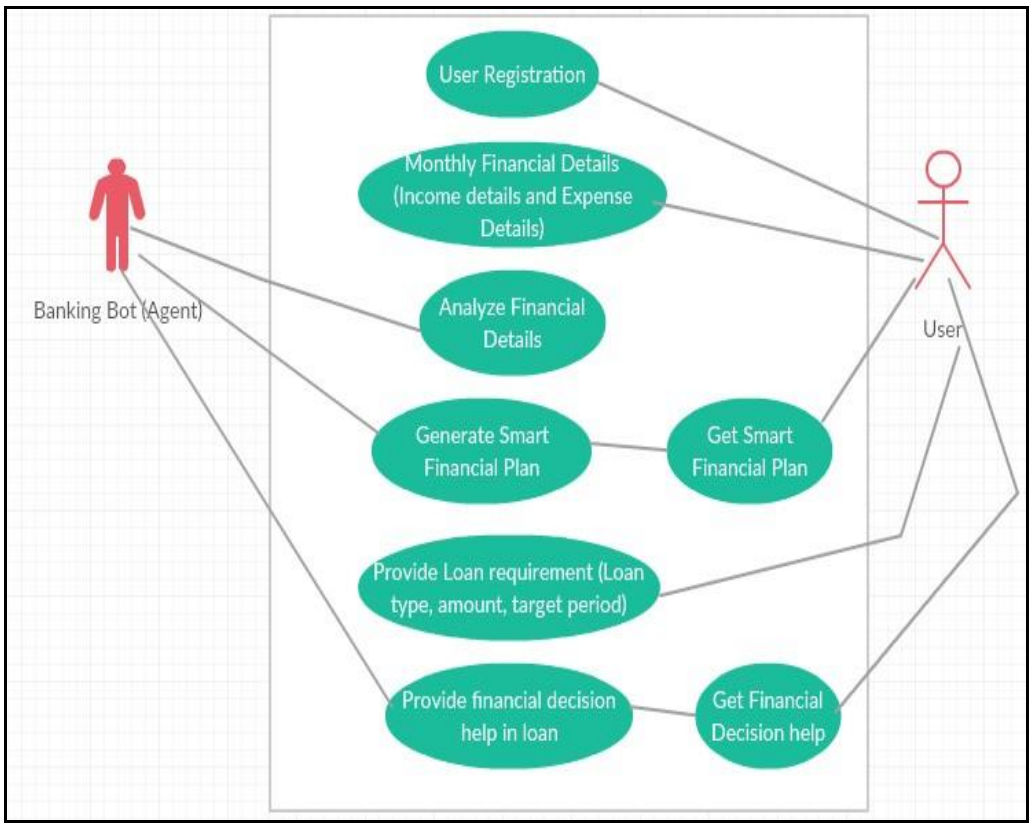


Fig. 2. System Use - Case Diagram

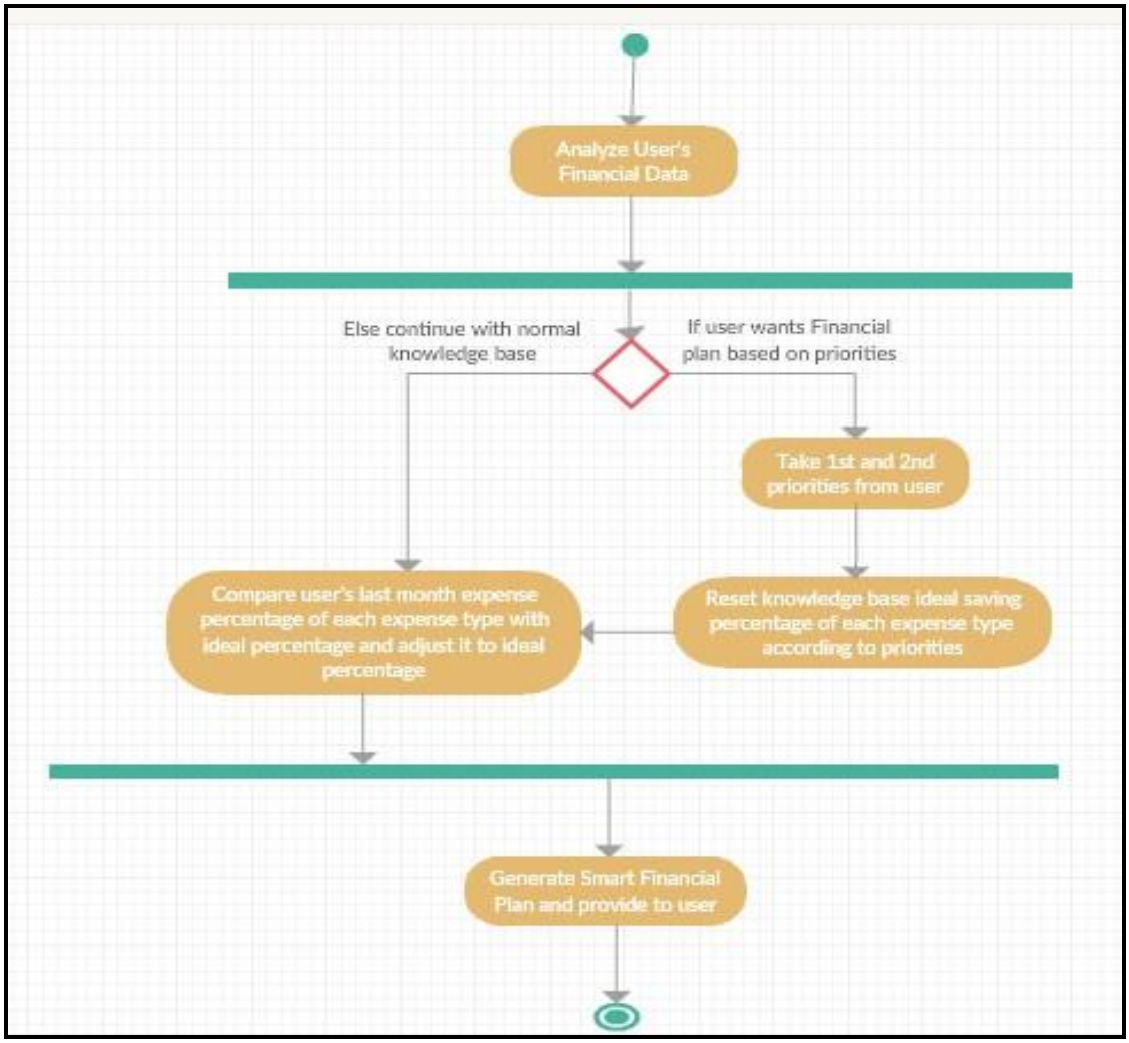


Fig. 3. Smart Financial Plan Activity Diagram

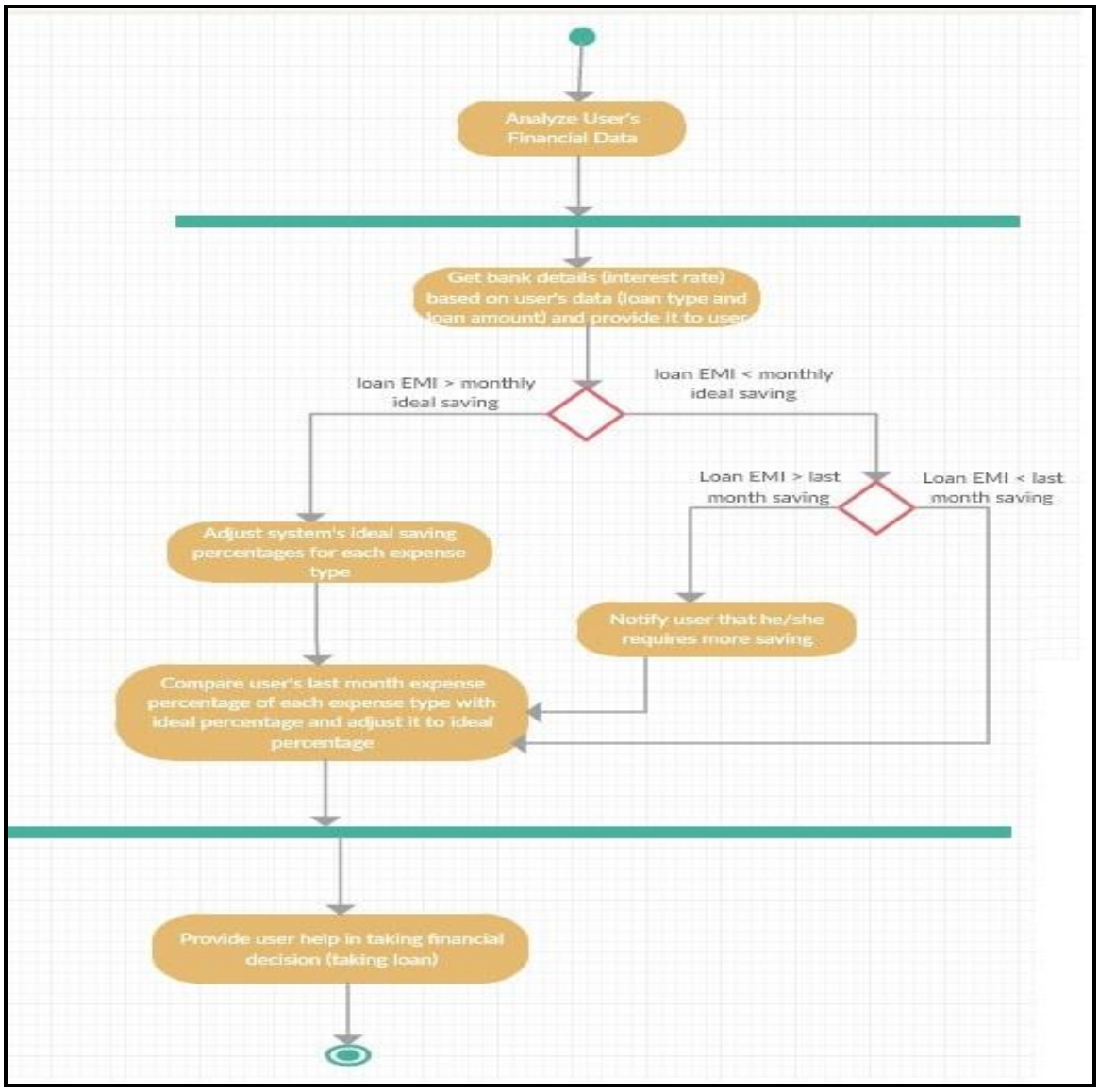


Fig. 4. Financial Decision Help Activity Diagram

B. Working of System

The system deploys 4 models :

1. ***Smart Financial Plan*** - This feature helps users to make financial planning. User is asked to provide the information about financial activities and system provides smart plan to user. In this module, a system that analyzes user's priorities is used to generate intelligent financial plan.

2. ***Financial Decision Help*** - This feature assists users in making various financial decisions e.g. in case of applying for any type of loan, the intelligent system provides options of banks and financial companies which are providing minimum interest rate on that type of loan. It also provides information to users whether it will be smart for them to apply for the loan now or whether they should wait for a certain amount of time after which it will be advisable for applying for the loan, based on financial data analysis.

3. ***Smart Financial Reports*** - This feature generates financial reports based on the analysis of user's financial data. In this module, an intelligent financial report generation system is developed based on the smart analysis of financial data.

4. ***Chatbot*** - This feature analyzes customer's queries about services, loan, account, banks' policies and provides answers to these queries.

III. SMART FINANCIAL PLAN

It is of utmost importance that a person distribute and manage his/ her expenses optimally. So, the first intelligent feature provided by the Banking Bot is Smart Financial Plan. Here, banking bot or agent learns about basic needs and priorities of human being. This agent first learns different financial and economic models. From this learning, agent categorizes all the expenses in five main expense type categories viz. 1. Household (Expenses coming under this category : house rent, expense on utilities, expenses on groceries) 2. Health 3. Education 4. Travel/Transport 5. Lifestyle (Expenses coming under this category : expenses on shopping, eating outside, other expenses etc.). The banking bot then learns that home, health, education are basic needs of human being. So the Bot sets priorities of expense types and percentages of total income the user can save to spend on these expense types for coming month as follows:

1. Household : 23%
2. Health : 14%
3. Education : 10%
4. Travel : 10%
5. Lifestyle : 9%

These are the ideal expense percentages advised by the bot. Here banking bot does not advice user to spend his entire income (100%) on these expense types because there may arise an emergency situation where user will require money and if all of it is spent, the user will get into big trouble if he does not have enough money. As per the above percentage distribution, the bot advices user to save at least 34% of total income. Considering all these as knowledge base, our banking bot provides user smart financial plan.

Sometimes, user may have different priorities for expense types. In this case banking bot asks user for his first two priorities and based on these priorities, the bot provides user with smart financial plan. Banking bot first analyzes the user's financial data, i.e., monthly income details and last month's expense details. Based on this analysis, it provides user the details of next month's financial plan. Internally, banking bot performs various operations and at the end provides user smart financial plan. Internal operations are explained in detail through the following algorithm.

A. Algorithm

This section explains the algorithm for smart financial plan feature of the system in detail. This algorithm is a combination of different tasks or operations the system performs to provide user the smart financial plan which the user can follow or which can help user to plan financial budget. The algorithm includes the system's data analysis task, computational task and presentation of the information to user. Here the optimality of algorithm is based on various factors like user's last month's expenses and income details. However for any kind of input, this algorithm provides optimal solution or result. In this feature user has two options to get financial plan. First one is based on priorities, like, for which types of expense types user wants to save money. Other option is one in which user can get financial plan without any priorities.

Step1 : Check whether user is a first time user or an already registered one.

1. If user is first time user then, user needs to enter monthly financial details i.e. income and expense details
2. Else if user has already registered then, check whether financial information is updated or not,
   a. If financial information is updated then goto step 2.
   b. Else update financial information before proceed further.

Step2 : Function : Financial_Plan_Generation (Input : User's monthly income and expense details).

1. First,
   a. If user wants financial plan based on priorities, adjust ideal saving percentages for each expense type as :
      i. Assign highest saving percentage to user's first priority expense type
      ii. Assign second highest saving percentage to user's second priority expense type.
      iii.And then remaining expense type saving percentages get assigned by the importance of expense type that intelligent system decided.

   b. Else If user wants financial plan without any priorities,

i. Consider the existing ideal saving percentage for expense types.

2. Compute total income
3. Now,
   a. If user's expenses on education are null, then distribute the ideal saving percentages for education expense type over other expense types as follows :

New saving percentage for expense type = existing saving percentage + (((existing saving percentage/66)*100) * ideal saving percentage on education expense type)/100.

(Here, the increment in ideal saving percentage for expense type is decided based on the percentage

of this ideal saving percentage for expense type of total of all saving percentages).

b. Else, continue with existing ideal saving percentage for each expense type.

4. Compute the difference between last month's expense percentage on expense type and ideal saving percentage for expense type. This computation is performed for all expense types.

a. If difference > 0 (Means actual expense percentage for last month is greater than ideal percentage, then store that difference in one array (positive array) and expense type in other array.

arrExpense[i] = difference;
arrexpName[i] = expense type;

b. Else if (difference <= 0), then store that difference in one array (negative array) and expense type in other array.

arrexpenseneg[m] = Math.Abs(difference);
arrexpName1[m] = expense type;

5. Sort both arrays in descending order.

6. Calculate saving difference as
savingdiff = basicSaving - (100 - totalExpensePercentage)
7.Calculate remaining saving as : 66 - totalExpensePercentage;

a. Adjust expense types percentages as follows :
For each entry in negative array (means expense types for which saving percentage is less in last month compared to ideal saving percentage for expense type)

i . While difference for expense type in negative array != 0

For each entry in positive array :

If (expense difference for particular expense type >= percentage entry in negative array)

- Reduce the positive difference by this negative difference
- Negative difference= 0;

Else if(expense difference for particular expense type < percentage entry in negative array)

- Reduce the negative difference by this positive difference
- Positive difference =0. Go to i.

b. After adjusting saving percentages for expense types, if still there is percentage for expense type exceeds ideal percentage for that expense type then, add this extra percentage to basic saving (34%).

c. Then update all expense values as per the results obtained from above steps.

8. At the end of above all operation and computations, the final smart financial plan get generated.

IV. FINANCIAL DECISION HELP

People often seek professional guidance to help them in making good financial decisions. This system feature implements the functionality and develops the system in three tier architecture in ASP.Net technology using C# for front end and all logic development, and Microsoft's SQL server for database functionality. In this three tier architecture, first the financial data like user's monthly income details (income sources and amount), expense details (user's monthly spending on type of expenses) and bank details like bank name, types of loan offered by banks and their yearly interest rates get stored in database. This data is then fetched to front end using data access layer logic in system architecture. When user wants financial decision help from system, this intelligent feature fetches the required data from database, analyzes it and provides user help in the form of detailed information like banks providing loan with minimum interest rate, loan EMI user needs to pay. Based on analysis of user's income and expense details, system provides user information regarding how to increase his/her savings in case user doesn't have enough savings to pay loan EMI. The user needs to save in order to satisfy monthly loan EMI goal and ultimately meet the target period that user set for paying total loan amount at the end of period. In case where user has already saved sufficient amount to easily pay monthly EMI, system also provides user detailed information about his spending on each expense type and if user fails to achieve the target saving for these expense types (user's priorities), system will assist user to plan his monthly financial budget where system arranges the ideal expense percentages in a way that user's priorities get highest percentages and according to this, percentages of remaining expense types get arranged. System compares the user's monthly expenses percentages with ideal percentages that system has proposed based on analysis of economic models of personal financial budget. And as per the comparison result, this banking advisor bot provides information about which expense type the user need to save on and how much of the total income does he need to save. In case user fails to save amount based on ideal percentages for user's priority expense types, system helps user by providing information about

excessive amount user is spending on certain expense types which can be otherwise saved for prioritized expense types. If goal of savings for these priorities is achieved and there is still some more remaining than desired ideal amount (calculated based on ideal percentage), system tell user that this additional amount can be saved and used for other expense types in case of any urgency or emergency.

A. Algorithm

This section explains the financial decision help algorithm in detail. This algorithm is combination of different tasks or operations that system performs to help or assist user in taking financial decisions like applying for a loan, how much saving needs to be done on different expense types, etc. The algorithm includes the system's data analysis task, computational task and presentation of the information to user. Here the optimality of algorithm is based on various factors like user's target period of paying total loan amount, interest rate bank provides for particular loan type. However for any kind of input, this algorithm provides optimal solution or result.

Step1 : Check whether user is first time user or already registered user.
1. If user is first time user then, user need to enter monthly financial details i.e. income details and expenses details.
2. Else if user has already registered then, check whether financial information is updated or not,

a. If financial information is updated then go to step 2.
b. Else update financial information before proceed further.

Step2 : To provide help in taking financial decisions, banking bot provides user bank details viz. bank name and interest rate on the loan type provided by user based on details provided by user i.e. loan type and target period in months. Here, system provides user details of banks offering minimum interest rate.

Step 3 : Function : financialDecisionHelp (Input: loan type, interest rate, target period, loan amount)
1. Analyze user's income details and expenses details and categorized the expenses types and combine some expenses in particular expense type.
2. Banking bot computes following things :

- Total income
- Monthly ideal saving amount: (totalIncome * 34) / 100;
- Last month saving = totalIncome-lastMonthexpense;
- Last month saving percentages
- Monthly Interest amount : (loanAmount*(interestRate/100))/12;
- Annual Interest amount: annualinterestAmount*targetMonths;
- Total amount user has to pay at the end of target period: loanAmount+ interestAmount;
- Monthly loan EMI: totalTargetLoanAount / targetMonths;
- Loan EMI percentages: (loanEMI/totalIncome)*100;

3. If 34.00 < loanEMIpercentage AND loanEMIpercentage <= 50.00, then
a. Compute the difference between current ideal saving percentage(34%) and monthly loan EMI percentage.i.e.

Difference = loanEMIpercentage - 34.00;

Adjust the ideal saving percentages on each expense types in order to save more amount for loan EMI as follows :
New Ideal saving percentage for expense type= (((Current ideal percentage / total percentage of expenses) x 100) x difference) /100.

b. Compute the difference between last month's expense percentage on expense type and ideal saving percentage for expense type. This computation is perform for all expense types.
If diffenence> 0 (Means actual expense percentage for last month is greater than ideal percentage), then store that difference in one array (positive array) and expense type in other array.

arrExpense[i] = difference;
arrexpName[i] = expense type;

else if (difference <= 0), then store that difference in one array (negative array)and expense type in other array.

arrexpenseneg[m] = Math.Abs(difference);
arrexpName1[m] = expense type;

c. Sorting both arrays in descending order.

d.Calculated remaining saving: Total ideal saving percentage on expense type - Last month's total expense percentage.
e. Adjusting expense types percentages as follows :
For each of the entry in negative array (means expense types for which saving percentage is less in last month compare to ideal saving percentage for expense type)
I. While difference for expense type in negative array != 0

For each entry in positive array :
If (expense difference for particular expense type >= percentage entry in negative array)

- Reduce the positive difference by this negative difference
- Negative difference= 0;

Else if(expense difference for particular expense type < percentage entry in negative array)

- Reduce the negative difference by this positive difference
- Positive difference =0. Go to I.

4. Else If 34.00 >= loanEMIpercentage (means Loan EMI amount will get save easily as per ideal saving percentage)
i.->If loanEMIpercentage > lastMonth Saving Percentage AND lastMonthSavingPercentage < 34.00 (Means user need to save more amount as compare to last month's total saving to pay Loan EMI)

a. Compute the percentage that user need to save more to pay Loan EMI

MoreSaving=loanEMIpercentage - lastMonthSavingPercentage

b. Now check in positive difference array means check for which expense type the last month's expenses is greater than ideal saving percentages for that expense type. For each such entry in positive array,

I. While MoreSaving !=0
If(positive difference >= MoreSaving), then Reduce positive difference by MoreSaving and make MoreSaving=0

Else if (positive difference <MoreSaving), then reduce MoreSaving by positive difference and make positive difference =0. Go to I.

c. Now after satisfying saving goal for Loan EMI, adjust the saving percentages of expense types as follows :
For each of the entry in negative array (means expense types for which saving percentage is less in last month compare to ideal saving percentage for expense type)
I. While difference for expense type in negative array != 0
For each entry in positive array :
If (expense difference for particular expense type >= percentage entry in negative array)

- Reduce the positive difference by this negative difference
- Negative difference= 0;

Else if(expense difference for particular expense type < percentage entry in negative array)

- Reduce the negative difference by this positive difference
- Positive difference =0. Go to I.

d. After adjusting saving percentages for expense types, if still there is percentage for expense type exceeds ideal percentage for that expense type then, add this extra percentage to basic saving (34%)
ii.-> Else if lastMonthSavingPercentage >= loanEMIpercentage AND lastMonthSavingPercentage <= 34.00 (Means saving is enough to pay EMI of loan but user may follow ideal financial plan)

a. Compute the difference between last month's expense percentage on expense type and ideal saving percentage for expense type. This computation is perform for all expense types.
if diffenence > 0 (Means actual expense percentage for last month is greater than ideal percentage), then store that difference in one array (positive array) and expense type in other array.

arrExpense[i] = difference;
arrexpName[i] = expense type;

else if (difference <= 0), then store that difference in one array (negative array) and expense type in other array.

arrexpenseneg[m] =Math.Abs(difference);
arrexpName1[m] = expense type;

b. Adjust the saving percentages of expense types as follows:
For each of the entry in negative array (means expense types for which saving percentage is less in last month compare to ideal saving percentage for expense type)
I. While difference for expense type in negative array != 0
For each entry in positive array :
If (expense difference for particular expense type >= percentage entry in negative array)

- Reduce the positive difference by this negative difference
- Negative difference= 0;

Else if(expense difference for particular expense type < percentage entry in negative array)

- Reduce the negative difference by this positive difference
- Positive difference =0. Go to I.

c. After adjusting saving percentages for expense types, if still there is percentage for expense type exceeds ideal percentage for that expense type then, add this extra percentage to basic saving (34%).
5. Then update all expense values as per the results obtained from above steps.

V. SMART FINANCIAL REPORTS

The objective of this feature is to generate different types of reports based on the analysis of user's financial data. System provides user reports in graphical form so that user can easily understand and use these reports. Types of reports generated by the system :

1. *Expense report* - based on user's expense data
2. *Income report* - Based on user's monthly income data
3. Comparison of expense percentages' amount with ideal saving percentage and amount report.

VI. CHATBOT

This feature analyzes customer's queries about services, loan, account, banks' policies and provides answers to these queries. The system will first take input from bank customer. Based on input, the system processes the query and gives response to the user. This module is implemented in C# language and .NET framework. The chatbot is added to C# application using SIML which is implemented in Syn Chatbot Studio and deployed in Visual Studio 15. Following steps are to be followed :

1. Import the Syn.Bot library into Visual Studio which is available as a NuGet package.
2. Create a new C# WPF application targetting .NET 4.5 or higher.
3. *SIML* - SIML stands for Synthetic Intelligence Markup Language. It is an XML base and the files support xml formatting. A typical siml code segment is as follows :

```
<Siml>
  <Concept Type="public" Name="Hello Bot">
    <Model>
      <Pattern>Hello Bot</Pattern>
      <Response>Hello User!</Response>
    </Model>
  </Concept>
</Siml>
```

- Each siml xml files start with the <Siml> tag and end with </Siml> one.
- *Concept* - It is the subject or topic of the siml knowledge base. It may contain thousands of Models.
- *Model* - It is the basic unit of knowledge. It contains Patterns and Responses. One Model can contain infinite Patterns but only one Response element.
- *Pattern* - It is a set of words or tokens that are inspected at runtime against the user input to see if user input matches the specified sequence of words or tokens.
- *Response* - After a Pattern is inspected and satisfies every specified condition, the Response element is evaluated and sent back to the user.

4. Import knowledge base into Visual Studio.
5. Deploy the chatbot in Visual Studio.

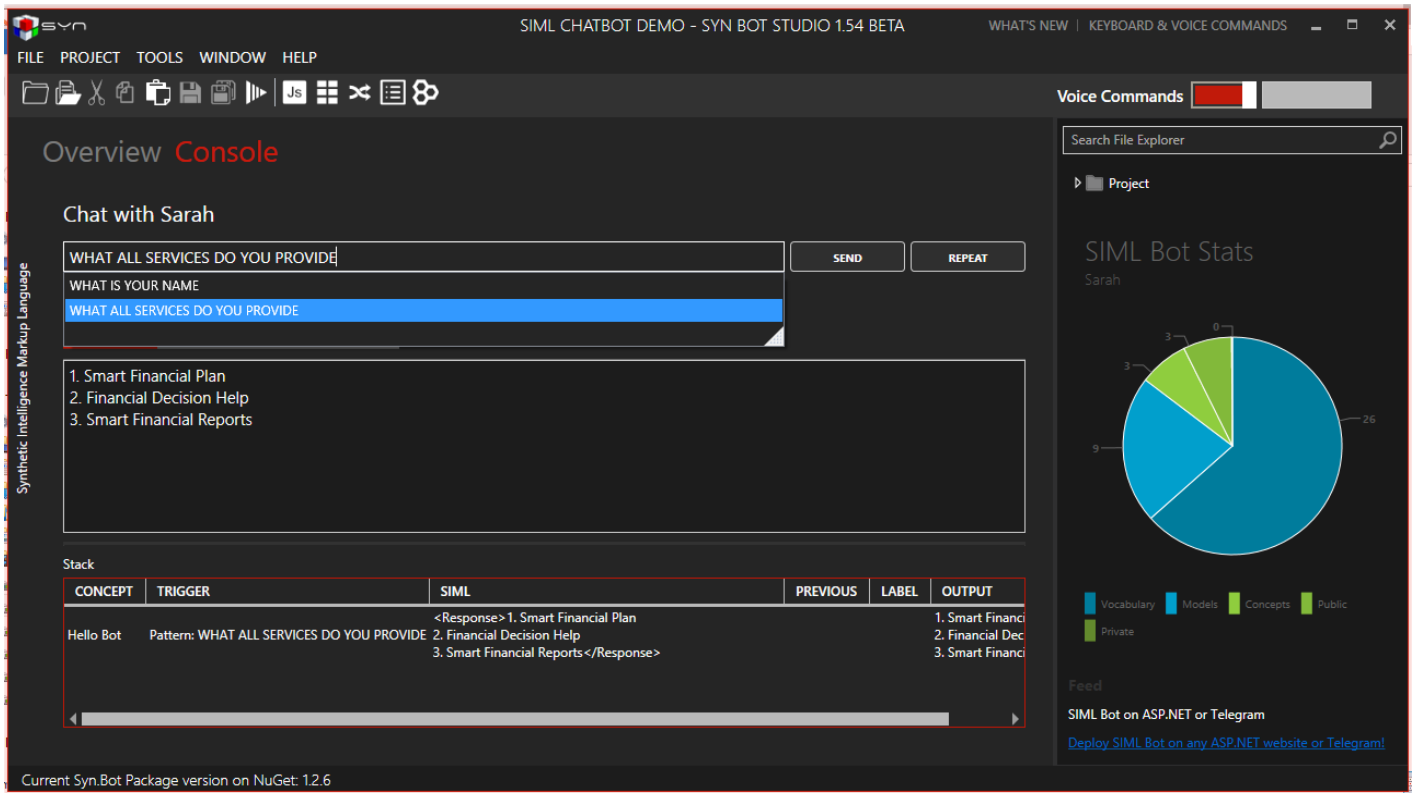


Fig. 5. Screenshot of ChatBot

VII. Evaluation

The system is evaluated for a C# application using ASP.NET architecture. Chatbot is evaluated for windows machine using Syn Chatbot Studio and Visual Studio. The chatbot supports just the text format.

VIII. Results

After the successful development of the banking advisor bot system, the system was run and the following results were obtained:

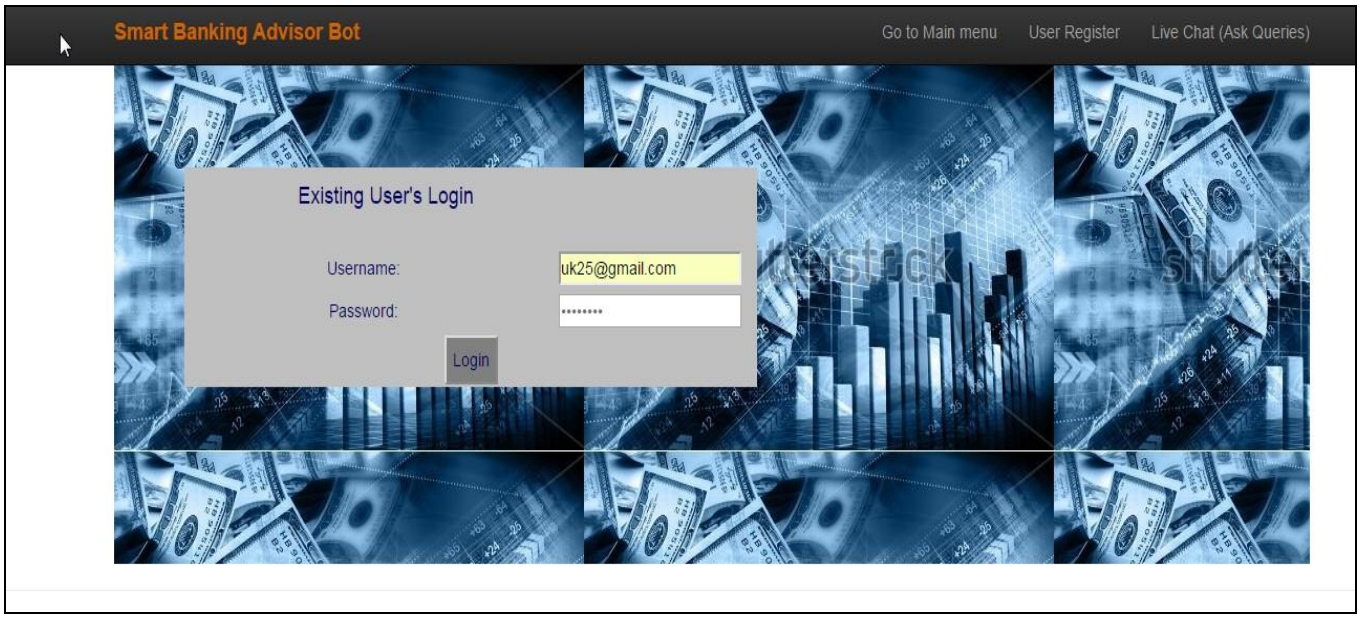


Fig. 6. User Login

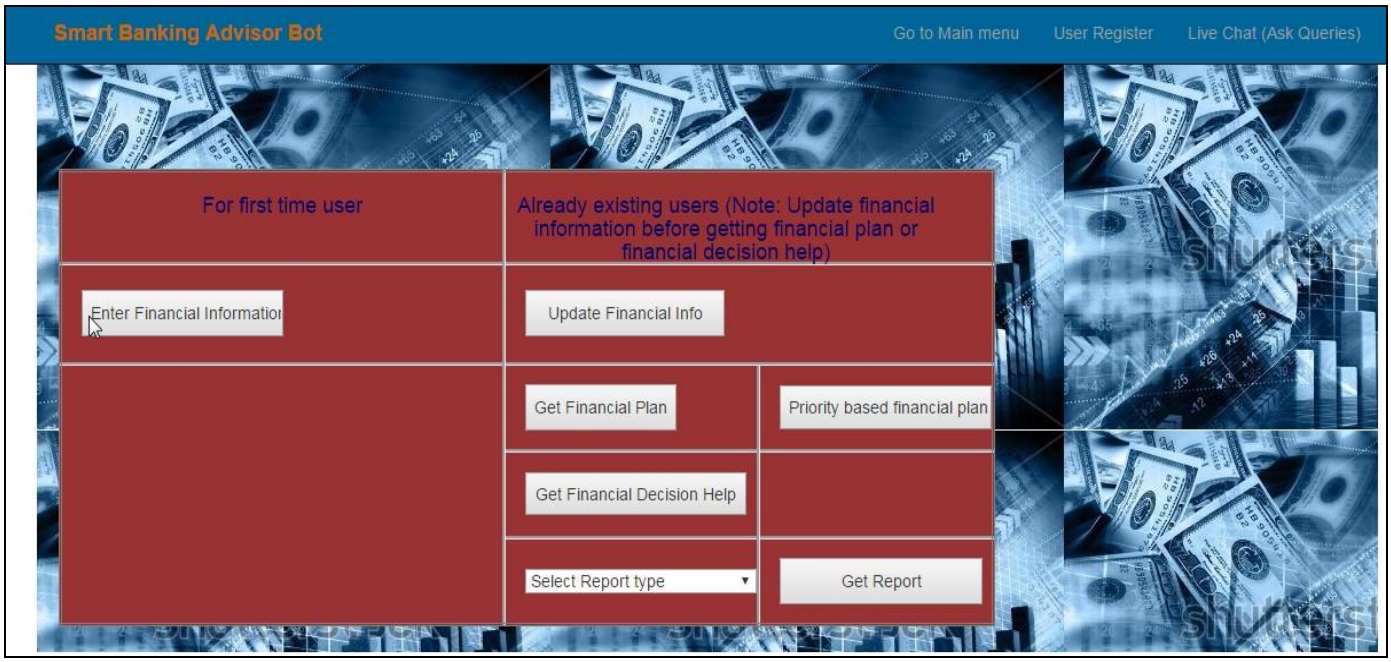


Fig. 7. Once user logins into system, user will get options (features of system) as above

Fig. 8. If it is first time user, he/she must enter financial information (last month's income and expense details); If user is already existing one, he/she can update financial information before getting financial plan or financial decision help

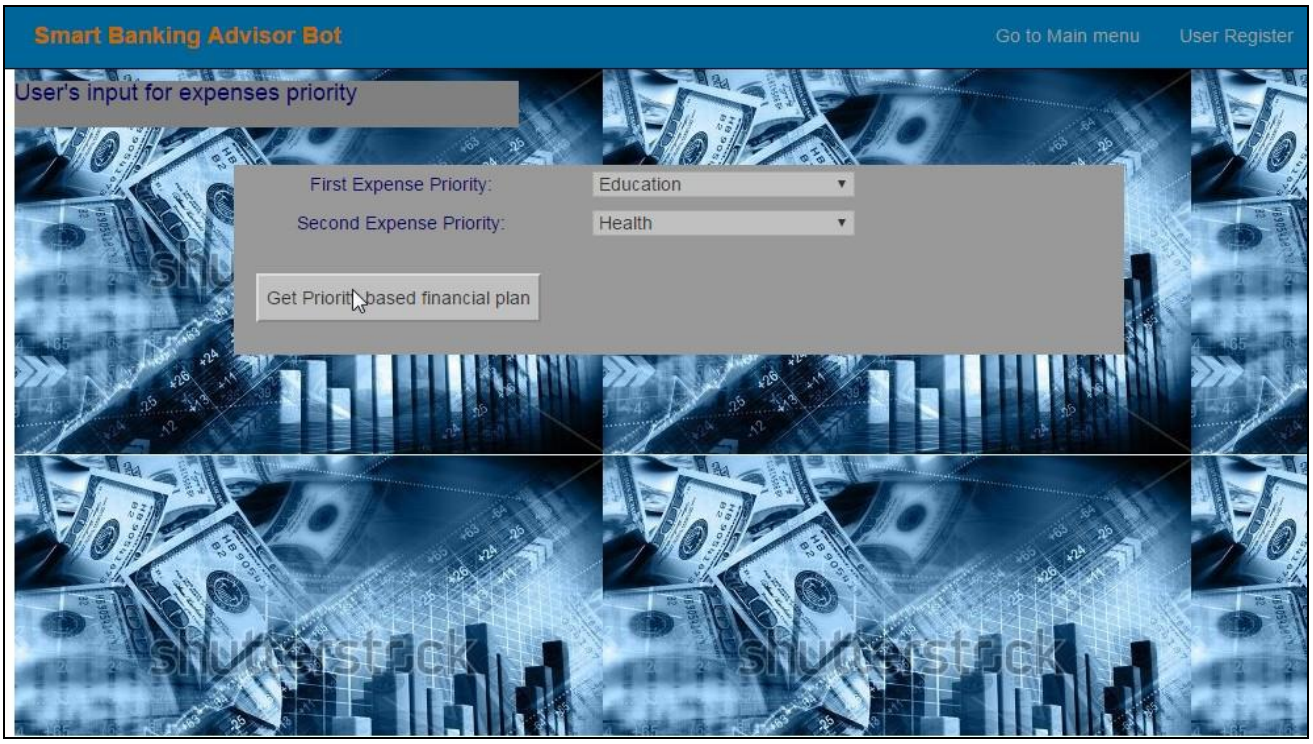


Fig. 9. Priority based Financial Plan

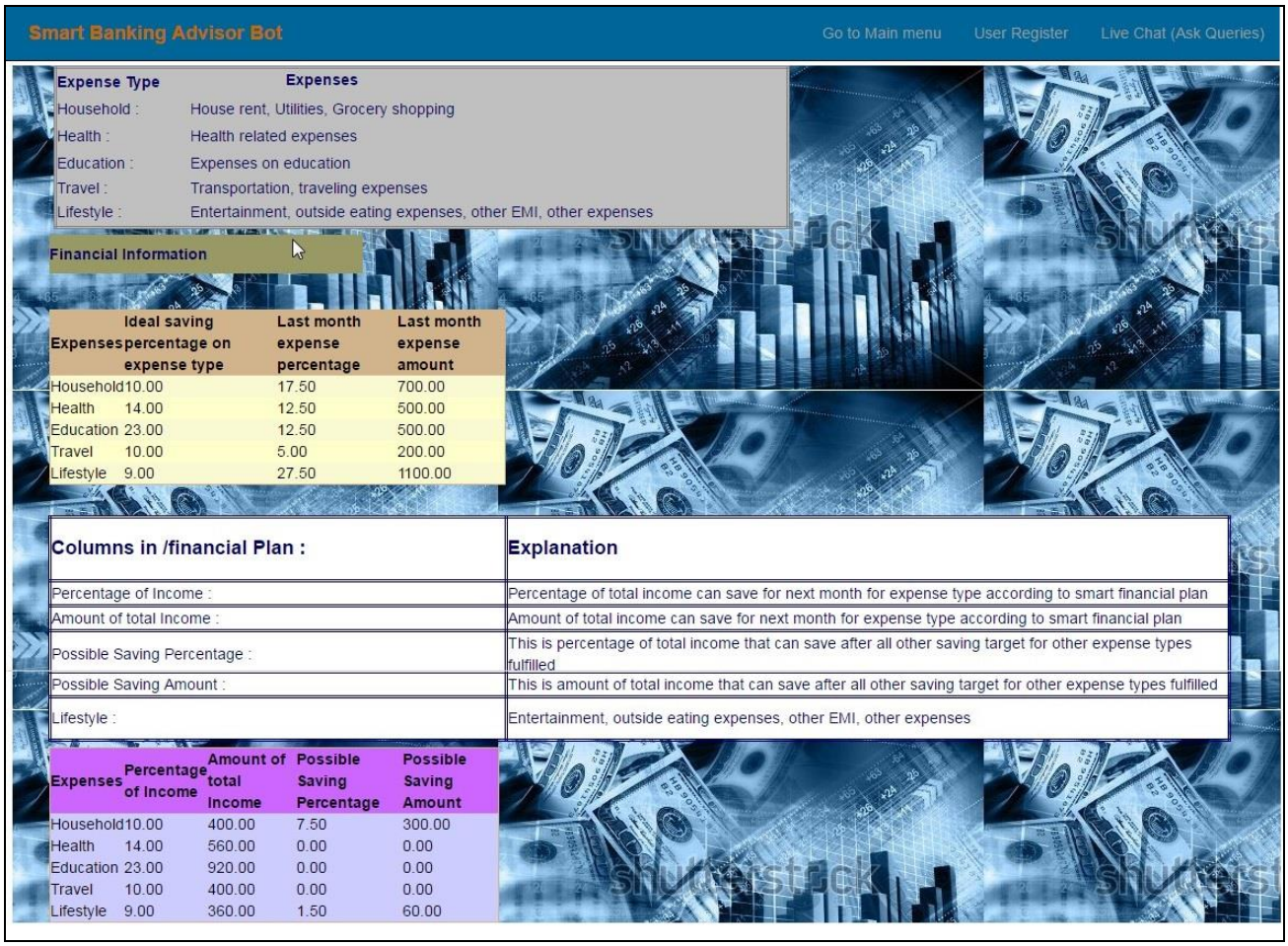


Fig. 10. Financial Plan as per user's priorities

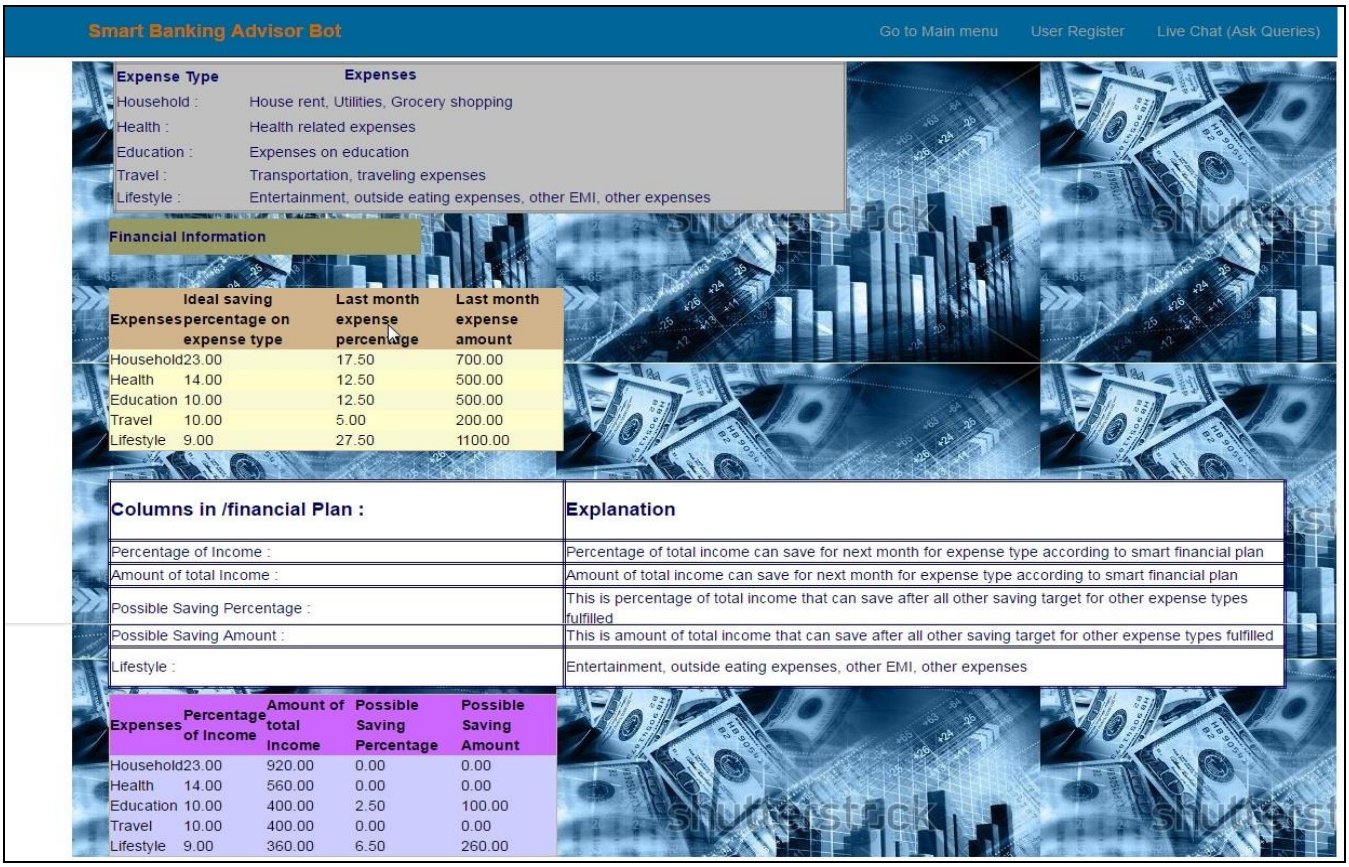


Fig. 11. Financial Plan without priorities

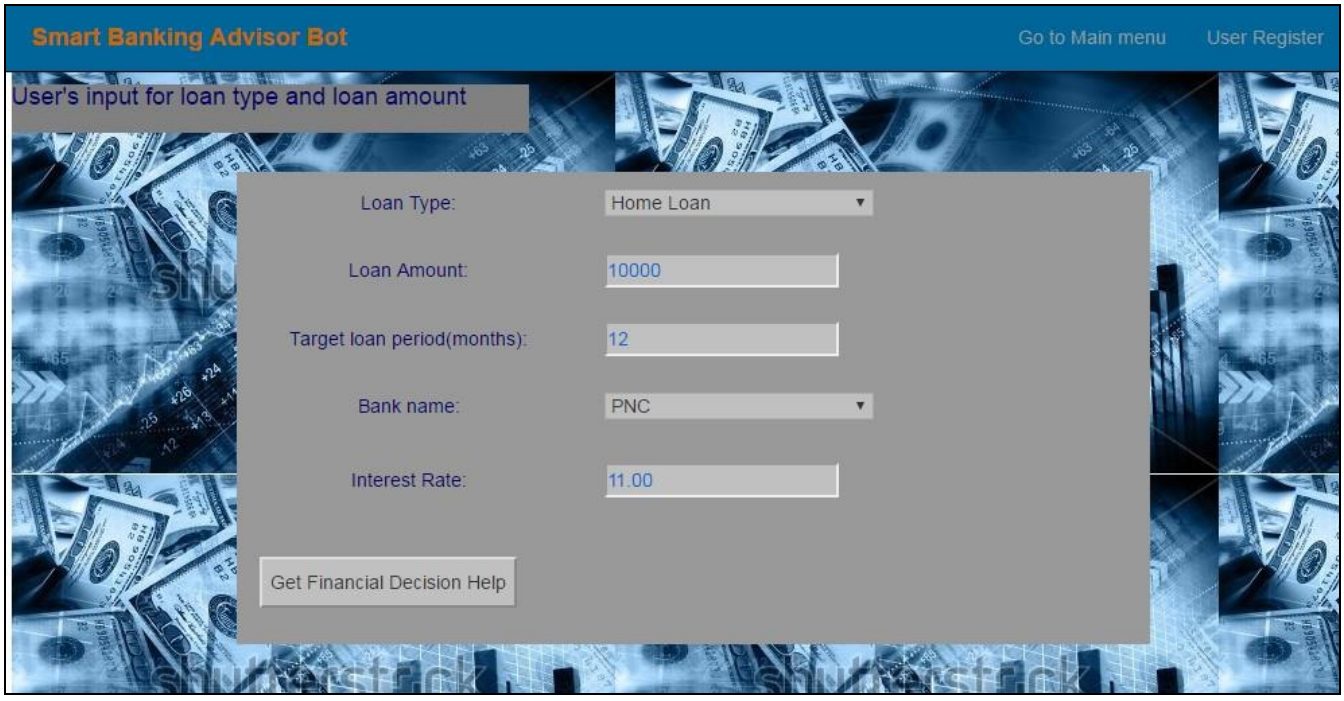


Fig. 12. User needs to provide details for financial help

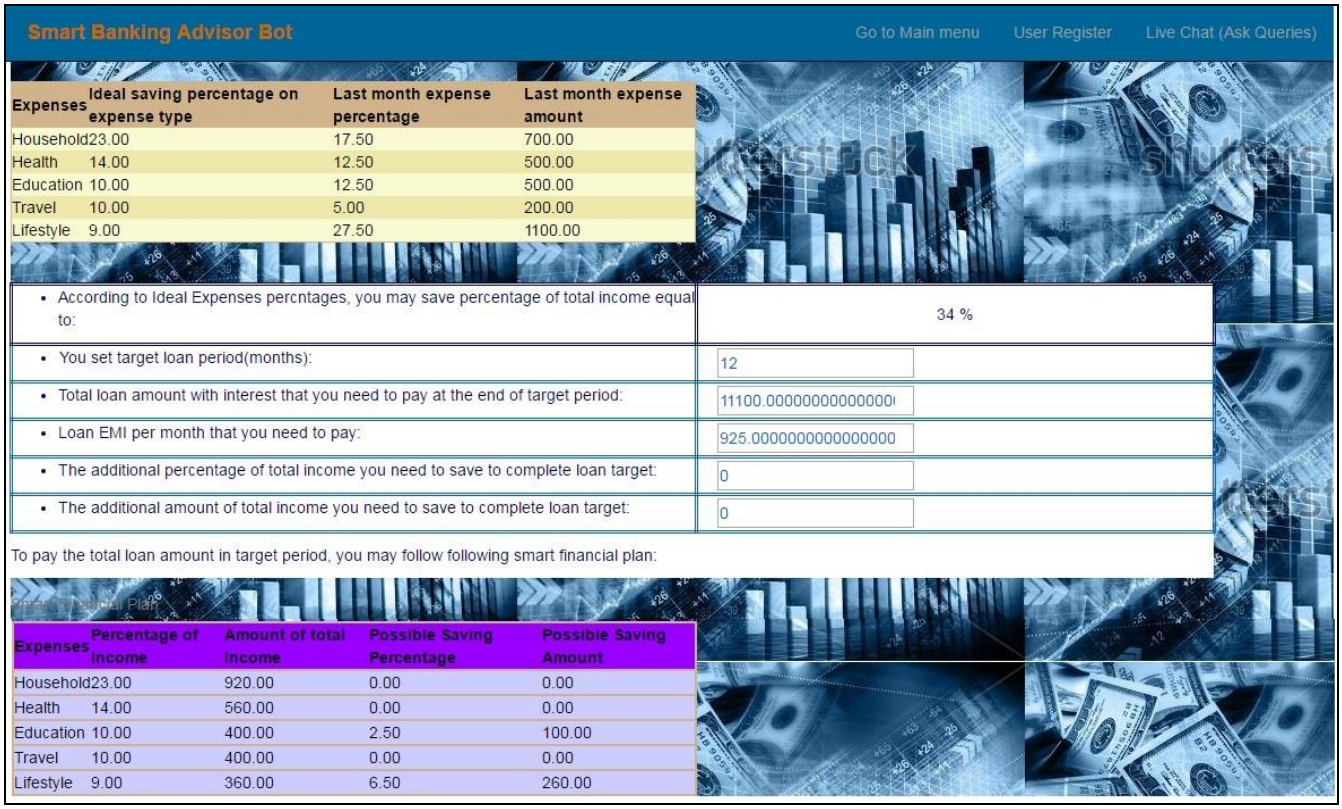


Fig. 13. Financial Decision Help

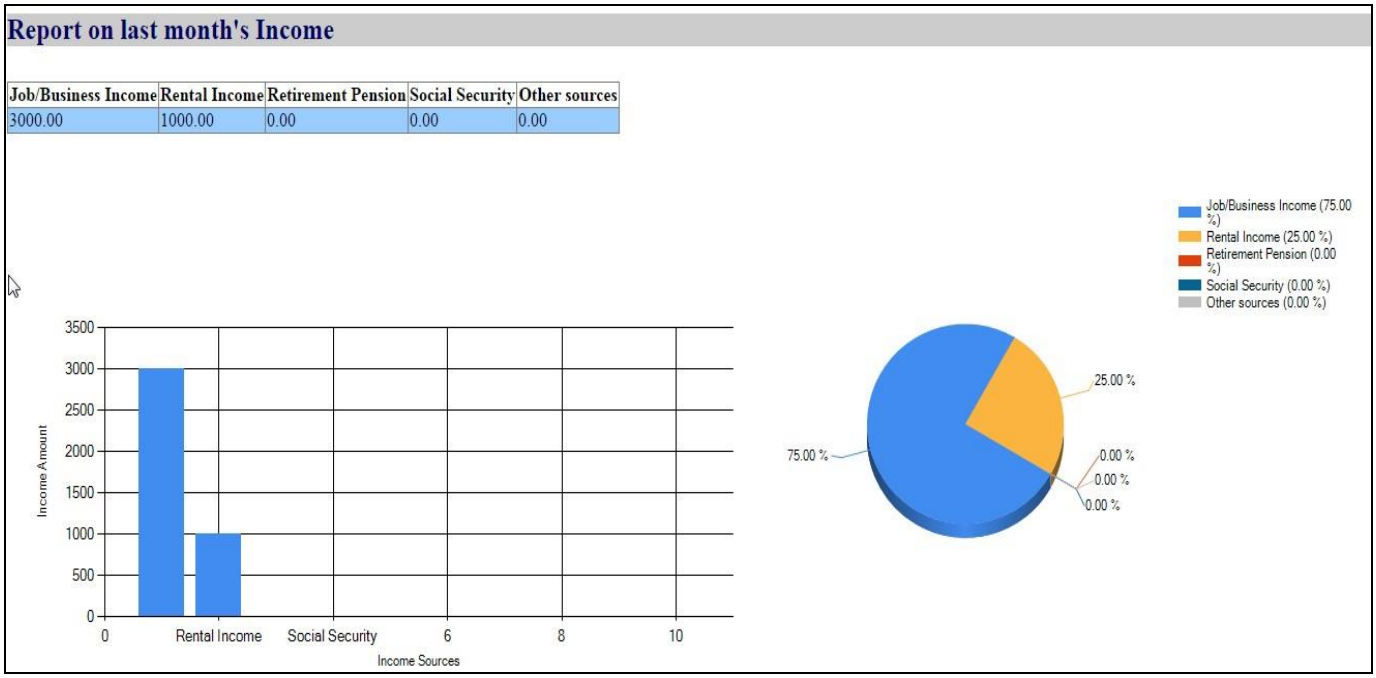
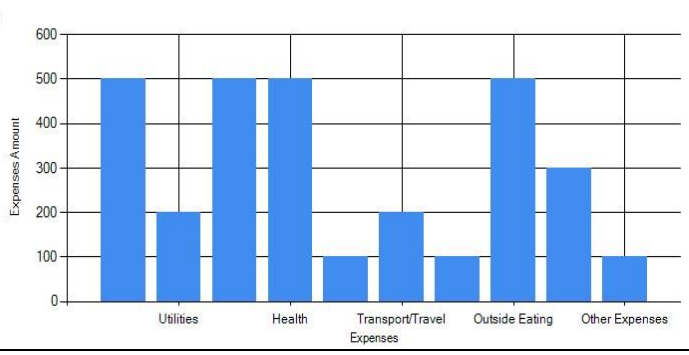


Fig. 14. Income Report

Report of Last month's Expenses

house Rent	Utilities	Education	Health	Shopping	Transport/Travel	Entertainment	Outside eating	EMI	Other
500.00	200.00	500.00	500.00	100.00	200.00	100.00	500.00	300.00	100.00



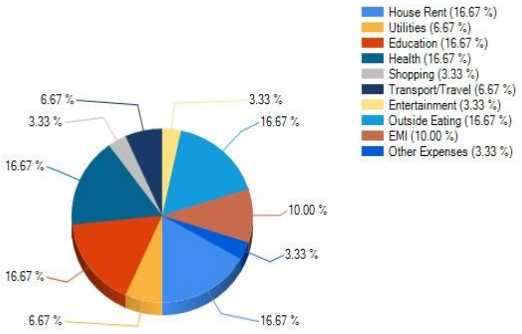
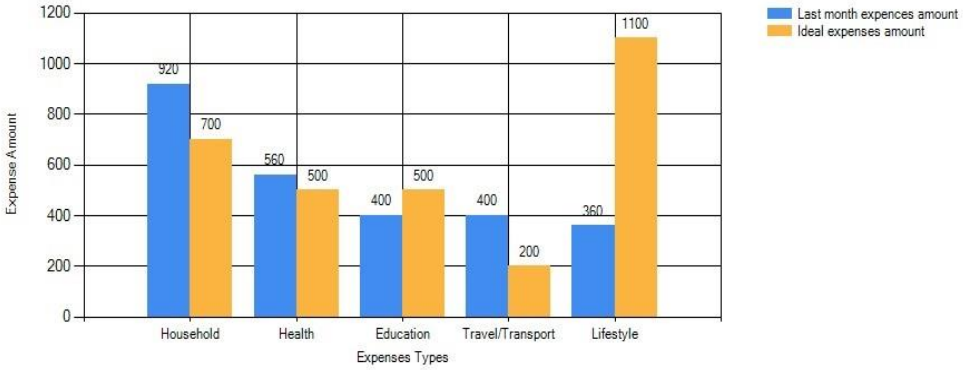


Fig. 15. Expense Report

Expense Type	Expenses
Household :	House rent, Utilities, Grocery shopping
Health :	Health related expenses
Education :	Expenses on education
Travel :	Transportation, traveling expenses
Lifestyle :	Entertainment, outside eating expenses, other EMI, other expenses

Expenses	Ideal expense percentage	Ideal expense value	Last month expense percentage	Last month expense amount
Household	23.00	920.00	17.50	700.00
Health	14.00	560.00	12.50	500.00
Education	10.00	400.00	12.50	500.00
Travel	10.00	400.00	5.00	200.00
Lifestyle	9.00	360.00	27.50	1100.00



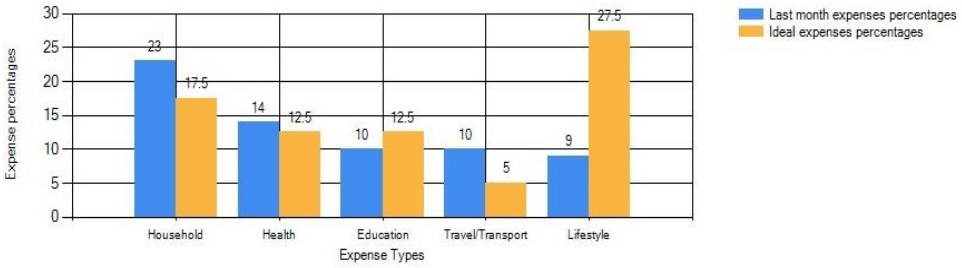


Fig. 16. Comparison Report of last month's expense data with ideal expense data

IX. Future Work

Following additions can be made to the system :

1. Provision for connecting directly to the database such that users will be able to directly access their account information.

2. Extending the chatbot to include support for audio or speech query evaluation.

X. Conclusion

The developed system tries to assist users in financial matters. It uses AI concepts to help users manage their finances in a smart and profitable way. It also provides answers to users' queries through the chatbot module.

Acknowledgment

We would like to take this opportunity to thank our professor, Dr. Feng Luo, for all the guidance and encouragement we needed. It has been a great learning process. We are really grateful to him for his kind support.

References

[1] Mr. Aniket Dole, Mr. Hrushikesh Sansare, Mr. Ritesh Harekar and Prof. Mithun Mane, "Intelligent Chat Bot for Banking System", in International Journal for Research in Applied Science & Engineering Technology (IJRASET), vol. 4, Issue IV, April 2016, ISSN : 2321-9653, IC Value : 13.98