

# Development of Intelligent Tutoring System Framework (Using Socratic Teaching Strategy)

**M. Tech. Project Report**

Submitted in partial fulfillment of the requirements  
for the degree of

**Master of Technology**

by

**Vikash Kumar**

**Roll No: 10305059**

under the guidance of

**Prof. Sridhar Iyer**



Department of Computer Science and Engineering  
Indian Institute of Technology, Bombay  
Mumbai

## Acknowledgments

I would like to sincerely thank *Prof. Sridhar Iyer* for his motivating support, guidance and valuable suggestions during this project work. His guidance has helped me to understand the concepts and he has been always motivating to me. I am extremely grateful to him for spending his valuable time to help me clarify my concepts. Working with him has been a great learning experience for me.

## **Abstract**

Here we have built a framework for Intelligent Tutoring System(ITS) which can support multiple teaching strategy. Most of the ITS support only one type of Strategy or limited to one subject only. Here we have discussed the framework of ITS which can support more than one teaching strategy and also independent from subject domains. ITS is a computer based instruction system which is used to teach a student with minimal interaction of human tutor. System has its own intelligence to make his decisions.

In this report, I have discussed ITS framework for a teaching strategy called Socratic Questioning. Socratic Questioning is teaching-learning strategy in which students learn through questioning and answering. Next question comes according to response of previous question. I have also done some literature survey like Wayang Outpost, SQL-tutor, Thermo-tutor etc to understand the architecture used in their system. Other 3 strategies Game Based Learning, Scaffolding, Guided Discovery are discussed by Praveen, Chandrapal Singh and Rajashekhar respectively. In the last section we have discussed how we have integrated these 4 different teaching strategies in one system.

# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	ITS . . . . .	5
1.2	ITS framework & Strategies . . . . .	6
1.3	My Work Summary . . . . .	6
1.4	Screenshot . . . . .	8
<b>2</b>	<b>Literature</b>	<b>9</b>
2.1	General component of ITS: . . . . .	9
2.2	Specific Case Studies . . . . .	10
2.2.1	Wayang outpost . . . . .	10
2.3	Strategies . . . . .	13
2.3.1	Socratic Questioning . . . . .	13
2.3.2	Scaffolding: . . . . .	15
2.3.3	GameBased Learning: . . . . .	15
2.3.4	Guided Discovery: . . . . .	15
<b>3</b>	<b>ITS Design Details for Socratic Teaching Strategy</b>	<b>16</b>
3.1	User Interface . . . . .	17
3.2	Steps . . . . .	18
3.2.1	Steps to be followed by instructor . . . . .	18
3.2.2	Steps to be followed by learner . . . . .	18
3.2.3	Steps to be followed by ITS . . . . .	19
<b>4</b>	<b>Overall Framework Of Our ITS</b>	<b>20</b>
4.1	Architecture . . . . .	20
4.2	Workflow . . . . .	20
4.3	Time Sequence Diagram . . . . .	21
<b>5</b>	<b>Implementation</b>	<b>23</b>
5.1	Description of Modules . . . . .	23
5.2	Databases . . . . .	26
5.3	Some Source code . . . . .	28
5.4	Challenges . . . . .	33
<b>6</b>	<b>Integration of all strategies</b>	<b>34</b>
6.1	Modules . . . . .	34
6.1.1	Common modules to all 4 strategies . . . . .	34
6.1.2	Non common modules . . . . .	35

6.2	Tables . . . . .	35
6.2.1	Common tables . . . . .	35
6.2.2	Non-common Tables . . . . .	36
6.3	Integration of modules and tables . . . . .	36
6.3.1	Module used for integration . . . . .	36
6.4	Content for Testing . . . . .	37
6.5	Screen-shot of demo . . . . .	43
6.5.1	For New User . . . . .	43
6.5.2	Learner . . . . .	44
6.5.3	For Instructor . . . . .	44
<b>7</b>	<b>Future work</b>	<b>56</b>
<b>8</b>	<b>Conclusion</b>	<b>57</b>
<b>A</b>	<b>Structure Of Tables</b>	<b>58</b>
<b>B</b>	<b>File Description</b>	<b>63</b>

# List of Figures

1.1	Home Page of ITS . . . . .	8
2.1	General Architecture Diagram for an ITS . . . . .	9
2.2	Architecture Diagram for Wayang Outpost . . . . .	11
4.1	Architecture Diagram For ITS . . . . .	20
4.2	Time sequence Diagram for Instructor in Our ITS . . . . .	21
4.3	Time sequence Diagram for learner in Our ITS . . . . .	22
5.1	Types of Forgeries . . . . .	27
5.2	Question Sequencing Table . . . . .	27
6.1	Registration Page For New User . . . . .	43
6.2	Student Login . . . . .	44
6.3	Student Home page . . . . .	45
6.4	Attempt Exercise . . . . .	45
6.5	Select Exercise . . . . .	46
6.6	Warrning Message when Dependency break . . . . .	46
6.7	1st Question . . . . .	47
6.8	Next Question . . . . .	47
6.9	Result of Previous Question . . . . .	48
6.10	Subtopic Completion . . . . .	48
6.11	Student Performance . . . . .	49
6.12	Instructor Login Page . . . . .	49
6.13	Instructor Home Page . . . . .	50
6.14	Add Course . . . . .	50
6.15	Add Topic . . . . .	51
6.16	Add Subtopic and Choose Strategy for Quiz creation . . . . .	52
6.17	Subtopic Dependency Creation Interface . . . . .	52
6.18	Socratic Quiz Creation Interface . . . . .	53
6.19	Next question relationship with OptionA . . . . .	53
6.20	Next question relationship with OptionC . . . . .	54
6.21	Quiz Creation Complete . . . . .	54
6.22	Add strategy . . . . .	55
6.23	Decide Priority Between Strategy . . . . .	55

# List of Tables

# Chapter 1

## Introduction

### 1.1 ITS

Intelligent tutoring systems (ITSs) are computer based programs that are designed to provide a tutor which knows two things what to teach and how to teach. On these two things all architecture, strategy, and logic work. Intelligent Tutoring Systems have the ability to reason about the domain and the learner. ITSs are used in many domains such as in traditional education, distance learning and training. It is a multidisciplinary area.

Now a days population is very large. The student-to-teacher ratio is not sufficient to make one-to-one interaction. **Why one-to-one interaction is necessary?** Education research has shown that the best learning environment is one-to-one with an expert human teacher[1]. In 1984 Bloom[4] proposed a theory that 98% students with private tutor performed better than classroom students. Therefore in classroom teachers are forced to suit their teaching to the average student. High achieving students may become bored due to slow pace and lack of challenge. So these students do not get what they deserve? And in second part low achieving students find the work difficult and never receive the level of attention they require. It means one-to-one teaching performed better than collaborative teaching. So the primary advantage of ITS is possibility of providing one-to-one tutoring. There are so many strategies which work on one-to-one teaching methodology.

**Now the question is why we need ITSs?** There are two motivating factor behind ITS development[4]. First is research need and second is practical use. The first generation of computer assisted education tools were called Computer-Aided Instruction (CAI) systems. One of the early examples of such a tutoring system is the system by Uhr in the year 1969[2]. This system generated problems on arithmetic and questions on vocabulary. But in this system there was problem this had no adaptation logic. It was not very successful. There is a difference between ITS and CAI , an ITS can provide individual attention to the student but CAI can't. ITSs provide the feedback according to cognitive profile of each student. Intelligent Tutoring Systems have the ability to reason about the domain and the learner. There so many ITSs come into existence like SQL tutor, C++ tutor, Algebra tutor, Auto tutor, Smart-tutor, Lisp tutor, Wayang Outpost tutor, thermo-tutor etc. They have their own architecture and adaptation logic but most of them has 4 main components domain model, student model, expert model and teaching model. Basically ITSs are of 2 types[10].



- **Standalone:** The ITS software is installed in learners system and learner can access the ITS when he runs the software.
- **Web-based:** The ITS software is installed and runs continuously on a system, which functions as server. The learners can access the ITS through Internet.

The web-based ITSs have following advantages over stand alone ITS.

- Learners are not constrained to use specific machines in their schools, and can access ITS from any location and at any time.
- Distributing software to learners and hardware/software compatibility problems are minimized.
- Updated versions of the ITSs will be available to learners.

ITSs are being used to teach various subjects. ITSs have been developed in geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, chemistry etc, subjects to help learners learn various subjects[2]. All these ITSs had one drawback they teach only one specific subject or limited to one strategy also. Most of these ITSs teach concepts directly. Teaching the concepts directly makes the learner passive, the learner memorizes the concepts and faces great difficulty to the real world problems. The learner should be able to discover the concepts and principles through interaction which increases the interest of learner, helps in remembering concepts for long time and teaches how to understand the concepts. Some subject or topics can be taught only with some specific teaching strategy. So an efficient ITS should worked on good teaching strategies. Above listed all ITSs use single teaching strategy for all topics of subjects. Thats why these ITSs are not more efficient.

## 1.2 ITS framework & Strategies

Goal of this research project is provide a generic framework which can support more than one teaching strategies and independent from subject domain.

For achieving this goal we have decided a pattern which will be independent from subject domain called multiple choice questions(MCQs).

After that we have studied different teaching strategies and picked 4 different strategies which can support our architecture. These strategies are

1. Socratic Questioning
2. Scaffolding teaching strategy
3. Guided discovery and
4. Game based Learning.

## 1.3 My Work Summary

Firstly I had studied architecture of different ITSs system like Smart-tutor, AutoTutor, SQL-Tutor, Wayang outpost, Mindspark. After studying these ITSs we have made our own framework. Then we started to research on different teaching strategy which is used to teach

students. Many of them come into pictures. Among them we have picked 4 strategies which can support our framework. These strategies are as follows

- Socratic Questioning : This strategy is implemented by me.
- Guided discovery : This strategy is implemented by RajaShekhar[17].
- Scaffolding : This strategy is implemented by Chandrapal[19].
- Game based learning : This strategy is implemented by Praveen Dhanala[19].

Here I am going to give brief introduction of my strategy called socratic Questioning. Remaining strategies have discussed by my other teammates *Chandrapal singh, Raja Sekhar and Praveen* respectively.

## **Socratic Questioning**

It is an approach in which teaching-learning is performed in the form of question and answer. It is a kind of series of questioning in which an original question is spitted into more than one low level question. It is just like bottom up approach. In this strategy we start from the question which student or learner knows and goes to our target questions which we want to teach him. Socratic questioning is basically a dialogue conversion between two teacher and student. First, instructor starts the question and student responses. In return instructor reformulates a new question according to the response given by student. Questioning and answering is structured systematically to reach at ultimate goal. This strategy looks like simple MCQs pattern, but it is not. In simple MCQs only series of questions are available but in this strategy there is a relationship between next question and option chosen by learner.

Firstly I have implemented my strategy individually and in last integrated it with our system. I have also collected questions from different resources which supports my strategy. Right now our ITS supports only multiple choice questions.

## 1.4 Screenshot

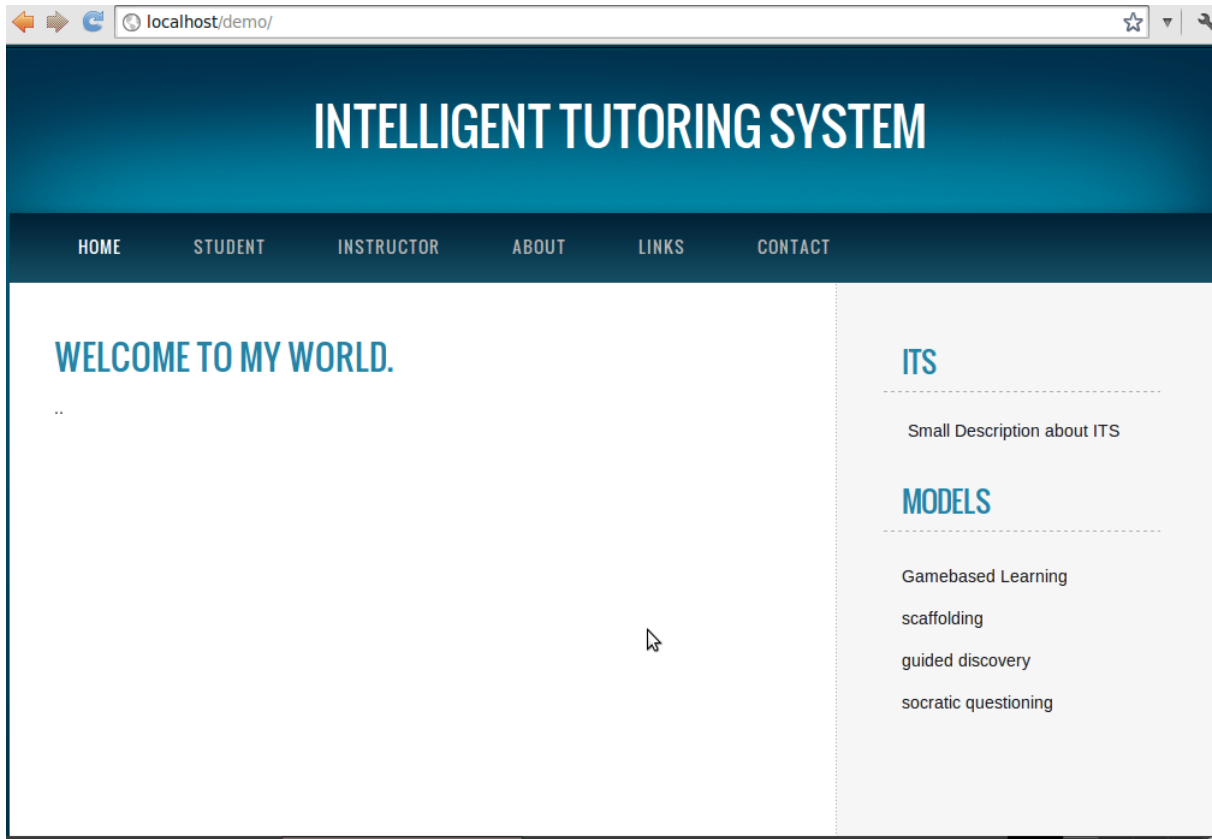


Figure 1.1: Home Page of ITS

# Chapter 2

## Literature

### 2.1 General component of ITS:

There are 4 basic components student model, teaching model, domain models and user(student/teacher) interface model. It is a top level architecture diagram of ITS.

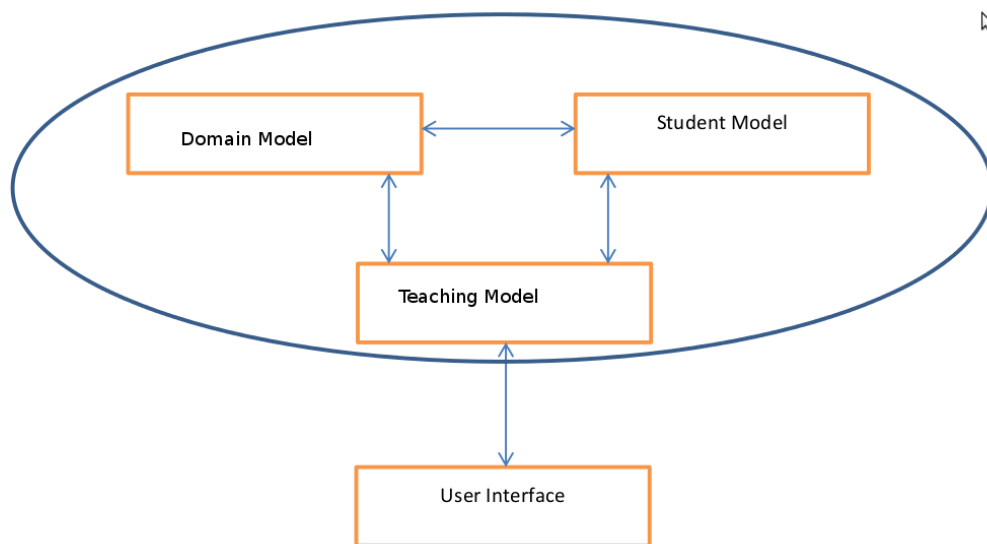


Figure 2.1: General Architecture Diagram for an ITS

- **Domain model** The Domain model represents the content knowledge that the student is acquiring. This module is at the heart of an ITS and provides the basis for interpreting student actions. It has a flexible structure and the different courses from different domains can be represented by using the same representation scheme. It has two parts first part contains course structure (CS) and topic structure (TS) and other part contains quiz module on the basis of different teaching strategy.
- **Student Model** Student model is a most basic component of ITS system. Without student model ITS cannot be completed. It refers to the dynamic representation of the emerging knowledge and skill of the student. This model is connected to all other models.

It contains

- Student’s profile: It contains the personal information of student like name, student id, age, sex, email id.
- Student’s performance table: It contains the information about student performance question wise. It stores the response of each question given by student.
- Student’s progress status: It contains the final student’s performance topic wise. After covering each topic control manager see student’s performance table and update the student’s status. Before going to another topic control manager first check the Student’s progress status and take steps according to this.

In student model we are taking only one criteria to make student performance table, is his/her program solving ability. After giving his/her response on each question we make student’s performance table and student’s progress status.

- **Teaching Model** This model decides all knowledge base of ITS. In this model all work is done by instructor. First work is to decide strategy. Here we are using socratic strategy. Instructor work is as follow
  - Design Course Structure: In our ITS data structure used for course design is **tree**. In this course name remains on the root and topic remains on leaf. Leaf node will be the lowest unit of tree.
  - Design Topic Structure: Data structure used for topic design is **graph**. Because if there are n topics so there is a possibility for a particular topic prerequisite is some another topic. So we have to make **topic dependency graph**. It will be maintained by instructor. He will decide the dependency among the topics.
  - Design Rule structure: In this part instructor have to decide cutoff level for a particular topic. After achieving cutoff level student can attempt another topic.
- **User(student/instructor) interface model**The user interface module is the communicating component of the ITS which controls interaction between the user and the system. It works in both directions, it translates between the system’s internal representation and an interface language that is understandable to the user.

## 2.2 Specific Case Studies

### 2.2.1 Wayang outpost

Wayang Outpost[3] is a web-based intelligent tutoring system. Wayang Outpost was designed as a supplement to high school geometry courses. Its orientation is to help students learn to solve math word problems typical of those on high stakes achievement tests, which may require the novel application of skills to tackle unfamiliar problems, as well as the need to work quickly due to the time constraints. Wayang Outpost provides instruction via a web site. The student begins a session by logging into the site and receiving a problem. Each math problem is presented as a Flash movie. If the student answers incorrectly, or requests help, the teacher

character provides step-by-step instruction and guidance in the form of Flash animations with audio. For example, on a geometry problem, the student might see an angle with a known value rotate and move over to the corresponding angle with an unknown value on a parallel line, thus emphasizing the principle of correspondence. The explanations and hints provided in Wayang Outpost therefore resemble what a human teacher might provide when explaining a solution to a student, e.g., by drawing, pointing, highlighting critical parts of geometry figures, and talking, in contrast to previous mathematics ITS which relied heavily on screen-based text.

## Architecture:

In Wayang Outpost, there is a centralized database. Every Data like problem, its solution, hint, student work, his performance are stored in a centralized database. From this data, the system makes inferences on an ongoing basis to select problems at the appropriate level of challenge, and chooses hints that will be helpful for the student.

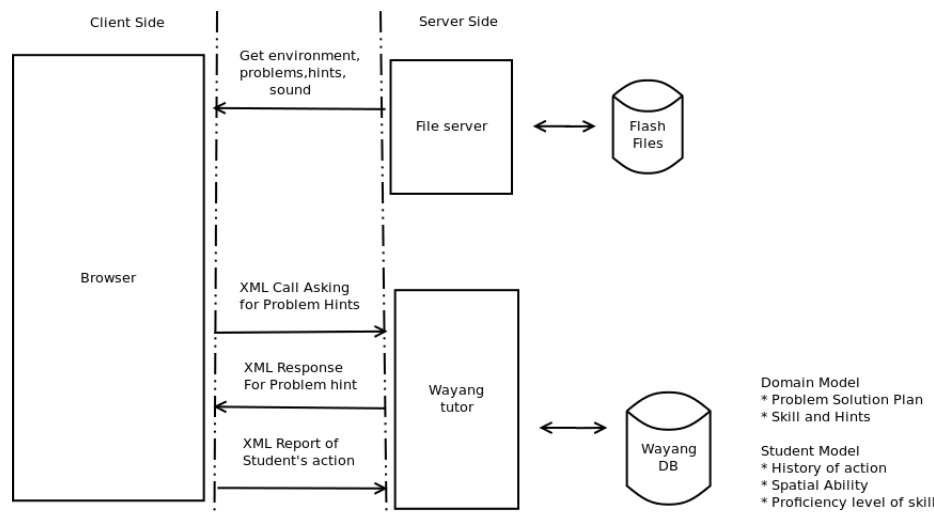


Figure 2.2: Architecture Diagram for Wayang Outpost

In Wayang Outpost architecture there are three part *client side, server side and interface*. Client side is basically a browser which is made of flash plug-in. It is the start point for users after login they directed to another side. Server side is completely database where all data are stored. It has some models like domain models[3], student models and support models. Domain model contains problem, its solution. It also contains the hints to solve the problem. Student model contains the information of student, its performance chart, its level of skills. Support model contains animation, sound effect etc to make it effective.

## Help Logic in Wayang Outpost

There are two types of hints:

- First hint provides a computational and numerical approach
- Second provides spatial transformations and visual estimations.

Hints are generally provided according to students cognitive profile.

Wayang Outpost doesnt trace each step of the students solution, because this would be too expensive to implement for so many different problems. It uses the concept of data-centric approach with Bayesian Networks to categorized 3 types of student who already knows a skill , is learning a skill and is not learning skill? The tutor observes the hints requested by the student to reach the solution .if student requests very less hint and give correct solutions (not all) it means he already knows a skill. If student makes correct answer after using hints it means he is learning a skill but A student did not learn the skill when the student had a low score for the skill at the beginning of the session and there is not a significant improvement in the number of correctly solved problems involving the skill from the beginning to the end of the session.

## Evaluation strategy and Conclusion:

In Wayang Outpost there are two help strategies one is spatial and second is computational. Now the question was which one is better. Is the spatial approach effective for students with high spatial ability (because it capitalizes on their cognitive strengths) or for those with low spatial ability (because it compensates for their cognitive weaknesses)? Is the computational help better for students with high retrieval speed of mathematics facts from memory, or is it better for students with low speed of math fact retrieval? Given a student with a specific cognitive profile, what type of help should be selected for him/her? For getting the data experiment is performed in 2 group. One group can use only spatial help and another can use only computational help. In this experiment Students used Wayang Outpost for about 2 hours. They were given a survey after using the tutor, to evaluate their perceptions of help and their willingness to use the system again.

Learning was measured with a Learning Factor that described how students decrease their need for help in subsequent problems, on average. Ideally, students would need less help as time goes by, after accounting for the difficulty of each problem seen. This learning measure describes the percentage of help requests the student made, minus what is expected for the problem. After collecting all data they conclude[?] that-

- There was significant difference in the number of correct responses, females having significantly less correct answers than males. At the same time, females spent more time in each test item.
- If both abilities(spatial and computational) are low, computational help was best.
- If both abilities are high, spatial hints should be provided, as they yield higher learning.
- Girls are specially motivated to use the adventures of the system comparison to boys.

Some of other tutors which i have studied are :-

- **SQL Tutor**[10]: It is an ITS, which as the name suggests is to teach SQL. Artificial Neural Network (ANN) was used in SQL- Tutor for decision-making. In this an agent was available for the selection an appropriate problem from the database it also analyses the student's status.

- **Thermo-Tutor**[16]: An ITS that teaches thermodynamic cycles in closed systems (the first law of thermodynamics). It assumes that the students have already learned the basic concepts in lectures, and therefore it provides lots of problem-solving opportunities. Performance of this tutor was good. But it requires high mental effort. It means student has to participate very actively. One another disadvantage of this tutor was it took a while to realise how to use it.
- **Auto tutor**: It is a complex system that simulates a human or ideal tutor by holding a conversation with the learner in natural language[9]. It presents a series of challenging questions (or problems) that require approximately a paragraph of information to answer correctly. This system performs good when the shared knowledge between the tutor and learner is low or moderate. If the shared knowledge is high, then it doesn't perform well because it requires very high precision to satisfy both instructor and learner.

But there is a **one common problem** with most of the tutor, they are designed only for one strategy. Their architecture supported only one teaching methodology. ITSs have been developed in geography, circuits, medical diagnosis, computer programming, mathematics, physics, genetics, and chemistry etc, subjects to help learners learn various subjects. All these ITSs are made to teach only one specific subject. Here in our ITS framework we are trying to remove this problem.

We are going to build a system *which can support at least 4 different strategies*. We have also planned to make our ITS framework generic (it means independent from subject domain and can support more than one strategy).

Our ITS provides a problem-solving environment, in which students are given many opportunities to practice their skills. For example in scaffolding we are providing hint for a student and in guided discovery level of questions are arranged. It collects the information about student's knowledge and also update this knowledge according to student's action. In our ITS there is one module called controller who manages adaptation logic. In our ITS adaptation is applied in terms of feedback, selecting next question, change the level of student according to student's performance, decide which topic will be next, student has fulfilled the prerequisite of this topic or not.

We have first developed an individual ITS system for each strategy. Now we decided to integrate these all 4 different strategies in one ITS.

## 2.3 Strategies

### 2.3.1 Socratic Questioning

#### Motivation

The Socratic Questioning Method is an approach to learning and thinking that draws out knowledge from learner using questions. It has a number of beneficial aspects, it is primarily used to teach students how to think critically through a thoughtful examination of ideas and issues in any discipline. It is not limited to a specific area of inquiry, like law or philosophy.



Teachers can employ Socratic pedagogy in any domain of thought, including the "hard sciences," like math, physics, and astronomy.[Rudd 1977].

In Socratic Questioning instructor starts from a questions and get response from learner. According to his/her response he asks another question. In this process learners are forced to engage with the subject matter. There is no way to get around it. Here conversation is done through question and answer that yields a deeper understanding of subject matter. Both instructor and learner are engaged with subject matter in systematic way which is called Systematic engagement[8]. Systematic engagement is a combination of active engagement and systematic practice.

During Socratic questioning, First teacher split the big question in many smaller parts and he starts from a small question and reach to final answer of big question by combining all the responses given by student. Here teacher is a model of critical thinking who takes students' response, probes their understanding, and shows interest in their thinking. The teacher waits for response after putting the questions. The teacher creates a learning environment, in which he challenges the students. The teacher also tries to make sure that student is comfortable during answering the question. After getting each response teacher asks next question according to his response. The overall purpose of Socratic questioning, is to challenge accuracy and completeness of thinking in a way that acts to move students towards their ultimate goal.

Socratic questioning helps students to think critically by focusing explicitly on the process of thinking. During disciplined, carefully structured questioning, students must slow down and examine their own thinking processes (i.e., reflective thinking). According to Kathleen Cotton (MAY 1988)[6], disciplined questioning in the classroom can achieve the following teaching and learning goals:

- Model scientific practices of inquiry
- Support active, student-centered learning
- Help students to construct knowledge
- Help students to develop problem-solving skills
- Improve long-term retention of knowledge

## **Pedagogy of Socratic Questioning Teaching Strategy:**

When teacher uses Socratic questioning in teaching, his main goal is estimate the student thinking, to determine the extent of student knowledge on a given topic, issue or subject. In teaching, teacher can use Socratic questioning for at least two purposes[7][8]:

- To deeply probe student thinking, to help students begin to distinguish what they know or understand from what they do not know or understand (and to help them).
- To foster students' abilities to ask Socratic questions, to help students acquire the powerful tools of Socratic dialogue, so that they can use these tools in everyday life (in questioning themselves and others).

Socratic questioning illuminates the difference between systematic and fragmented thinking. It teaches students to dig beneath the surface of their ideas. It also teaches the value of developing questioning minds in cultivating deep learning. When teacher uses this strategy in classroom teaching then he followed a particular type of questioning[7].

1. Getting students to clarify their thinking  
e.g., Why do you say that?
2. Challenging students about assumptions  
e.g., Is this always the case?
3. Evidence as a basis for argument  
e.g., Why do you say that?, Is there reason to doubt this evidence?
4. Alternative viewpoints and perspectives  
e.g. What is the counter argument for?. Can/did anyone see this another way?
5. Implications and consequences  
e.g. But if what happened, what else would result?
6. Question the question  
e.g. Why was that question important? Which of your questions turned out to be the most useful?

## **Steps in Socratic Questioning Teaching Strategy**

1. Ask question
2. Wait for response
3. Take response
4. Ask next question based on response

### **2.3.2 Scaffolding:**

This strategy is implemented by ChandraPal Singh[19].

### **2.3.3 GameBased Learning:**

This strategy is implemented by Praveen[18].

### **2.3.4 Guided Discovery:**

This strategy is implemented by RajaShekhar[17].

# Chapter 3

## ITS Design Details for Socratic Teaching Strategy

We have built an ITS framework which is based on socratic teaching strategy. In this learner will learn through Questioning and Answering. Questions will be multiple choice questions. Each questions have 4 options among them one will be correct. Next questions will come in front of learner according to his response given on previous questions. That means there is a relation between the next questions and response given by previous questions. Here initially instructor will add questions, decide sequencing of questions. After login, the learner will follow the instructions given by our ITS and participate the quiz. And finally ITS will provide feedback according to performance of learner in given quiz.

**Example :** There are 5 MCQ questions.

1. Que: `int a=5/2;` the value of a will be
  - (a) 2.5
  - (b) 0
  - (c) 2
  - (d) Error
2. Que: Variable a is integer type then what is the value of a in the given expression `a=30*100+2768;`
  - (a) 3000
  - (b) 30276
  - (c) 3276.8
  - (d) Out of range
3. Que: If data type of a variable is integer than it return
  - (a) Always zero
  - (b) Depend upon assigned value
  - (c) Always integer
  - (d) Can't Say

4. Que:If int a=3.5;It will return a=
- (a) 3
  - (b) 3.5
  - (c) 0
  - (d) we cannot store float value in integer type variable.
5. Que: What is the range of data type integer?
- (a) -32768 to 32767
  - (b) 0 to 65535
  - (c) -128 to 127
  - (d) NONE

In the first question correct answer is (C) i.e 0. If anyone gives correct answer then system will give him question number 2. But if anyone gives option (A) i.e 3.5 It means he has some confusion on about datatype so system will give him question number 4. So after seeing question 4 he can understand the concept of int and float. If anyone chooses option (B) i.e is 0 then system will provide question number 3. All these sequencing will be done at the time of content creation.

### 3.1 User Interface

The interface provided should be interactive to teacher and learner.It should be user friendly.  
**Interface to instructor**

- Login interface: It is for authentication purpose.
- Course interface: It will provide functionality to instructor to create, delete and modify any course.
- Topic interface:It will provide functionality to instructor to create, delete and modify any topic
- Topic Dependency interface: Make dependency between topics.
- Questions interface: Each question with 4 options and 1 right answer
- Question sequencing interface: Which question will be next
- Feedback interface: Instructor can set particular comment for giving any feedback to students.
- Edit option to all above fields

## 3.2 Steps

### 3.2.1 Steps to be followed by instructor

The instructor inputs the following things to the ITS.

- If course or topic is present in ITS system
  1. Select Strategy: First instructor will decide with which strategy he is going to create a quiz.
  2. Select course
  3. Select topic
  4. Select subtopic
  5. Establish topic dependency
  6. create quiz
  7. Put questions, multiple options and right answer for the corresponding question.
  8. Maintain sequencing of questions (i.e make relationship with question and various options).
  9. Types of Feedback (Feedback per question or Final feedback or both)
- If course or topic is not present in ITS system
  - Create course name, course\_id, course description
  - Create topic name, topic\_id, topic description
  - Create subtopic name, subtopic\_id, subtopic description

After creating course or topic then follow the above steps.

### 3.2.2 Steps to be followed by learner

1. Select course
2. Select topic
3. Select subtopic
4. Attempt Quiz
5. Participate actively when called upon.
6. Follow feedback carefully.

### 3.2.3 Steps to be followed by ITS

- After submitting course name, topic name and subtopic name by student system will check this student\_id is attempting this subtopic first time or not. If it is first time then system makes its entry in database.
- In second step system goes to strategy\_seuencing\_table where it gets the strategyId for presenting the question.
- According to Response given by learner system presents next question to learner.
- Give feedback to learner.
- Result generating.
- Maintain student performance Table.
- Apply adaptation logic.

# Chapter 4

## Overall Framework Of Our ITS

### 4.1 Architecture

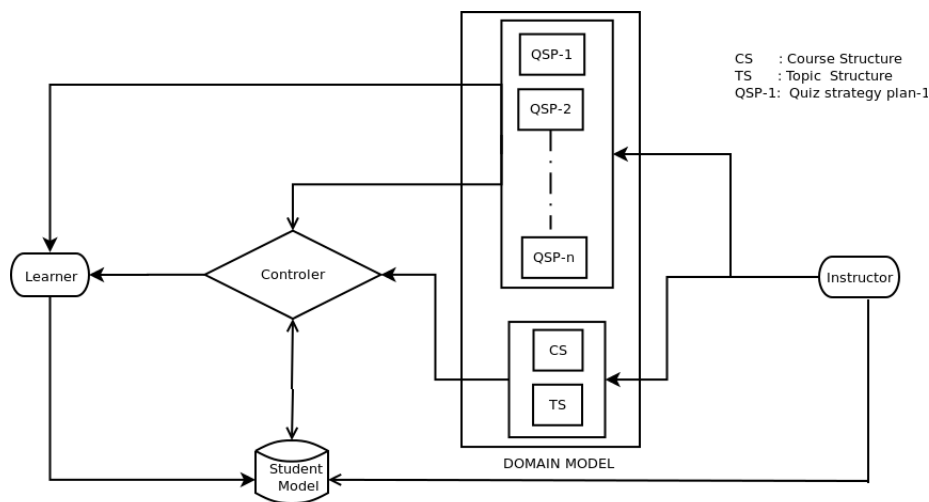


Figure 4.1: Architecture Diagram For ITS

This is overall structure of our ITS framework. Here there are 4 parts learner, instructor, controller and databases. Instructor plays very important role in this system. There are so many tasks he performs like course creation, topic creation, subtopic creation, strategy sequencing for particular subtopic, decide adaptation logic for strategies etc.

### 4.2 Workflow

Work flow of our ITS can be categorised in 3 ways.

- **With respect to new user:** When a new user comes then it has to first register himself after that he can perform his task according to his login type (like instructor or learner).
- **With respect to instructor:** Instructor plays very vital part in this system. Instructor has first required to login with authenticated user-name and password. After authentication instructor can perform his task like add, delete, modify courses, topics and subtopics.

Instructor provide question for appropriate subtopic with appropriate teaching strategy. He will have to choose the strategy by which he want to teach the student. Instructor then can upload required resources for specific subtopic. Instructor can see the details of student like his name, level and his progress report. After having information about the progress of students instructor can provide more resources and can switch to the new teaching strategy also.

- **With respect to learner:** Learner first required to login with authenticated user-name and password. Then student can see the courses offered and can select the subtopic of the course which he want to learn. Student can learn from the resources if available and after that try to attempt the questions given by teacher. If student is not able to give answer and tick wrong option then ITS will prompt for hint and students level will decrease by one point. If student is not able to achieve the threshold of marks then ITS will switch that student to new strategy and provide him questions from other available teaching strategy.

## 4.3 Time Sequence Diagram

- Time Sequence Diagram For Instructor

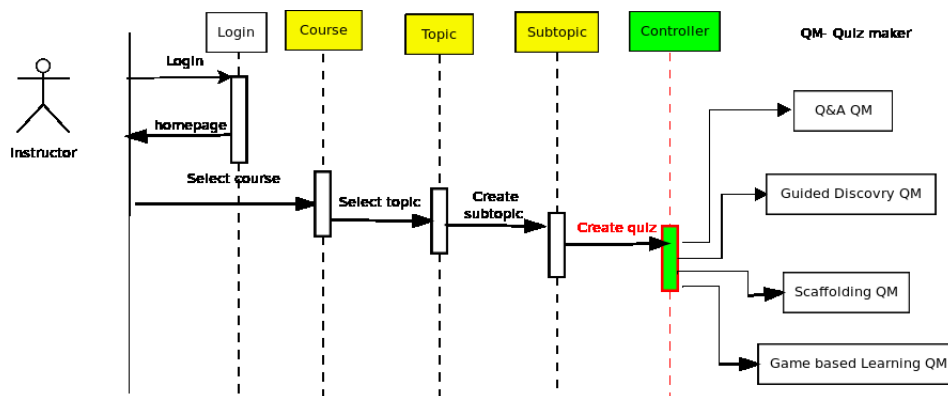


Figure 4.2: Time sequence Diagram for Instructor in Our ITS

In this time sequence diagram there is one module called controller, Who first asks instructor to specify the strategy. When instructor specifies his strategy then it provides an interface according to selected strategy. Because each strategy has different requirements. For example Scaffolding needs hints module, Guided discovery needs more than one level of questions. So controller manages these things. According to strategy specified by instructor it calls quiz-maker module of each strategy.



• Time Sequence Diagram For Learner

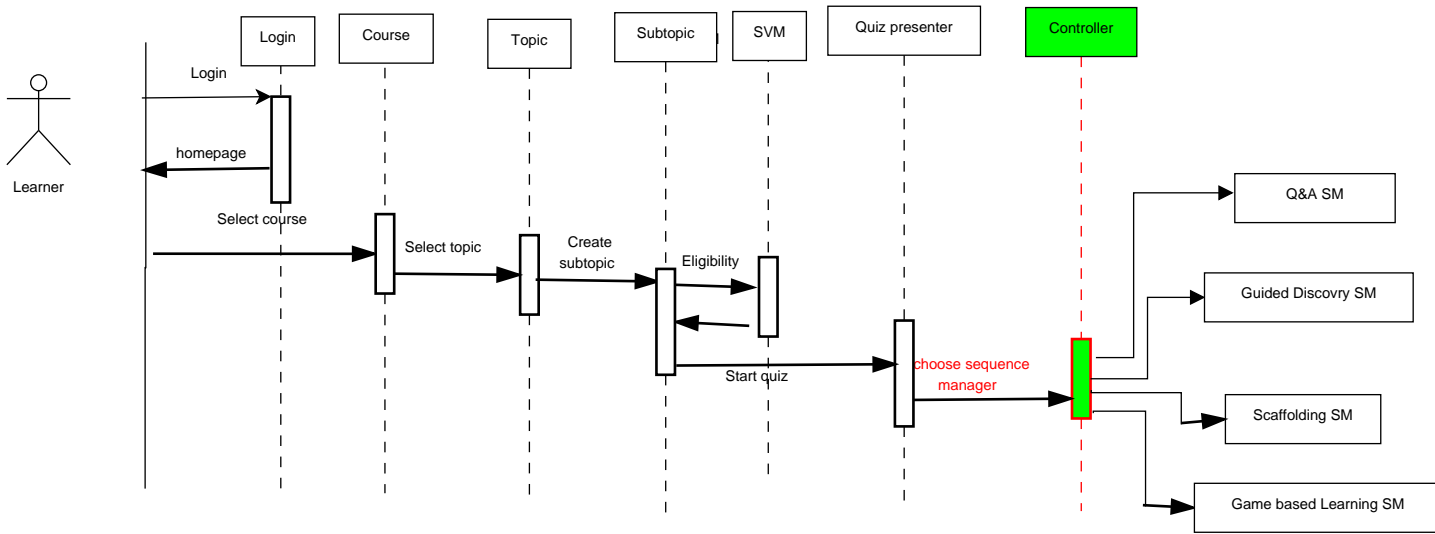


Figure 4.3: Time sequence Diagram for learner in Our ITS

This shows how learner will interact with system. After login learner will have some options like attempt exercise, see his level, see his performance subtopic wise etc. For example if he decides to attempt exercise then he has to select course, topic and subtopic respectively after that system will present question. During attempting exercise he has to follow instruction given by system like use hints.

# Chapter 5

## Implementation

### 5.1 Description of Modules

This section explains the different modules of our ITS system. What is the functionality of each module and how these modules interact with other modules.

1. **GUI module:**

- **Functionality:** Display interface to users.(student/teacher)  
Handles all actions like displaying forms, inserting, deleting, updating elements (course, topic, questions, options etc) to form.

2. **Sequencing Module:**

- **Functionality:** It Handles relationship between questions and option in question database. In the sequencing table instructor stores each option of any question with next question.
- **What other modules it uses**  
Database handler Module: From the database it finds the next question.

3. **Strategy Sequence Module:**

- **Functionality:** For each subtopic and studentId this module first finds the highest priority order strategyid and gives to quiz maintenir module for presenting the questions.
- **What other modules it uses**  
Database handler Module: From the database it finds the question according to strategy.

4. **Input Validation module:**

- **Functionality:** Validates the user input data. Checks whether mandatory fields are filled or not and performs validation of other constraints imposed by administrator.

5. **Student module:**

- **Functionality:** Handles all learner actions. It co-ordinates the modules dealing with learner like learner's registration, learner's profile information etc.

#### 6. **Authentication module:**

- **Functionality:** It validates user is valid or not. Check user\_id and password and compare it with stored database. For student authentication It uses student module to authenticate.

#### 7. **Topic Validation module:**

- **Functionality:** It validates the topic selected by student. Student has completed prerequisite topics or not. It uses topic Structure Module for validating it.

#### 8. **Subtopic Validation module:**

- **Functionality:** It validates the subtopic selected by student. Student has completed prerequisite subtopics or not. It uses subtopic Structure Module for validating it.

#### 9. **Quiz Maintainer Module:**

**Functionality:**

- It starts quiz according to status of student's progress table. i.e At this stage how many questions student has solved? when student comes 2nd time for same topic then it will start from there.
- This module also interacts with GUI module, Database handler module, Sequencing module and Student module. This module with the help of sequencing module gets next question id from sequencing table and presents it to learner. Database handler module helps this module to pick a question from the database.

#### 10. **Evaluation module:**

- **Functionality:** It checks the option (response) given by student is right or wrong and also updates the student's progress table. It generates results after completion of each session or topic by student and updates the user knowledge base.
- What other modules it uses
  - Database handler Module: For finding correct answer.
  - Student Module: For updating student's knowledge

#### 11. **Database handling module:**

**Functionality:** Handles basic database operations like insert, delete, update and select etc. Here we have mainly 5 databases.

- **Question database:** Stores questions, its options and its right answer.
- **Sequencing database:** Stores the information about next question corresponding to given option.
- **Domain database:** Stores course information, topic information and subtopic information.

- Student database: Stores student's personal information username, password, student performance table, student's status for particular topic or subtopic i.e it is complete or not.
- Logic database: Stores administrative rules and regulation like the minimum percentage of correct answers to promote a learner to next level.

## 12. Feedback module :

- Functionality: It provides feedback to student when it gives response to any question.
- What other modules it uses
  - Student Module: After each session it provides feedback on the basis of student progress table
  - Evaluation Module: It takes result of each response from EM and provides feedback to student.

## 13. Logic module :

- Adaptation logic module  
Functionality: It decides when system has to change strategy for particular topic. If learner has interacted all strategies then this module also decides which strategy will be better for the learner.
- Control logic module  
Functionality: Here instructor sets some cut-off value for qualifying some topic. This modules check the cut-off value then pass to another topic.

## 14. Topic Structure Module :

- Functionality: It maintains topic structure (topic dependency graph) provide by instructor.
- What other modules it uses
  - Input validation Module: All fields for particular topic is filled or not
  - GUI Module: This module helps for interaction.
  - Database handler Module: It stores the information of topic.

## 15. Quiz maker Module:

- Functionality: It provides an interface to teacher to make quiz for particular topic.
- What other modules it uses
  - Input validation Module: All fields for particular course is filled or not'
  - Database handler Module: It stores all attributes question. It also stores the information of quiz.

16. **Status Module:** Functionality: It provides the complete report of students course wise or topic wise. Instructor can see the status of any students.

17. **Logout module:** Logs user status when user logs out.

## 5.2 Databases

I have used following tables to implement the socratic teaching strategy for our ITS. The complete structure of all tables are given in the **Appendix A**. Brief description of these tables are:

- **course\_table** : It contains the information regarding courses like Course name, Course Id and little description of course.
- **login\_table**: It contains the information like login name, login id, password and type of users (learner/instructor). Authentication module takes help of this table to authenticate the user.
- **logtable**: It contains the information regarding student performance question wise.i.e response of each question is stored in this table. After completion of each subtopic on the basis of this this table student performance table is maintained.
- **parsed\_table**: It is a temporary table which is used to parse all the questions in the question table. It helps to track that option of each question is related to next question or not.
- **question\_table**‘: It contains information regarding questions like question description, its 4 options, correct option and feedback given by instructor.
- **student\_performance\_table**: This table contains the record of the student’s performance for a particular subtopic on the basis of strategy. It also stores the percentage\_performance for this subtopic and strategy. On the basis of this percentage\_performance system decide to change the strategy or not. If percentage\_performance is below cutoff level then system changes the strategy.
- **student\_progress\_table**:This table keep the record of students progress. Information, like how many subtopic he has done and what was his result in these subtopics is stored in this table. This table stored the result (Marks obtained by student) by which ITS decides that this student is required to switch the strategy or not.
- **student\_subtopic\_status**: It contains the information regarding status of subtopic for a particular student. i.e subtopic is completed or not.
- **subtopic\_table**:It contains the information regarding subtopic like subtopic name, Course Id and little description of subtopic.
- **topic\_table**:It contains the information regarding topics like topic name, topic Id and little description of topic.
- **strategy\_table**:It contains the information regarding courses like Course name, Course Id and little description of course.

- **strategy\_sequencing\_table**: It contains the information regarding priority of strategy for a particular subtopic. Here in the table 1,2,3,4 values shows the priority order of

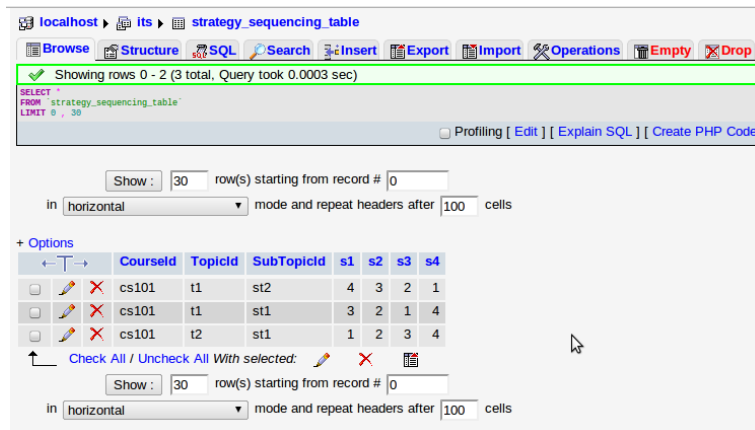


Figure 5.1: Types of Forgeries

strategies. If there is any subtopic which can be teach by more than one teaching strategy then instructor will decide the priority order of each strategy. In this case strategy selector will choose the first priority strategy and present question associated with this strategy. This table also helps when system wants to change strategy. Because if student does not perform well with any strategy then system pickes another strategy on the basis pf priority order.

- **sequencing\_table**: It contains information regarding next question associated with option of previous question. Here in the table '9999' value is used. If 9999 comes then quiz

CourseId	TopicId	SubtopicId	QuestionId	ChosenOption	NextQuestion
cs101	t1	st1	8	C	9
cs101	t1	st1	8	D	9999
cs101	t1	st1	9	A	9999
cs101	t1	st1	9	B	11
cs101	t1	st1	9	C	9999
cs101	t1	st1	9	D	10
cs101	t1	st1	10	A	11
cs101	t1	st1	10	B	11
cs101	t1	st1	10	C	11
cs101	t1	st1	10	D	11
cs101	t1	st1	11	A	15
cs101	t1	st1	11	B	17
cs101	t1	st1	11	C	16
cs101	t1	st1	11	D	17
cs101	t1	st1	12	A	19
cs101	t1	st1	12	B	9999
cs101	t1	st1	12	C	9999
cs101	t1	st1	12	D	19
cs101	t1	st1	13	A	14
cs101	t1	st1	13	B	14
cs101	t1	st1	13	C	14
cs101	t1	st1	13	D	14
cs101	t1	st1	14	A	15
cs101	t1	st1	14	B	15
cs101	t1	st1	14	C	15
cs101	t1	st1	14	D	16

Figure 5.2: Question Sequencing Table

maintainer module searches next questionid associated with correct option. And present this question to student.

## 5.3 Some Source code

- **Question Sequencing Module** : This module is maintains relationship between next question and option of previous question. For implimenting this module i have used **Breadth first traversal** algorithm.

```
#####          Breadth First Graph Traversal Algorithm

if($txtQuestionDes==''||$txtOption1==''||$txtOption2==''||$txtOption3==''||$txtOpti
||$txtCorrectAns=='')
{
    if($txtQuestionDes==''){echo "Enter Question";}
    elseif($txtCorrectAns==''){echo "Select Correct Answer";}
    else{echo "Option Field Is empty";}

    {header('Location:'.old_socarticquiz.php');}
}

$query3= "SELECT * FROM parsed_table WHERE (TeacherId='$instructorid') LIMIT 1";

$result = mysql_query($query3,$conn) or die('table is empty' . mysql_error());
$row    = mysql_fetch_assoc($result);

if($row)
{
    echo "First complete your previous session";
    $p_questionId = $row['QuestionId'];
    $parsed_Id=$row['Parsed'];

    if($parsed_Id < 5 && $parsed_Id!='')
    {
        while ($parsed_Id == 4)
        {    # Delete first row and select another row

            $query4= "DELETE FROM parsed_table where (QuestionId='$p_questionId')";
            $result = mysql_query($query4,$conn) or die('Could not connect.....:30000 ' . mysql

            $query3= "SELECT * FROM parsed_table WHERE (TeacherId='$instructorid') LIMIT 1";

            $result = mysql_query($query3,$conn) or die('table is empty2' . mysql_error());
            $row    = mysql_fetch_assoc($result);

            $parsed_Id=$row['Parsed'];
```

```

}

if($parsed_Id == 0) {header('Location:'. 'nextquestion4a.php');}
elseif($parsed_Id == 1) {header('Location:'. 'directnextquestion4b.php');}
elseif($parsed_Id == 2) {header('Location:'. 'directnextquestion4c.php');}
elseif($parsed_Id == 3) {header('Location:'. 'directnextquestion4d.php');}
else { echo "something is wrong";}

}
}

else
{

    if($txtQuestionDes!='')
    {

        $query = "INSERT INTO question_table(StrategyId,CourseId,TopicId,SubtopicId,QuestionID,QuestionText) VALUES ($txtCo, $txtCo, $txtCo, $txtCo, $txtCo, $txtCo)";
        mysql_query($query,$conn)or die('Could not connect:1001 ' . mysql_error());

        # Find Questio_id from question table

        $query1= "SELECT QuestionID, TopicId FROM question_table WHERE (CourseId='$txtCo)";

        $result = mysql_query($query1,$conn) or die('Could not connect.....:212 ' . mysql_error());
        $row = mysql_fetch_assoc($result);
        $Que_Id=$row['QuestionID'];
        # Add Question to database parsed_table

        $query2 = "INSERT INTO parsed_table (TeacherId,CourseId,TopicId,SubtopicId,QuestionID,QuestionText) VALUES ($txtCo, $txtCo, $txtCo, $txtCo, $txtCo, $txtCo)";
        mysql_query($query2,$conn)or die('Could not connect:213 ' . mysql_error());
        }

        {header('Location:'. 'nextquestion4a.php');}
}

```



In this code I have used one table called `parsed_table` which acts like queue. When instructor adds question than its `question_id` is stored in this table. There is one filed in this table called "Parsed" which helps to record this `question_id` is parsed(or traversed) or not. For each option of this `question_id` there should be one question. When instructor adds any question for corresponding option then `parsed` values of this `question_id` is increased. Because there are 4 options for each question so maximum value of `parsed` reaches 4. When the value reaches 4 it means instructor has filled next question for each option. After that system deletes this `question_id` from the `parsed` table and takes next question from the `parse.table`.

## Topic & Subtopic Dependency

In socratic questioning topics and subtopics are related with each other. Each topic and subtopic(except first one) is somehow related with one or many topic and subtopics. Because for each topic and subtopic there are some pre-requisite. When a student attempts any topic and subtopic than our system first checks its prerequisite subtopics. If pre-requisite condition is fulfilled then student can attempt this topic and subtopic. For maintaining the prerequisite relationship among topics and subtopics, we store topic dependencies in graph structure. Whenever instructor changes any dependencies then the system checks cycle in the graph.For checking the cycle we have used depth first algorithm.

```
#####          Depth First Graph Traversal Algorithm
$i=1;
$id= mysql_result($result,0,"id");
$arr=array($id),$full=array(),$full_count=0,$cycle=0;
#if cycle=1 then cycle is present in given graph
while ($full_count<$num && $cycle==0) {
    if(array_diff($arr,$full)){
        $query2='SELECT depends_on FROM dependency where topic=\'\'.$id.\'\'';
        $stored_data=mysql_query($query2) or die('select.. query failed');
        $stored=mysql_result($stored_data,0,"depends_on");
        $var=explode(" ", $stored);
        if($var[0]=="") {
            $full[$full_count++]=$id;
            for($m = (count($arr)-1) ; $m>=0 ;$m--) {
                if(!in_array($arr[$m], $full)) {
                    $id=$arr[$m];break;
                }
            }
        }
    }
    else {
        for($k=0;$k<count($var);$k++) {
            if(!in_array($var[$k],$full)) {
```

```

        if(in_array($var[$k],$arr)) {
            $cycle=1; #cycle detected
            break;
        }
        else {
            $arr[$i++]=$var[$k];
            $id=$var[$k]; break;
        }
    }
}
if($k==count($var)) {
    $full[$full_count++]=$id;
    ##now select the next node to be visited
    for($m= (count($arr)-1) ; $m>=0 ;$m--) {
        if(!in_array($arr[$m], $full)) {
            $id=$arr[$m]; break;
        }
    }
}
}
}
else { ###code for disconnected graph
    for($n=0;$n<$num;$n++) {
        if(!in_array($topics[$n],$arr)) {
            $arr[$i++]=$topics[$n];
            $id=$topics[$n]; break;
        }
    }
}
}
}

```

- Strategy Changer Module :** This module helps to change the strategy when student is not performing well with any strategy and there is another strategy is available in database to teach this subtopic. For implementing this module we have made our own algorithm. When there is any subtopic which can be taught by more than one teaching strategy than instructor have to maintain priority order between these strategies and this is stored in strategy\_sequene\_table. When student selects any subtopic than this module first checks how many strategies are associated with this subtopic. If there is one strategy than nothing to do special this module presents question according to this strategy. But in the case of more than one strategy this module goes to strategy\_sequene\_table and finds the highest priority order strategy and present question according to this. After the completion of each subtopic system calculates percentage perform of a student which keeps the record of student's performance on the basis of strategy. Next time when

student selects any subtopic and this subtopic has more than one strategy to teach. After selecting the strategy from sequencing table system also checks student performance for this strategy. If student previous performance for this strategy is below cutoff level than system picks 2nd priority order strategy for teaching.

## 5.4 Challenges

The goal our MTP thesis was to make a generic framework for an ITS which will can support more than one teaching strategy and it should be domain independent. For achieving this target we have faced mainly following challenges:

- Find an appropriate pattern or structure which can support more than one teaching strategy and this structure also suits for most of the domain
- Find teaching Strategies which can be compatible with these structure
- Find common and uncommon module between these strategy. According to this make a generalised architecture
- Decide user interface for each strategy because each strategy has its own requirements.
- In the last the big challenge was DATABASE integration.

The goal of any ITS is to act as a private tutor for each user in order to make one to one relationship in teaching that is not possible with other teaching style like classroom teaching. The design and development of such tutors involved intersection of three areas: computer science, cognitive psychology and educational research [1]. Computer science will be used to store knowledge efficiently and present it dynamically. Cognitive psychology is important to understand student behavior to identify what material is relevant for individual. Education research aims to identify strategy used by such systems to teach. So, mutual understanding of all three disciplines is important to develop such computer systems. Here we have focused mainly first and third i.e computer science and teaching strategy. After research of so many tutoring system we found that there is no such ITSs which can support more than one teaching strategy together.

After looking architecture of so many ITSs We have decide MCQs pattern to make our system. So for supporting MCQs pattern we have also studied so many teaching strategies. Among them we have picked 4 strategies 1. socratic Questioning 2.Scaffolding 3.Guided Discovery 4. Game Based Learning.Each strategy has its own requirements so we have researched individual requirements for each strategy. We have developed single ITS for each strategy and after that we have figured out common module and non common module in all strategy. According to this common module and non common module we have developed our databases and user interfaces.

# Chapter 6

## Integration of all strategies

This chapter includes combined work of all four of us. First of all each of us implements his own strategy individually. Then we compare the modules of each strategy. We found many modules and tables are same in implementation of all of us strategies. Here we listed all common and non-common modules and tables.

### 6.1 Modules

#### 6.1.1 Common modules to all 4 strategies

- GUI module(GUIM)
- Input Validation module(IVM)
- Student module
- Authentication module(AM)
- Course Validation module (optional for my strategy)
- Quiz Maintainer Module
- Evaluation module
- Database handling module
- Log module
- Feedback module
- Result generator module
- Logic module
- Course Module
- Topic Structure Module
- Quiz maker Module

## 6.1.2 Non common modules

### 1. Scaffolding Strategy

- Hint Module
- Logic Module

### 2. Guided Discovery Strategy

- Sequence Module

### 3. Socratic Strategy

- **Sequencing Module** : It is an independent module. It is used by socratic strategy only. This module established a relationship between option and next question. [Detailed explanation is done in section 5.3.]
- **Subtopic Validation Module** : It is an important module for Socratic teaching strategy because for attempting any specific subtopics it is necessary to complete all prerequisite subtopics. [Detailed explanation is done in section 5.3.]
- **Student Previous Progress** : Information: In Socratic strategy student previous progress report is very necessary because on this information system will present next question in front of the student. On this information system maintains percentage performance of a student. With the help of this percentage performance system will decide whether to change the strategy or not.

## 6.2 Tables

### 6.2.1 Common tables

- Login table
- Course table
- Topic table
- Subtopic table
- Strategy table
- Question table : we combine all fields required by each strategy in one table.
- Upload\_file table
- Student\_info table
- Student\_progress table
- Log table

## 6.2.2 Non-common Tables

### 1. Socratic Questioning Strategy

- **parsed\_table** : This table acts like temporary table during content creation by instructor. It store teacher\_id,course\_id,topic\_id,subtopic\_id,question\_id and parse value. Initially when a question is added to question\_table then in this table its entry happens and its parse value is set to 0. Parse value indicate option of this question\_id is related to another question or not. i.e if parse value is 0 it means for this question\_id sequencing is not done. If parse value is 3 it means for this question\_id sequencing has completed for optionA, OptionB, OptionC but not for OptionD. If parse value becomes 4 then this row is removed from table.
- Sequencing table : Already discussed in section 5.2.

### 2. Scaffolding Strategy

- Student level

### 3. Guided Discovery Strategy

- Sequence Module

### 4. Table used for integration

- Strategy Sequencing table: Already discussed in section 5.2.

## 6.3 Integration of modules and tables

For integration purpose first we have found out common and uncommon modules and table requirements. We put all tables and module together and implement the part which is useful for adequate functioning of system. After that we have made some module for integtating these non-common mdules to our system.

### 6.3.1 Module used for integration

- **Strategy Sequencing Module** :This module works when one subtopic is taught by more than one teaching strategy. In that case instructor has to made priority order between teaching strategy. Instructor has to also decide the cutoff level for each strategy. If student performs for any strategy below cutoff level. Then system will change the strategy.
- **Strategy Selector Module** :This module works in two ways. **First** it checks whether the subtopic selected by student is associated with single strategy or not. If this subtopic is associated with single strategy then this module finds out which strategy it is. According to chosen strategy this module presents question in front of student.**Second** if the selected subtopic is associated with more than one teaching strategy then this module goes to strategy\_sequencing\_table and checks the priority order of strategy and choose highest priority order strategy and presents question according to this strategy.

- **Strategy Changer Module** :Instructor sets some cutoff level for each strategy. If student performs below this cutoff level then this module comes in picture. This module goes to strategy\_sequencing\_table table and picks another strategy on the basis of priority order. If all the strategy are already used than system will set default strategy in which student is performing good among in all strategies.

## 6.4 Content for Testing

- Subject: Arrays in Data structure using C.
- Target audience: Cs101 students.
- Teaching Objective: Here I have tried to end up the topic array with 20 questions. And I have also arranged the questions to make it sequential. Through these questions My goal is to covered following facts about array-
  - Definition of Array
  - Initializations of Array (1-D,2-D,3-D)
  - How to calculate sizeof Array when length is not given.
  - Array elements accessed with pointer
  - different representation of array elements
  - different combination with pointer

### MCQs On Topic(Array) Based on socratic Teaching Strategy

1. Which one is correct?
  - (a) Array is a collection of similar data types variables..
  - (b) Array can be a collection of different data types variables.
  - (c) Array's elements are always stored in non contiguous memory locations.
  - (d) None
2. Which one is wrong array initialization? [array definition]
  - (a) `int num[6] = 2, 4, 12, 5, 45, 5 ;`
  - (b) `int n[ ] = 2, 4, 12, 5, 45, 5 ;`
  - (c) `float press[ ] = 12.3, 34.2, -23.4, -11.3 ;`
  - (d) `int a[5]=2, 4, 12.3, 34.2,12;`
3. `int a[5]= 12,14,16,18,20;` An address of an element 12 and 14 are 65508 65510 respectively. what will be the address of `a[4]` ? [array definition]
  - (a) 65522
  - (b) 65516
  - (c) 65508



- (d) Can't be said
4. Which one is wrong?
- (a) Array is a collection of similar data types variables..
  - (b) Array is subscript variables.
  - (c) Array's elements are always stored in contiguous memory locations.
  - (d) The entire above are wrong.
5. What is the difference between the 5s in these two expressions? (Select the correct answer)
- ```
int num[5] ;  
num[5] = 11 ;
```
- (a) First is particular element, second is type
  - (b) First is array size, second is particular element
  - (c) First is particular element, second is array size
  - (d) Both specify array size.
6. `int a[5]= 12,14,16,18,20;` What will be the value of `a[4]` ? [Initializations of Array (1-D)]
- (a) 12
  - (b) 5
  - (c) 20
  - (d) can't say.
7. What will be output if you will execute following c code? [Initializations and value retrieval]
- ```
#include<stdio.h>  
void main(){  
char arr[6]="Vikas";  
printf("%s",arr);  
}
```
- (a) Vikas
  - (b) V
  - (c) vikas
  - (d) Garbage value
8. Array supports bound checking ? [ Memory test question for array property]
- (a) True
  - (b) False
9. What will be output if you will execute following c code? [Initializations and value retrieval]

```
#include<stdio.h>
void main(){
char arr[11]="The Indian Queen";
printf("%s",arr);
}
```

- (a) The Indian Queen
- (b) The
- (c) Queen
- (d) Compilation error

10. What would happen if you try to put so many values into an array when you initialize it that the size of the array is exceeded? [Use of array property discuss in question 8]

- (a) Nothing
- (b) Possible system malfunction.
- (c) Error message from the compiler.
- (d) other data may be overwritten

11. What will be output if you will execute following c code? [Array with pointer]

```
int main()
{
char s[ ]="man";
int i=1;
printf("\n%c%c%c%c",s[i],*(s+i),*(i+s));
}
```

- (a) man
- (b) mmm
- (c) aaa
- (d) nnn

12. What will be output if you will execute following c code?

```
#include<stdio.h>
void main(){
int xxx[10]={5};
printf("%d %d",xxx[0],xxx[6]);
}
```

- (a) 0 5
- (b) 5 5
- (c) 5 0

(d) Compilation error

13. What will be output if you will execute following c code? [Array elements access with pointer]

```
#include<stdio.h>
void main(){
char *ptr="cquestionbank";
printf("%d",-3[ptr]);
}
```

- (a) 100
- (b) -100
- (c) 101
- (d) -101

14. What will be output if you will execute following c code? [use of Sizeof function and 2-D array]

```
#include<stdio.h>
void main(){
int arr[][3]={{1,2},{3,4,5},{5}};
printf("%d %d %d",sizeof(arr),arr[0][2],arr[1][2]);
}
```

- (a) 12 0 5
- (b) 18 1 5
- (c) 18 0 5
- (d) 12 0 5

15. What will be output if you will execute following c code? [calculate length with the help of sizeof]

```
#include<stdio.h>
void main(){
int arr[10]={1,2,3,4,5,6};
printf("%d",sizeof(arr)/arr[0]);
}
```

- (a) 40
- (b) 4
- (c) 10
- (d) 6

16. What will be output if you will execute following c code? [calculate length with the help of sizeof]

```
#include<stdio.h>
void main(){
int arr[]={1,2,3,4,5,6};
printf("%d",sizeof(arr)/arr[0]);
}
```

- (a) 40
- (b) 4
- (c) 10
- (d) 6

17. What will be output if you will execute following c code? [Array elements access with pointer]

```
#include<stdio.h>
int main(){
short num[3][2]={3,6,9,12,15,18};
printf("%d %d",*(num+1)[1],**(num+2));
}
```

- (a) 12 15
- (b) 12 12
- (c) 15 15
- (d) Compilation error

18. What will be output if you will execute following c code? [3-D array initalisation ]

```
#include<stdio.h>
#define var 3
void main(){
char data[2][3][2]={0,1,2,3,4,5,6,7,8,9,10,11};
printf("%o",data[0][2][1]);
}
```

- (a) 5
- (b) 6
- (c) 7
- (d) Compilation error

19. Compilation error means

- (a) Syntax in error
- (b) Program Logic error
- (c) Program Semantic error

20. What will be output if you will execute following c code? [combination of various concept]

```
#include<stdio.h>
void main(){
long myarr[2][4]={01,11,21,31,41,51,61,71};
printf("%ld\t",myarr[1][2]);
printf("%ld%ld\t",*(myarr[1]+3),3[myarr[1]]);
printf("%ld%ld%ld\t" ,*(*(myarr+1)+2),*(1[myarr]+2),3[1[myarr]]);
}
```

- (a) 6 66 777
- (b) 6 77 667
- (c) 5 66 777
- (d) 7 77 666

21. What will be output if you will execute following c code? [combination of various concept]

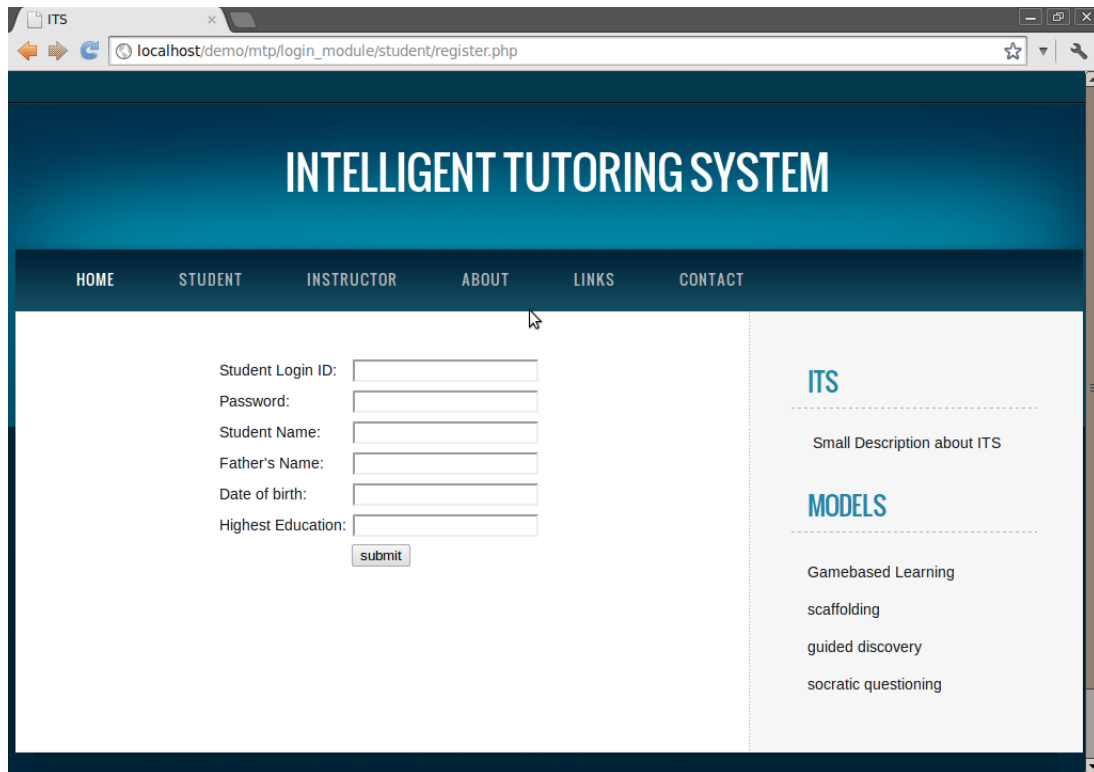
```
#include<stdio.h>
#define WWW -1
enum {cat,rat};
void main(){
int Dhoni[]={2,'b',0x3,01001,'\x1d','\111',rat,WWW};
int i;
for(i=0;i<8;i++)
printf(" %d",Dhoni[i]);
}
```

- (a) 2 98 3 513 29 73 0 -1
- (b) 2 99 3 513 29 73 1 -1
- (c) 2 98 3 513 29 73 1 -1
- (d) Compilation error

Relationship between next question and options (**Sequencing for above questions**) is shown in figure 5.2.

## 6.5 Screen-shot of demo

### 6.5.1 For New User



The screenshot shows a web browser window with the URL `localhost/demo/mtp/login_module/student/register.php`. The page features a dark blue header with the text "INTELLIGENT TUTORING SYSTEM" in white. Below the header is a navigation menu with links for HOME, STUDENT, INSTRUCTOR, ABOUT, LINKS, and CONTACT. The main content area is divided into two columns. The left column contains a registration form with the following fields: Student Login ID, Password, Student Name, Father's Name, Date of birth, and Highest Education. A "submit" button is located below the form. The right column contains a sidebar with the heading "ITS" and a sub-heading "MODELS". Under "MODELS", there is a list of learning models: Gamebased Learning, scaffolding, guided discovery, and socratic questioning.

Figure 6.1: Registration Page For New User

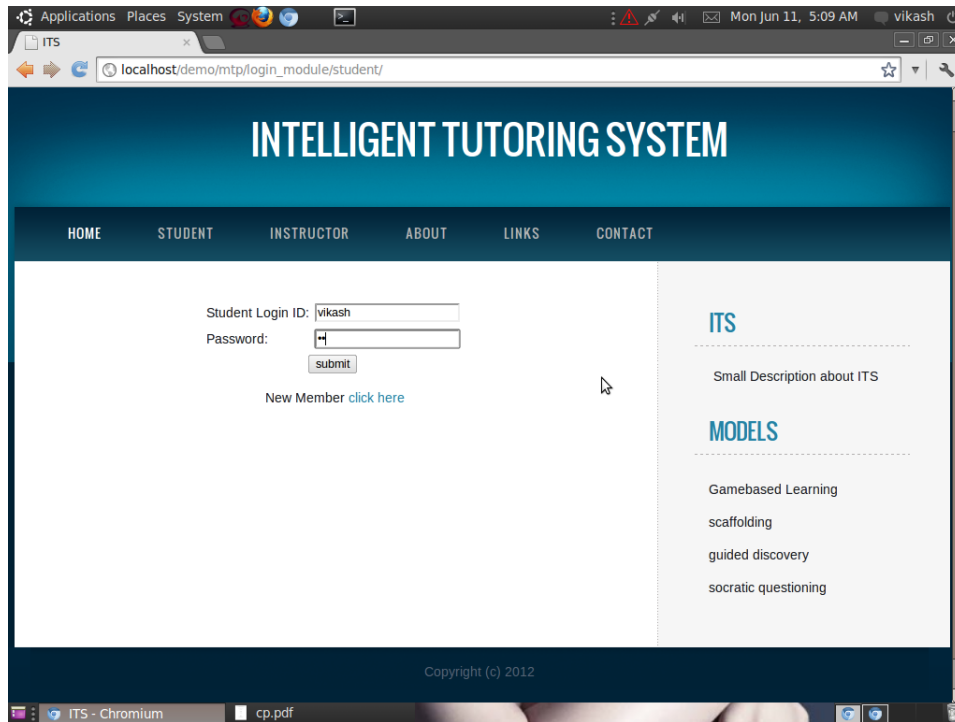


Figure 6.2: Student Login

## 6.5.2 Learner

## 6.5.3 For Instructor

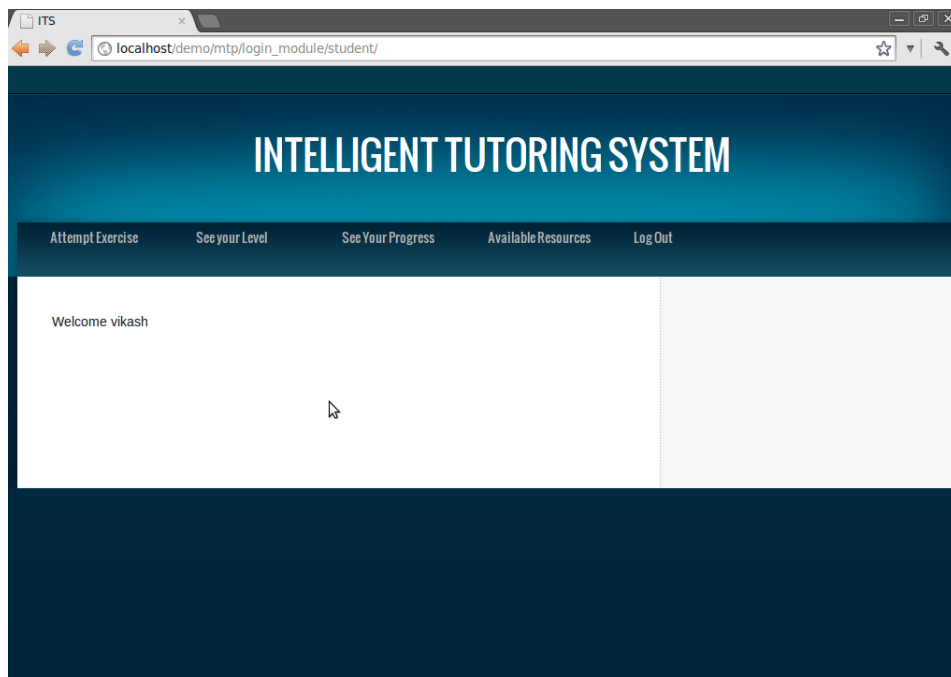


Figure 6.3: Student Home page

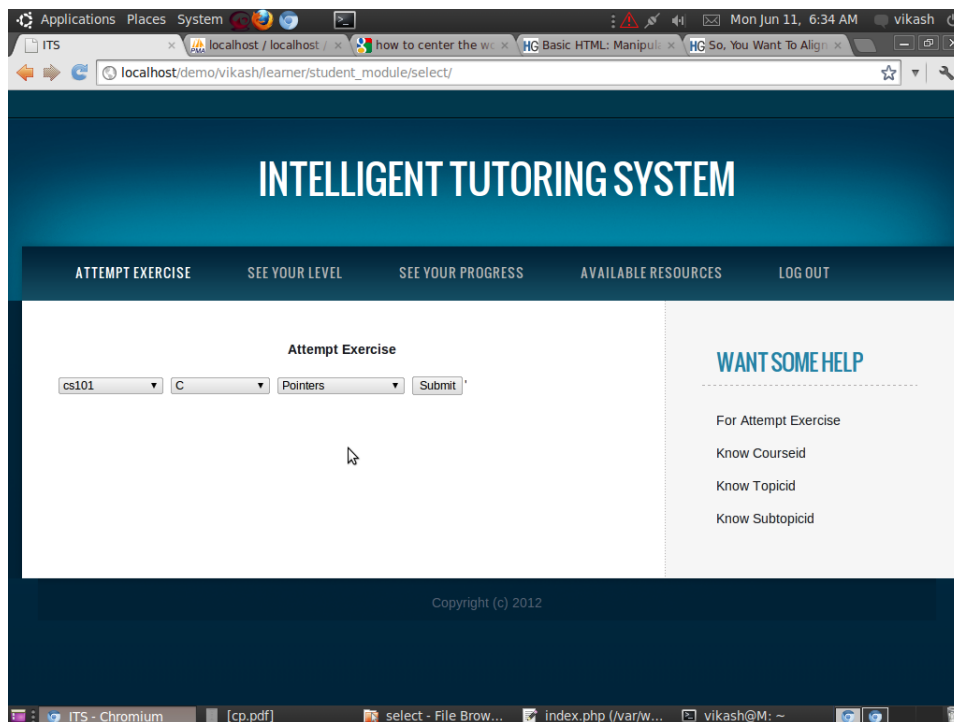


Figure 6.4: Attempt Exercise



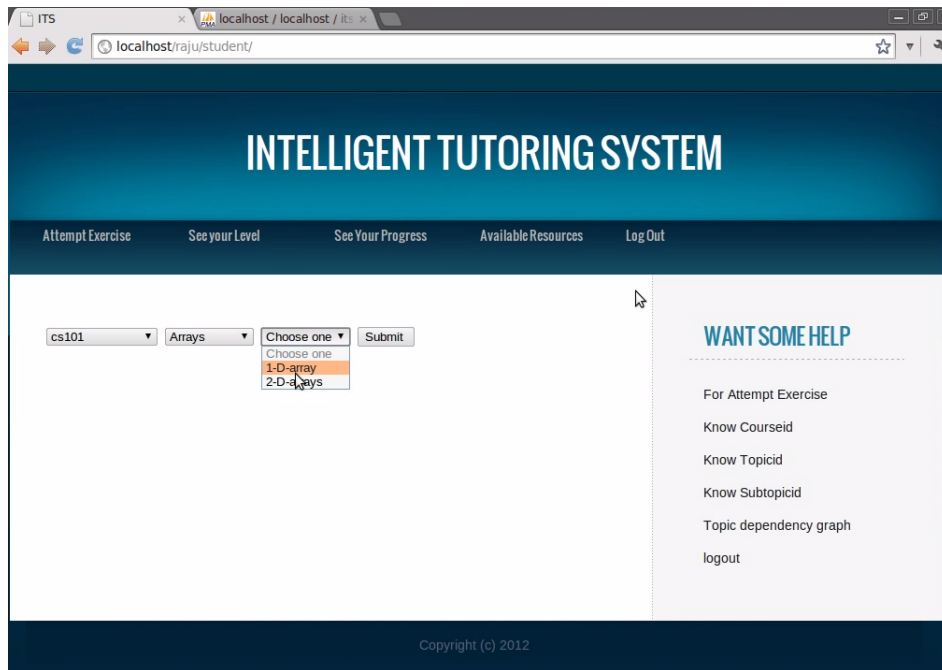


Figure 6.5: Select Exercise

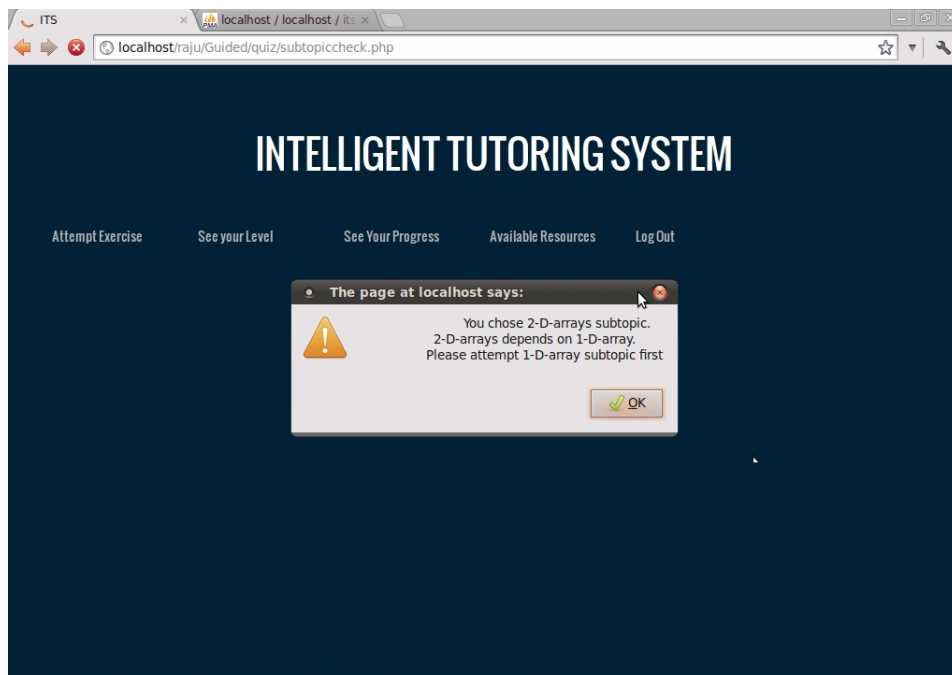


Figure 6.6: Warning Message when Dependency break

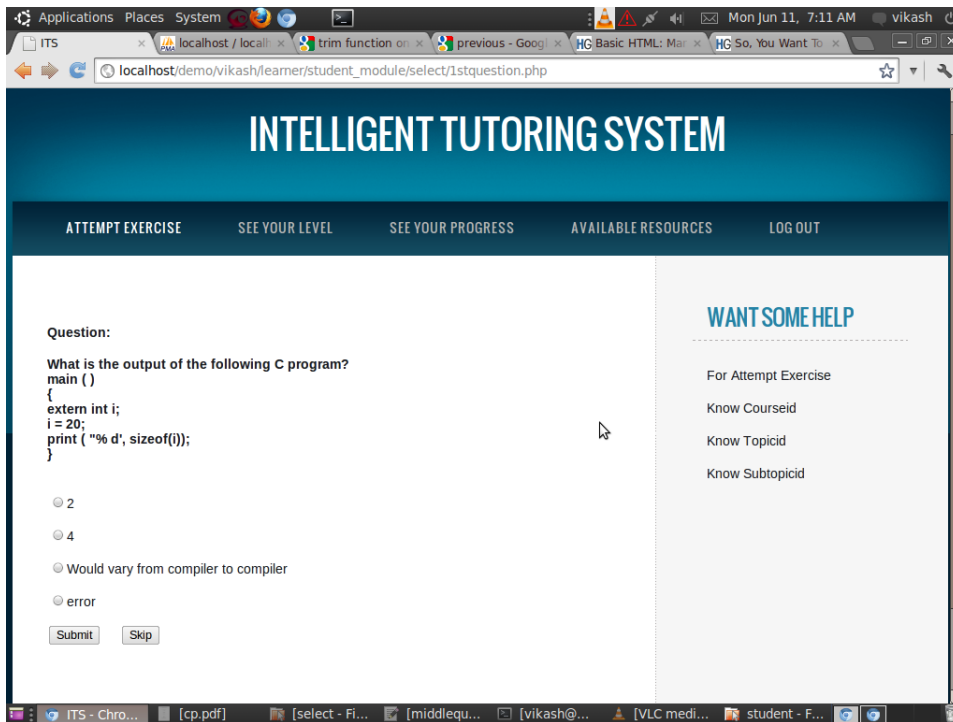


Figure 6.7: 1st Question

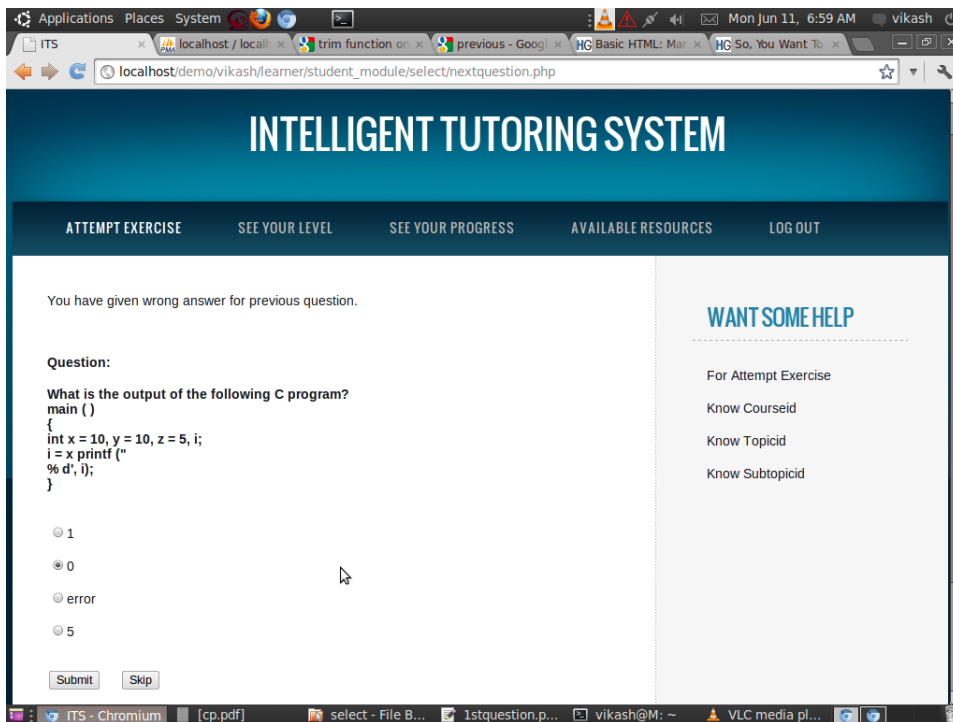


Figure 6.8: Next Question

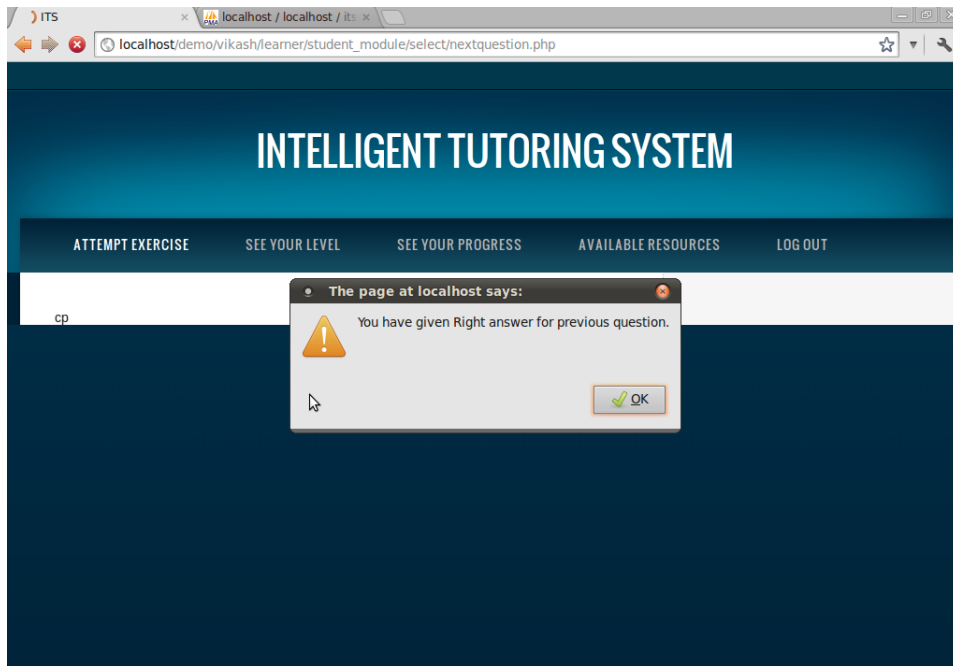


Figure 6.9: Result of Previous Question

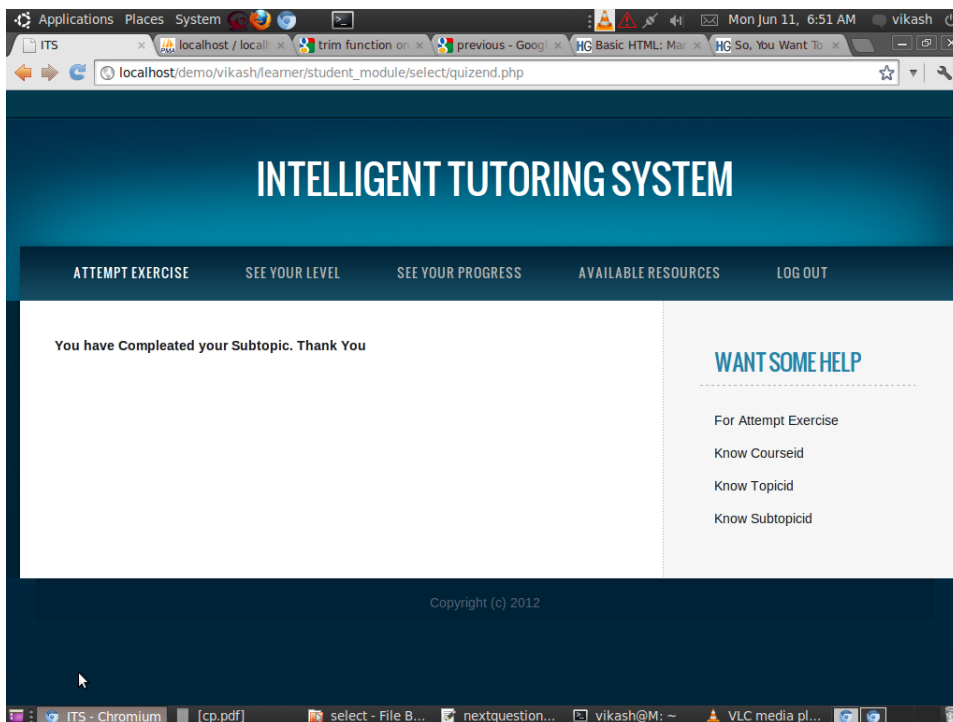


Figure 6.10: Subtopic Completion

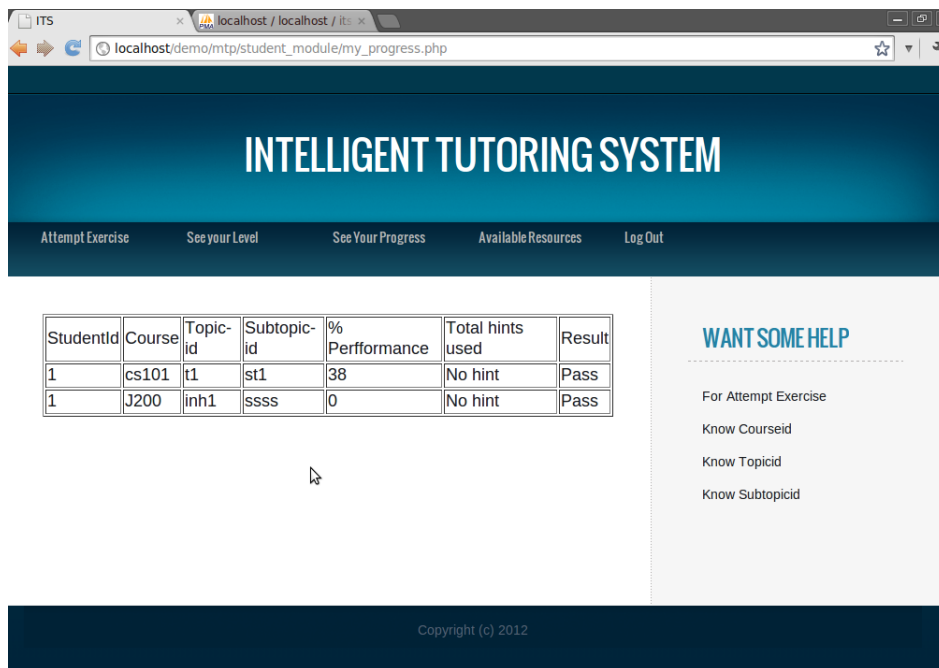


Figure 6.11: Student Performance

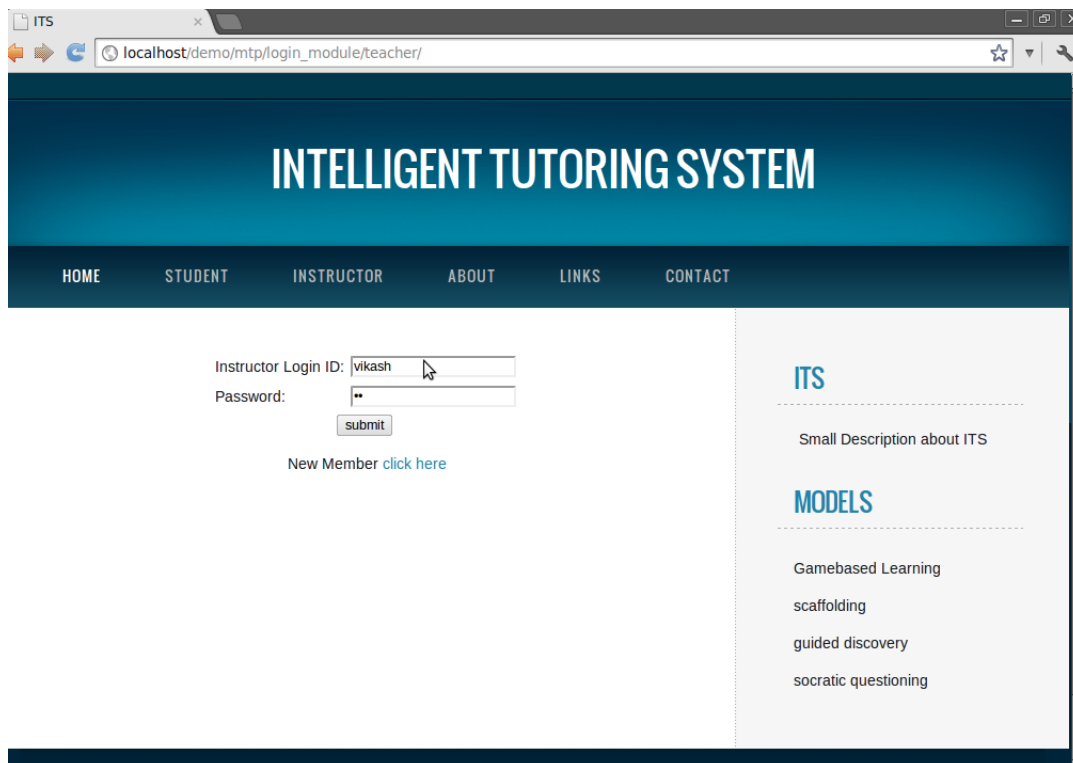


Figure 6.12: Instructor Login Page

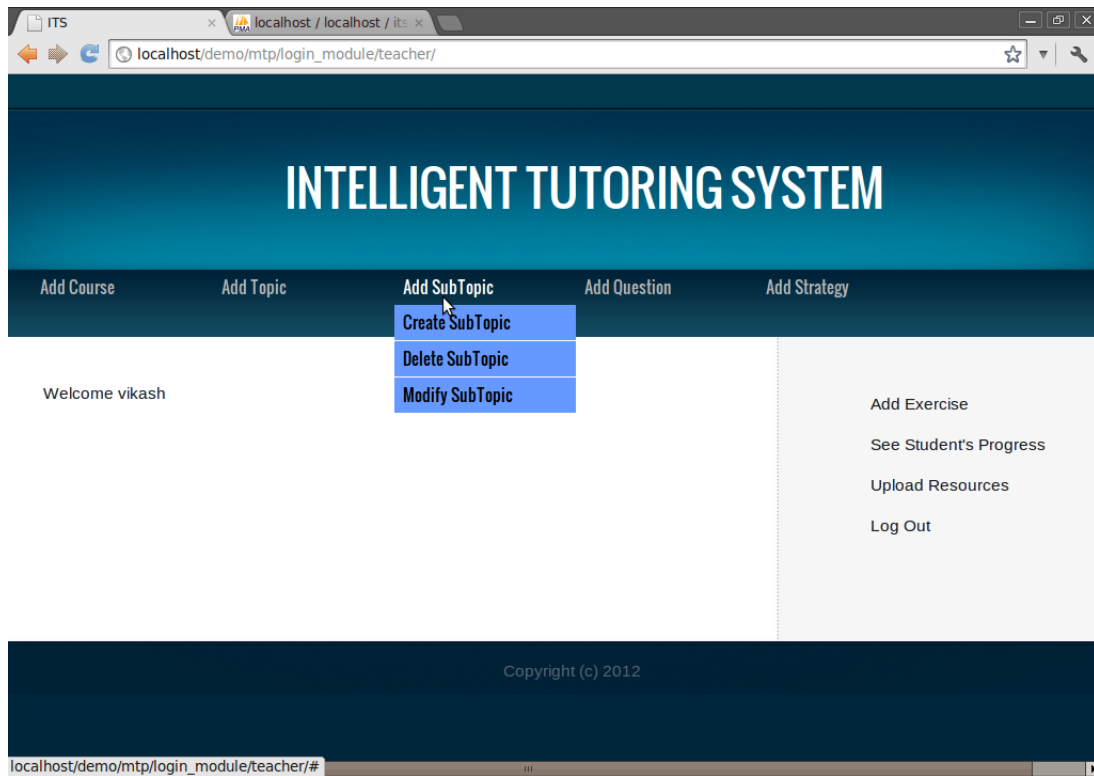


Figure 6.13: Instructor Home Page

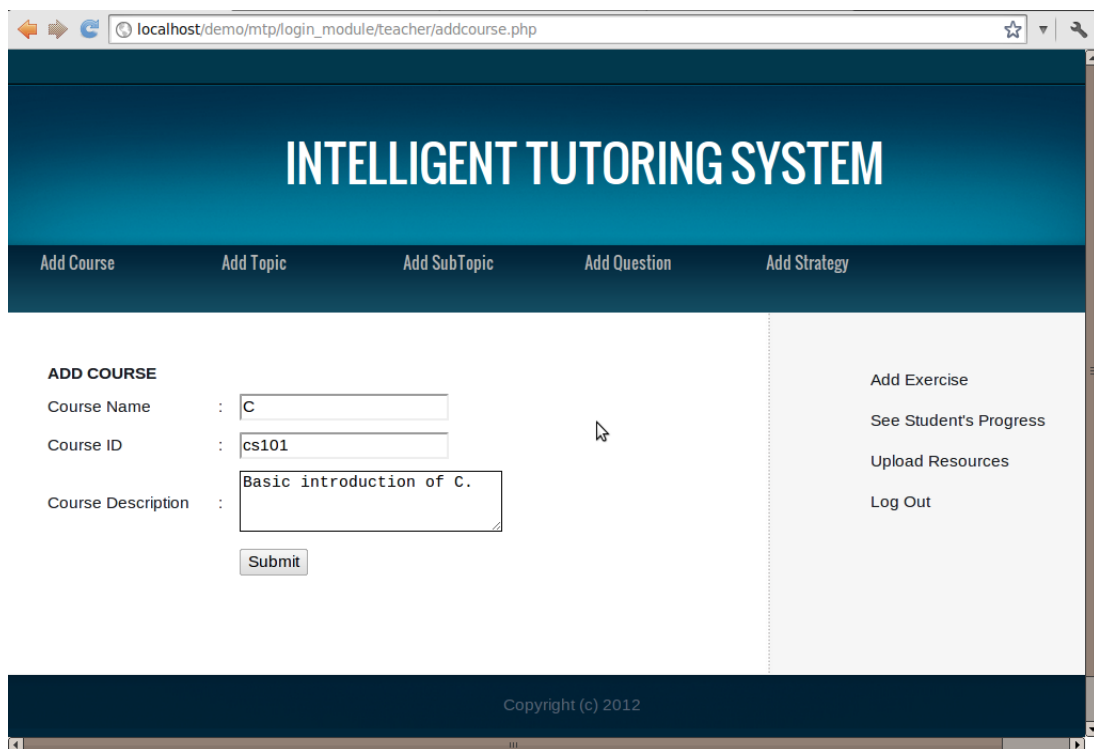


Figure 6.14: Add Course

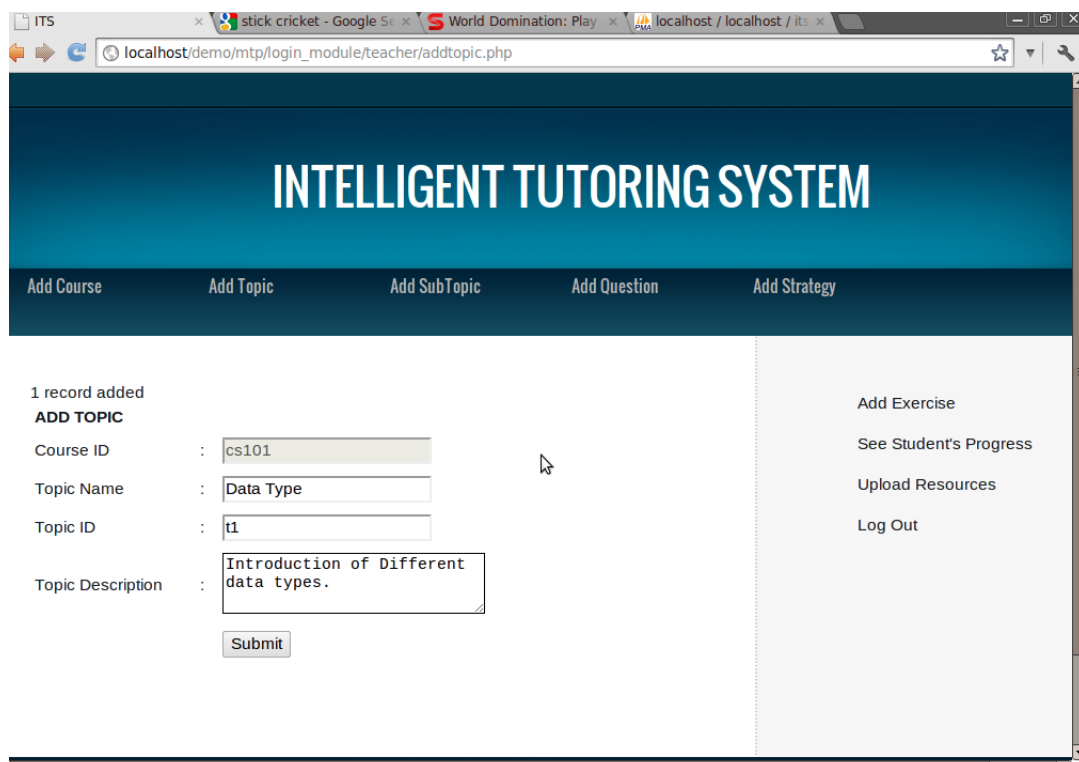


Figure 6.15: Add Topic

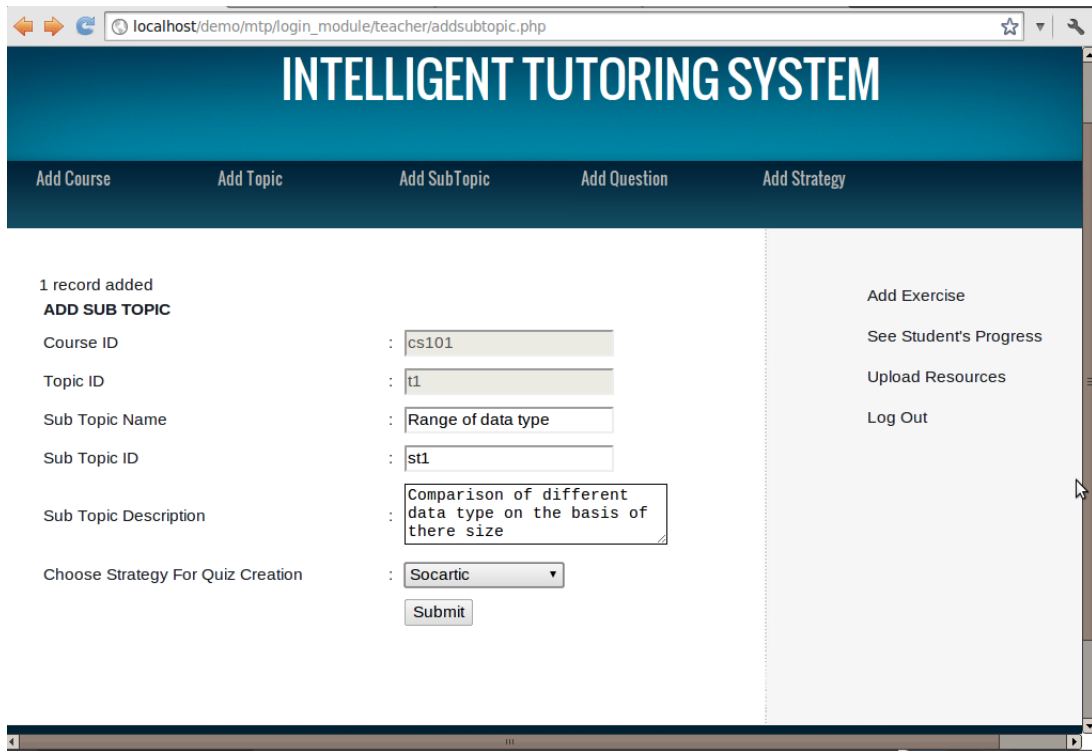


Figure 6.16: Add Subtopic and Choose Strategy for Quiz creation

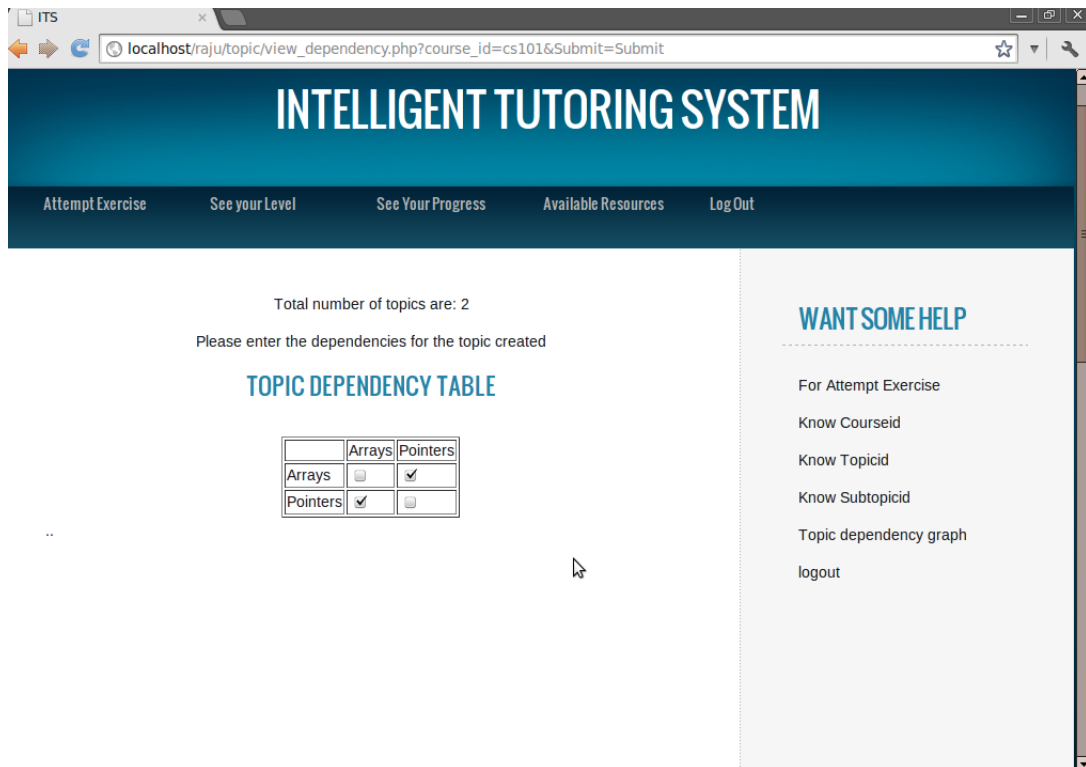


Figure 6.17: Subtopic Dependency Creation Interface

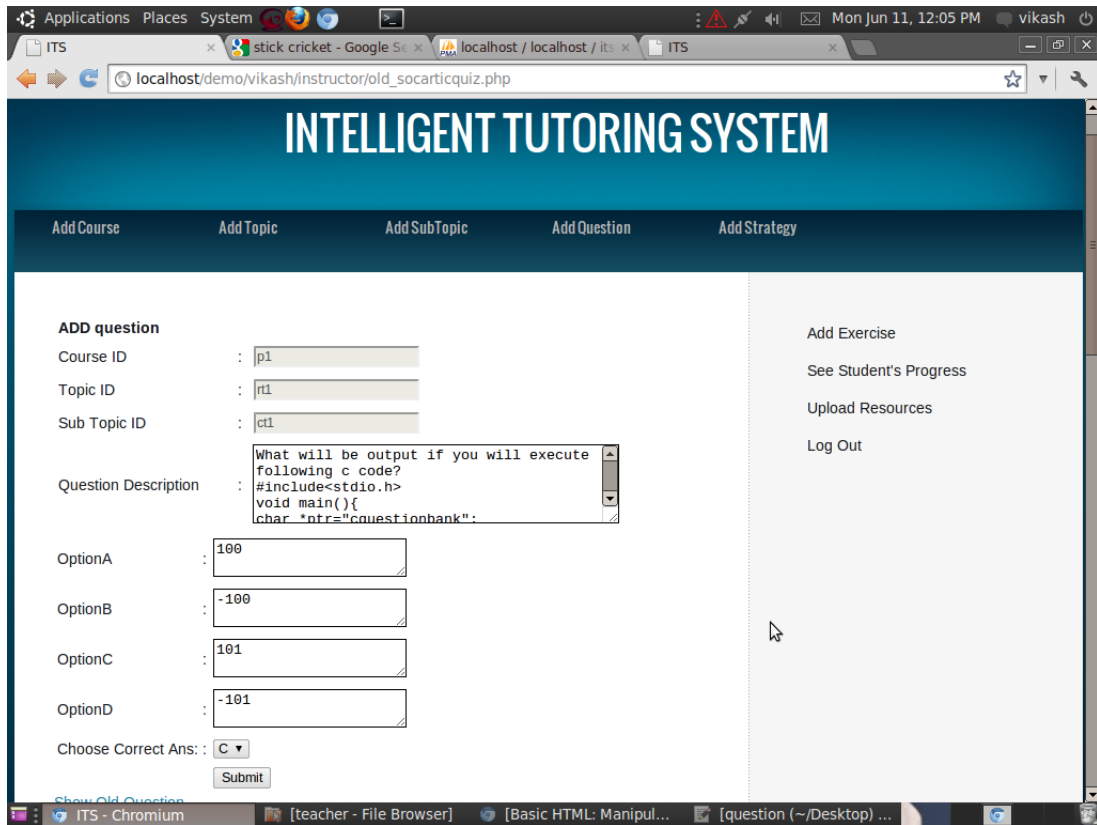


Figure 6.18: Socratic Quiz Creation Interface

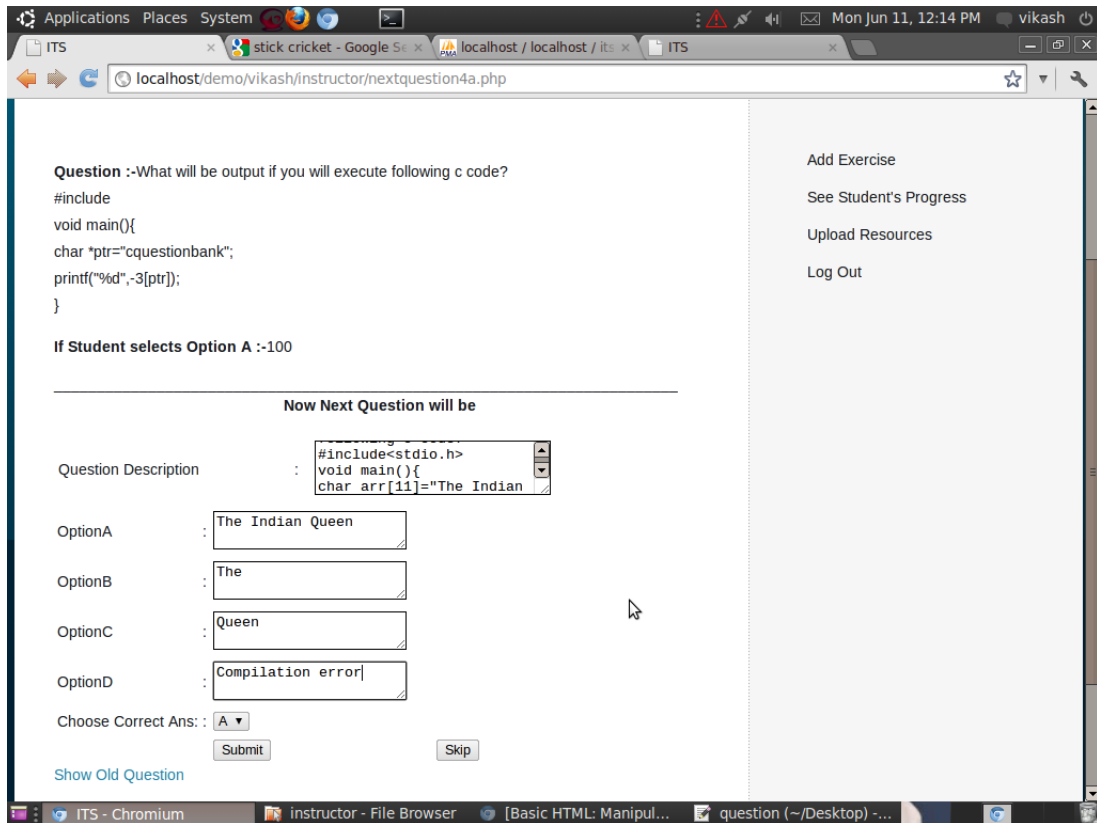


Figure 6.19: Next question relationship with OptionA



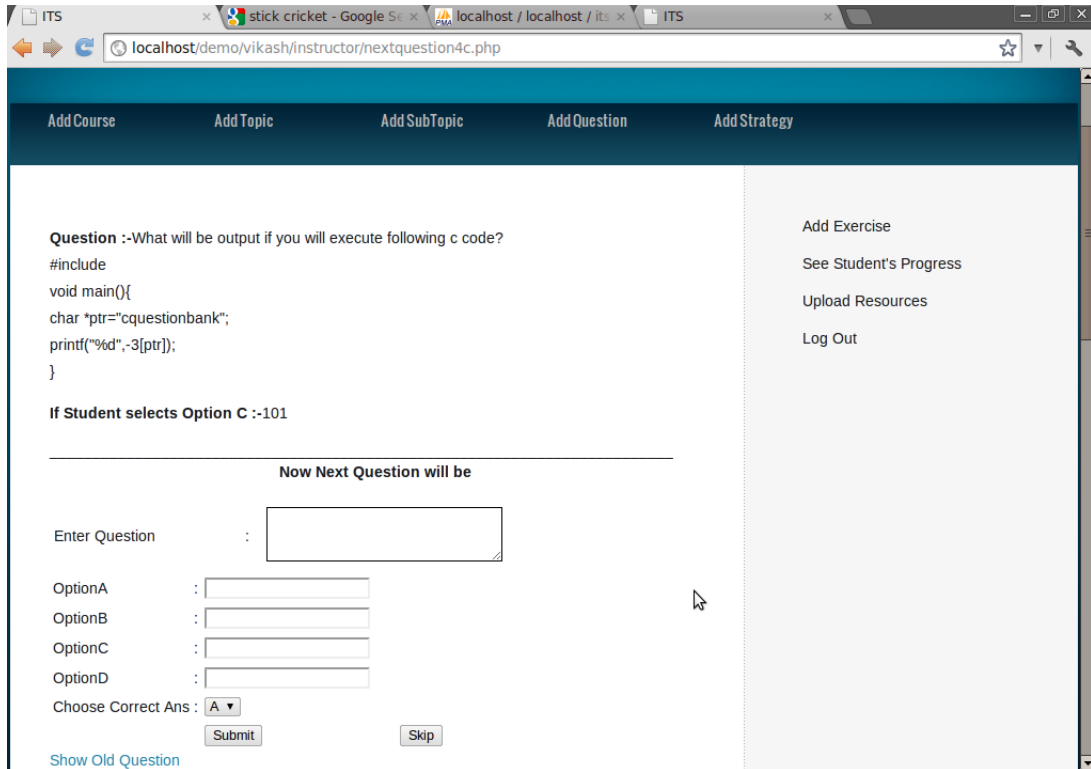


Figure 6.20: Next question relationship with OptionC

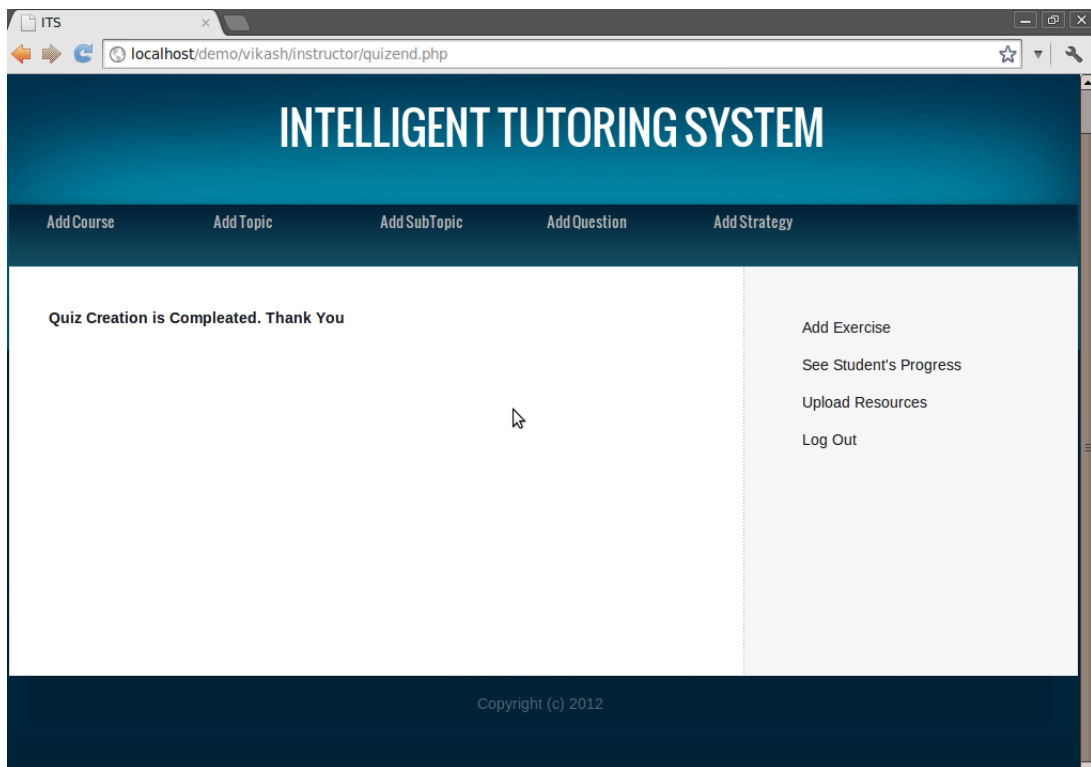


Figure 6.21: Quiz Creation Complete

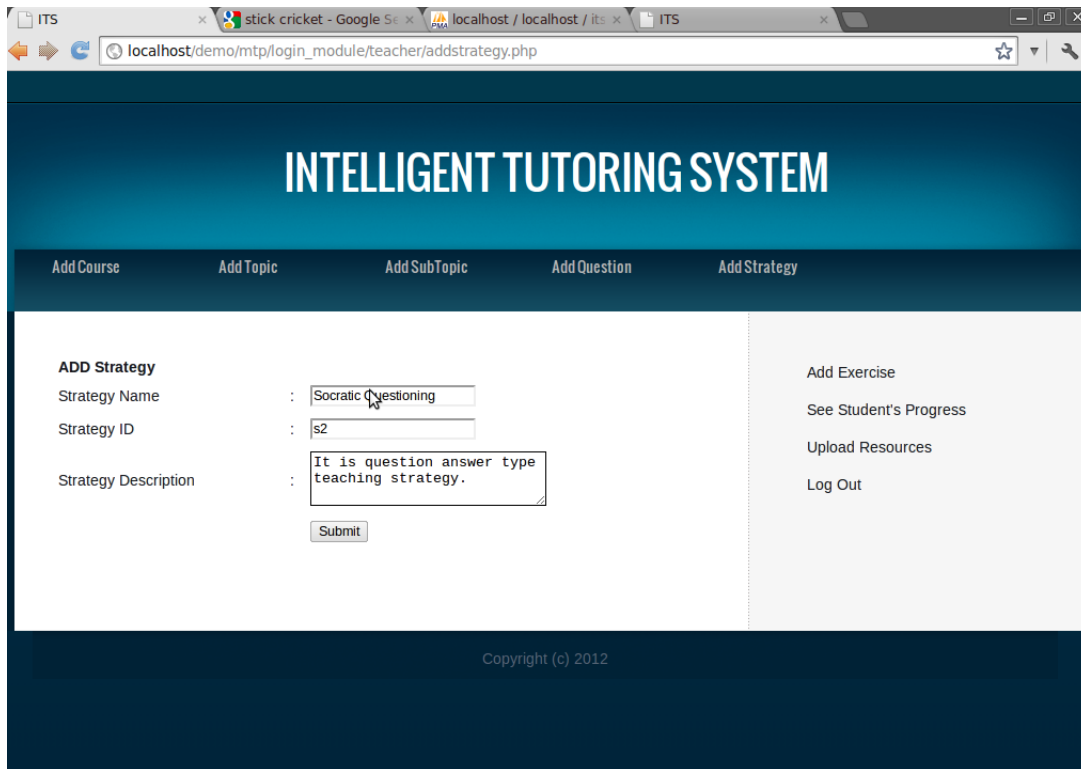


Figure 6.22: Add strategy

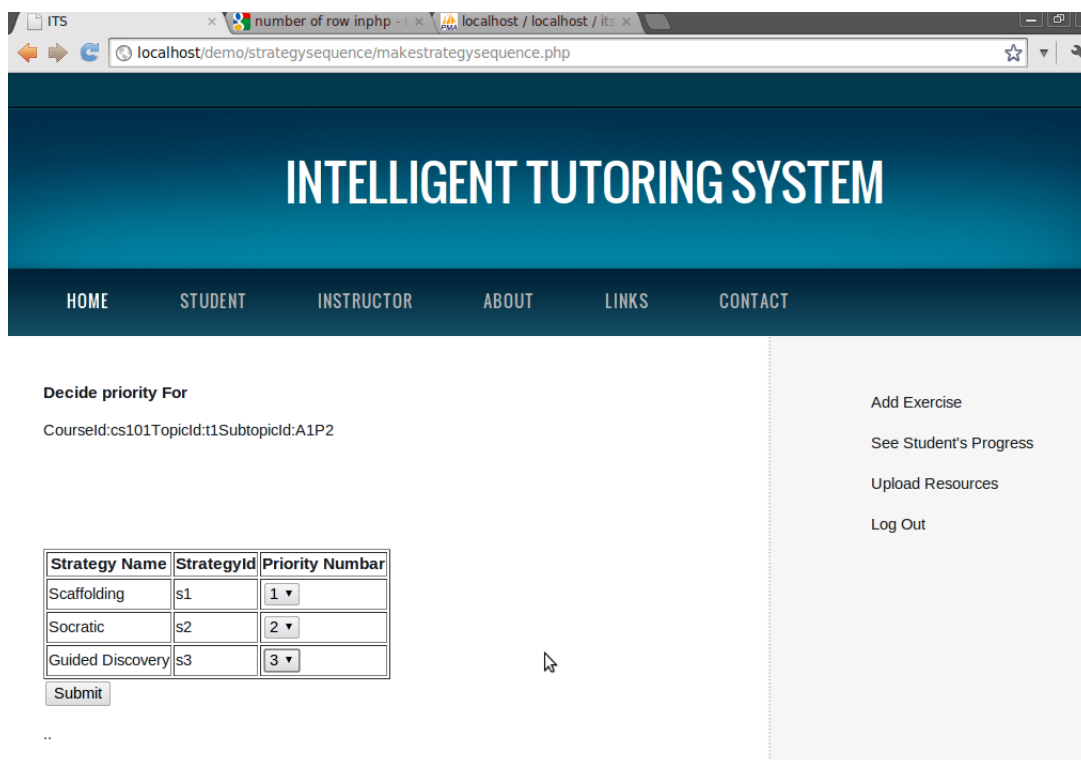


Figure 6.23: Decide Priority Between Strategy

# Chapter 7

## Future work

At this time we have a basic framework for developing ITS. There is lot more remains to be done. One can develop an ITS according to ones requirement. In future following thing can be done.

- **Collecting material for teaching:** The very first work is to prepare the material to student for learning like question with hints and some short tricks etc for scaffolding strategy, for socartic strategy proper sequencing between questions is very important. i.e material is very important.
- **Subjective Questions:** Our system is limited to work for MCQ questions but in future we can extend it for subjective questions also. If any strategy supports for subjective questions then we can extend it.
- **Response Time Theory :** One can implement a module for time response. So that time taken by student in attempting a question or subtopic can be stored. This stored time then can be used to take important decisions. This time will be helpful to find out the interest of student in topic or learning.
- **More teaching strategies :**strategy can easily added. Currently system is working for four strategy only. In future more teaching strategy can be added.
- **Collaborative learning :** Some module for providing collaborative learning also can be added to the system. So that students can interact with each other and learn more with each others experience. The level of student can be used as a competitive factor by showing top level students to each student.

At this time Intelligent Tutoring System (ITS) is a hot topic in industry. In India it is not quite famous but It is the need of education system. In other developed country research on ITS is going so fast. ITS is now part of many schools in developed country.

# Chapter 8

## Conclusion

At this time Our ITS is limited to teach any subject with the help of multiple choice questions (MCQs) only. But the scope of our ITS can be extended in future. So our ITS will provide the flexibility to the instructor to teach with the strategy of his interest.

# Appendix A

## Structure Of Tables

### 1. Login Table

Column Name	Type	Purpose
UserName	varchar(20)	Defing Course name
UserId	varchar(20)	Fou uniquely define the users(Instructor or Learner)
Password	varchar(100)	it stores password for users
users_type	varchar(10)	It stores type of users like instructor or learner

**Structure of login\_table**

### 2. Course Table

Column Name	Type	Purpose
CourseName	varchar(20)	Defing Course name
CourseId	varchar(20)	Fou uniquely define the course(Primary Key)
CourseDes	varchar(100)	Little Description About Course

**Structure of course\_table**

### 3. Topic Table

Column Name	Type	Purpose
CourseId	varchar(20)	Fou uniquely define the course(Primary Key)
TopicName	varchar(20)	Defing Topic name
TopicId	varchar(20)	Fou uniquely define the topic(Primary Key)
TopicDes	varchar(100)	Little Description About topic

**Structure of topic\_table**

#### 4. Subtopic Table

Column Name	Type	Purpose
CourseId	varchar(20)	Fou uniquely define the course(Foregin Key)
TopicId	varchar(20)	Fou uniquely define the topic(Foregin Key)
SubtopicName	varchar(20)	Defing Subtopic name
SubtopicId	varchar(20)	uniquely define the Subtopic(Primary Key)
SubtopicDes	varchar(100)	Little Description About Subtopic

**Structure of subtopic\_table**

#### 5. Parsed Table

Column Name	Type	Purpose
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
QuestionId	int(10)	Unique identification id for Question
Parsed	int(5)	Tamporary value to check whether particular question is parsed or not.

**Structure of parsed\_table**

#### 6. Student Response Table

Column Name	Type	Purpose
StudentId	varchar(15)	User unique identification id
StrategyId	varchar(10)	Unique identification id for Strategy
CourseId	varchar(10)	Unique identification id for Course
TopicId	varchar(10)	Unique identification id for Topic
SubtopicId	varchar(10)	Unique identification id for Subtopic
Hint_used	int(10)	toatal number of hint used by user
Marks_obtained	int(11)	marks obtained.

**Structure of student\_response\_table**

## 7. Question Table

Column Name	Type	Purpose
QuestionId	int(50)	Unique identification id for Question
StrategyId	varchar(20)	Unique identification id for Strategy
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
QuestionDes	varchar(500)	It stores the question description
Option1	varchar(400)	It stores first option
Option2	varchar(400)	It stores second option
Option3	varchar(400)	It stores third option
Option4	varchar(400)	It stores fourth option
CorrectAns	varchar(20)	It stores correct option

**Structure of question\_table**

## 8. Strategy Table

Column Name	Type	Purpose
StrategyName	varchar(20)	Defing Strategy name
StrategyId	varchar(20)	Fou uniquely define the Strategy(Primary Key)
StrategyDes	varchar(100)	Little Description About Strategy

**Structure of strategy\_table**

## 9. Sequencing Table

Column Name	Type	Purpose
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
QuestionId	int(50)	Unique identification id for Question
ChosenOption	varchar(15)	It stores option chosen by learner
NextQuestion	int(50)	Unique identification id for Question

**Structure of sequencing\_table**

## 10. Strategy Sequencing Table

Column Name	Type	Purpose
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtop
s1	int(15)	It stores priority order of strategy
s2zint(15)	It stores priority order of strategy s2.	
s3	int(15)	It stores priority order of strategy

**Structure of strategy\_sequencing\_table**

In this table s1,s2 and s3 represents the strategyId of Scaffolding, Socartic and Guided Discovery respectively. This tables increase or decrease dynamically i.e when any strategy is added or removed from databases then column in this table increases or decreases.

## 11. Student Performance Table

Column Name	Type	Purpose
StudentId	varchar(15)	User unique identification id
QuestionId	int(50)	Unique identification id for Question
StrategyId	varchar(20)	Unique identification id for Strategy
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
Percentage_performance	int(15)	It stores % of right question for paticular subtopic.

**Structure of student\_performance table**



## 12. Student Progress Table

Column Name	Type	Purpose
StudentId	varchar(15)	User unique identification id
QuestionId	int(50)	Unique identification id for Question
StrategyId	varchar(20)	Unique identification id for Strategy
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
result	varchar(15)	It stores only two value Right or Wrong according to Instructor's response
total_hint used	int(15)	It stores total number of hint used by instructor.
total_marks used	int(15)	It stores total marks obtained by learner i.e how many questions learner has given right ans or how many are wrong.

**Structure of student\_progress\_table**

## 13. Student Subtopic Table

Column Name	Type	Purpose
StudentId	int(15)	User unique identification id
QuestionId	int(50)	Unique identification id for Question
CourseId	varchar(15)	Unique identification id for Course
TopicId	varchar(15)	Unique identification id for Topic
SubtopicId	varchar(15)	Unique identification id for subtopic
Status	varchar(15)	It stores only two value Complete or Incomplete according to learner's attempt

**Structure of student\_subtopic\_status\_table**



# Appendix B

## File Description

Folder/file Name	Folder/file's work description
demo/vikash/instructor/select/addcourse.php	This file is used to create the new course.
demo/vikash/instructor/select/addtopic.php	This file is used to create the new topic.
demo/vikash/instructor/select/addsubtopic.php	This file is used to create the subtopic.
demo/vikash/instructor/select/direct_addsubtopic.php	This file is used to create the subtopic if course and topic are already available.
mtp/course_creator_module/select	this folder is used select the course, topic and subtopic for which resorses are uploading.
mtp/login_module/student	This folder contains files which used by student for login and register.
mtp/login_module/student/index.php	This file is used for authentication and validation of student.
mtp/login_module/student/register.php	This file is used for registering the new student.
mtp/login_module/teacher	This folder contains files which used by teacher for login and register.
demo/vikash/instructor/select/socraticquiz.php question creation. the subtopic.	This file is used to provide interface for
demo/vikash/instructor/nextquestion4a.php	This file is used to provide interface for making relationship between next question and first option of previous question.
demo/vikash/instructor/nextquestion4c.php	This file is used to provide interface for making relationship between next question and third option of previous question.
mtp/login_module/teacher/register.php	This file is used for registering the new teacher.
demo/vikash/learner/select/1stquestion.php <sup>64</sup>	This file gives 1st question to student for a particular topic.

# Bibliography

- [1] Bloom, B.S. *The 2 Sigma Problem: The search for Methods of Group Instruction as Effective as One-to-One Tutoring*, Educational Researcher, vol. 13, no. 6, pp. 416, 1984.
- [2] D. Sleeman, and J. S. Brown, *Introduction: Intelligent Tutoring Systems*” Ed. Academic Press, New York, 1982, pp. 1-11.
- [3] Ivon Arroyo, Rena Walles, Carole R. Beal, Beverly P. Woolf, *Tutoring for SAT-Math with Wayang Outpost* University of Massachusetts, Amherst
- [4] Hyancith S.Nwana. *Intelligent Tutoring Systems* Department of computer Science University of Liverpool,Liverpool L69,3BX,UK
- [5] Karen L. Hornsby and Wade M. Maki(2008), *The Virtual Philosopher: Designing Socratic Method Learning Objects for Online Philosophy Courses*. North Carolina A&T State University,University of North Carolina Greensboro
- [6] Cotton, K. (2001). *Classroom questioning*. School Improvement Research Series(SIRS). <http://www.nwrel.org/scpd/sirs/3/cu5.html> (25.10.2005).
- [7] [http://en.wikipedia.org/wiki/Socratic\\_questioning](http://en.wikipedia.org/wiki/Socratic_questioning)10July2011at00:33.
- [8] PETE BOGHOSSIAN *How Socratic Pedagogy Works* Portland State University College of Education
- [9] A. C. Graesser, S. Lu, G. T. Jackson, H. Mitchell, M. Ventura, A. Olney,and M. M. Louwerse, *AutoTutor: A tutor with dialogue in natural lan-guage*, Behav. Res. Methods, Instrum., Comput., vol. 36, pp. 180193, 2004.
- [10] Antonija Mitrovic. *An intelligent sql tutor on the web*. University of Canterbury,New Zealand, 2003.
- [11] Albert T. Corbett, Kenneth R. Koedinger, and John R. Anderson. *Hand book of Human-Computer Interaction*, chapter 37. USA, 1997.
- [12] Wikipedia.[http://en.wikipedia.org/wiki/Intelligent\\_tutoring\\_system](http://en.wikipedia.org/wiki/Intelligent_tutoring_system).
- [13] Tom Murray. *Authoring intelligent tutoring systems: An analysis of the state of the art*. In International Journal of Artificial Intelligence in Education, Computer Science Dept., University of Massachusetts, 1999.
- [14] *Artificial intelligence*. <http://www.aaai.org/AITopics/html/tutor.html>.
- [15] Valerie J. Shute and Joseph Psotka. *Intelligent tutoring systems present, past and future*. 1994.

- [16] Antonija Mitrovic, Chris Williamson, Aidan Bebbington, Moffat Mathews, Pramuditha Suraweera, Brent Martin, David Thomson, Jay Holland *Thermo-Tutor: An Intelligent Tutoring System for Thermodynamics* Intelligent Computer Tutoring Group University of Canterbury Christchurch, New Zealand
- [17] M. Rajashekhar, *Development of an Intelligent Tutoring System Framework for Guided Discovery*, M-Tech Thesis, IIT Bombay 2012
- [18] Praveen Dhanala, *Development of an Intelligent Tutoring System Framework for Game Based Learning*, M-Tech Thesis, IIT Bombay 2012
- [19] Chandrapal Singh, *Development of an Intelligent Tutoring System Framework for Scaffolding Learning*, M-Tech Thesis, IIT Bombay 2012