

CHAPTER – I

INTRODUCTION

1.1 Objective

Advances in computer technology and the widespread availability of ever more sophisticated computer-based systems is gradually changing the way teaching and assessment is undertaken in computer science and engineering departments. Educational institutions must explore and effectively utilize opportunities provided by such technologies to enhance the educational experience of their students.

Questions are an essential component of effective instruction. If questions are effectively delivered, they facilitate student learning and thinking and provide the opportunity for academics to assess how well students are mastering course content. Questions may be categorized as short answer, multiple choice, essay, etc. The two most commonly used categories are multiple-choice and short-answer questions. In the case of the authors' research, "short-answers" implies free-text entry, requiring answers that have to be constructed rather than selected, ranging from phrases to three to four sentences. Automated short-answer marking is one of the technologies that may be useably explored, and in recent years a number of attempts have been made to automate short-answer marking.

1.2 Motivation

Assessment is used to evaluate the student's understanding of the concepts learned. Computer based assessment can cut down the time that the teachers can take to teach students. A computer can analyse and examine student's answer deeply. It can ease the burden of teachers of assessing large number of questions. Automated assessment can be applied on objective-type questions like follows:-

- Fill in the Blanks
- True/False
- ne-word Answer
- Match the columns

- Mental Maths in which student writes resultant value only.
- Multiple Choice Questions

But evaluating Subjective-type questions is a difficult task. While evaluating Subjective-type questions, teacher has to evaluate so many aspects like student's extent of knowledge, number and depth of concepts learned, to evaluate Student's knowledge relevant to that particular topic etc. So subjective type evaluation is very time-consuming. Much of the teacher time is spent in evaluating Student papers.

This project, short answer assessment is a computer based assessment using the concept of artificial intelligence. This assessment will evaluate student's answer automatically and also checks grammatical structure in student's answer. Teachers can save their time that was previously spent in marking question papers. They can do their other tasks. This application is closer to Student and Teacher expectation.

1.3 Contributions

The contributions of this work are as follows:

- An assessment method that evaluates student's answer automatically which replaces the tedious job of teacher's manual evaluation.
- Satisfying student and teacher marks expectations through this evaluation.

1.4 Overview

Assessment is used to evaluate the student's understanding of the concepts learned. Computer based assessment can cut down the time that the teachers can take to teach students. A computer can analyse and examine student's answer deeply. It can ease the burden of teachers of assessing large number of questions. Evaluation is very time-consuming. Much of the teacher But evaluating Subjective-type questions is a difficult task. While evaluating Subjective-type questions, teacher has to evaluate so many aspects like student's extent of knowledge, number and depth of concepts learned, to evaluate Student's knowledge relevant to that particular topic etc, .So Subjective type evaluation is very time-consuming. Much of the teacher time is spent in evaluating Student papers. So in this paper, we will discuss a computer based technique that will evaluate student's answer automatically.

1.5 Problem Statement

The project aims to create an application for evaluating short answers overcoming the tedious conventional evaluation process. Teachers can save their time that was previously spent in marking answer papers. They can do their other tasks. This technique is closer to Student and Teacher expectation. An example of a short-answer objective question is: “What is a pointer?” The correct answer is: “A pointer is a variable which stores the address of the other variable.” Correct student responses are expected to be paraphrases of this concept, and therefore, the primary task of the automated marking system is to recognize which answers are paraphrases of the correct concept and which are not.

Short answer assessment systems compare students’ responses to given questions with manually defined regions or answer keywords in order to judge the appropriateness of the responses, or in order to automatically assign marks.

1.6 Layout Of Thesis

This thesis is organized as follows :

- Chapter I deals with Introduction to the project, objective, motivation, problem statement and contributions.
- Chapter II explains about Existing Automated Evaluators, Stanford Parser.
- Chapter III discusses about I/O specifications and system requirements.
- Chapter IV gives the design details of the system like overall architecture of the system and design(UML) diagrams.
- Chapter V explains the implementation of the system. It discusses in detail how the system is actually built
- Chapter VI deals about result analysis.
- Chapter VII discusses about conclusions and future scope.
- References followed by Appendix consist of sample code segments.

CHAPTER – II

LITERATURE SURVEY

This chapter mainly describes about, Applications, Advantages and Disadvantages of Automated Evaluators.

2.1 Existing Automated Evaluators

2.1.1 C-Rater

C-rater[2] tries to recognize when a response is equivalent to a correct answer, and so is, in essence, a paraphrase recognizer. As such, the scoring engine is designed to recognize a correct response when it exhibits the variations that are ordinarily associated with paraphrases, whether they be syntactic variation, different inflections of a word, substitution of synonyms or similar terms, or the use of pronouns in the place of nouns. In addition to these features, which are ordinarily associated with paraphrasing, c-rater recognizes words that are spelled incorrectly – an essential feature for the K-12 market.

The recognition of the syntactic structure, inflected words, the referent of a pronoun and spelling correction are all fully automated. In the case of synonyms or similar words, a suggested list generated from a corpus of over 300 million words of current fiction, nonfiction, and textbooks is presented to the model-builder, who can select from among them while building a c-rater model answer. A detailed description of the mechanisms that drive c-rater can be found in Leacock and Chodorow (Forthcoming) Essentially, a model needs to represent the full range of concepts that a response must contain to receive full or partial credit.

C-rater looks at each sentence in a student's response and determines whether it is a paraphrase of a sentence in the model. It is important to note that c-rater is not simply matching words – the paraphrases must obey syntactic constraints. For example, if "Peter ate the apple" is in the model, the sentence "The apple ate Peter" will not be recognized as a valid paraphrase. A question can be scored by c-rater if there is a finite range of concepts that satisfy it. Thus an open-ended question asking for an opinion or for examples from the student's own experience is not a question for c-rater.

2.1.2 WebLAS

One of the earlier systems is WebLAS, presented by Bachman et al. A human task creator feeds the system with scores for model answers. Regular expressions are then created automatically from these model answers. Since each regular expression is associated with a score, matching the expression against a student answer yields a score for that answer. Bachman et al. (2002) do not provide an evaluation study based on data.

2.1.3 CarmelTC

Another earlier system is CarmelTC by Ros   et al. It has been designed as a component in the Why2 tutorial dialogue system. Even though Ros   et al position CarmelTC in the context of essay grading, it may be considered to deal with short answers: in their data, the average length of a student response is approx. 48 words. Their system is designed to perform text classification on single sentences in the student responses, where each class of text represents one possible model response, plus an additional class for ‘no match’. They combine decision trees operating on an automatic syntactic analysis, a Naive Bayes text classifier, and a bag-of-words approach. In a 50-fold cross validation experiment with one physics question, six classes and 126 student responses, hand-tagged by two annotators, CarmelTC reaches an F-measure value of 0.85. They do not report on a baseline. Concerning the quality of the gold standard, they report that conflicts in the annotation have been resolved.

2.1.4 IAT

Information extraction templates form the core of the Intelligent Assessment Technologies system. These templates are created manually in a special-purpose authoring tool by exploring sample responses. They allow for syntactic variation, e.g., filling the subject slot in a sentence with different equivalent concepts. The templates corresponding to a question are then matched against the student answer. Unlike other systems, IAT additionally features templates for explicitly invalid answers. They tested their approach with a progress test that has to be taken by medicine students. Approximately 800 students each ploughed through 270 test items. The automatically graded responses then were moderated: Human judges streamlined the answers to achieve a more consistent grading. This step already had been done before with tests graded by humans. Mitchell et al[5] state that their system reaches 99.4% accuracy on the full dataset after the manual adjustment of the templates via the moderation process. Summarizing, they report an error of “between 5 and 5.5%” in inter-grader agreement and an error of 5.8% in automatic grading without

the moderation step, though it is not entirely clear which data these statistics correspond to. No information on the distribution of grades or a random baseline is provided.

2.1.5 Indus Marker

Indus Marker exploits structure matching, i.e., matching a pre specified structure, developed via a purpose-built structure editor, with the content of the student's answer text. The examiner specifies the required structure of an answer in a simple purpose designed language. The language was initially called QAL but later on the authors redefined it as a sublanguage of XML and named it Question Answer Markup Language (QAML). The syntax and semantics of QAL is intended to be suitable for educators with widely differing computing skills, i.e., QAL is intentionally simple enough to be readily understandable and hence easy to learn. The language also embodies the necessary constructs to express a structure for a natural language text.

2.1.6 The IE-based short-answer marking system

The IE-based short-answer marking system was developed at Oxford University to fulfil the needs of the University of Cambridge Local Examination Syndicate (UCLES). The system depends on pattern matching for the calculation of marks. A human expert discovers information extraction patterns. A set of patterns is associated with each question. This set is further divided into bags or equivalence classes. The members of an equivalence class are related by an equivalence relation, i.e., a member of an equivalence class conveys the same message and/or information as other members of the same equivalence class. The marking algorithm compares student answers with equivalence classes and awards marks according to the number of matches.

The evaluation of the latest version of the system was carried out using approximately 260 answers for each of the nine questions taken from a UCLES GCSE biology exam. The full mark for these questions ranged from 1 to 4. Two hundred marked answers were used as the training set (i.e., the patterns were abstracted over these answers) and 60 unmarked answers were kept for the testing phase. The average percentage agreement between the system and the marks assigned by human examiner was 84 percent.

2.1.7 Automark

Automark has been developed for robust automated marking of short free-text responses . IE techniques have been used to extract the concept or meaning behind free text and full effort has been made to make the software system tolerant of errors in typing, spelling, syntax, etc. Automark uses mark scheme templates to search for specific content in the student answer text. These templates are representatives of valid (or specifically invalid) answers. The templates are developed using an offline custom written configuration interface. The software system first parses the student answer text and then “intelligently” matches it with each mark scheme template so that marks for the student answer may be calculated. The answer representation of a mark scheme template may be mapped to a number of input text variations. Automark was tested in a real world scenario of “high importance” tests that were part of UK national curriculum assessment of science for pupils at age 11. Four items (of varying degrees of linguistic complexity) were taken from 1999 papers. For each item, 120 student answers were taken. The Automark system was able to correctly mark most of the answers containing spelling, syntax, and semantic errors. The system was able to show its “understanding” of badly expressed ideas and its marking co-relation with human marking ranged between 93.3 and 96.5 percent.

2.2 Advantages and disadvantages of Automated Evaluators

2.2.1 Advantages

- Achieve a fairer and more efficient assessment process: The computer does not get tired, bored, irritated or inattentive. It can examine and analyze essays in much more detail than a human and that it is free of judgments myths, false beliefs and value biases. Furthermore, according Williamson, Bejar and Hone the reproducibility, the consistency, the tractability, the item specification, the granularity, the objectivity, the reliability and the efficiency of the assessment process can also be improved by using computers as grader tools.

- Provide more feedback to the students: Computerized marking could avoid the problem of teachers that do not have time to assess their students' work. Hence, they only send them one term assignment that is clearly not enough to help them to develop their writing skills.
- Reduce costs: If the system can be used in several departments and without considering the initial implantation phase, it could reduce some costs.
- Improve the learning process: By engaging students with interesting questions that are given instant feedback(marks) once the students have answered them.
- Challenge the students: The core idea is that whoever student able to fool the machine to achieve a good score, it is a student with a good knowledge of the domain and hence, he or she deserves the score obtained.

2.2.2 Disadvantages

- The computer is not a credible grading machine: Computer-based analyses are sometimes based solely on symbols manipulation according to some simple previously schemes programmed by humans. Hence, students with a peculiar writing style could be prejudiced.
- The computer lacks human common sense and intelligence: There are still many things that only humans can do.
- The student-teacher relationship gets lost: If teachers stopped reading the students' works they would ignore their students' progress.
- The computer is quite limited in the assessing process: Lonsdale and Strong-Krause[6] advised to take consciousness that some systems could not be adaptable enough to meet the particular needs of an individual, class, teacher, or institution. Moreover, most of the system cannot deal all type of questions and Ford warned that even with writing samples, there might be some samples that cannot be evaluated.
- The computer is useless for high-stakes assessment: For students one point can be crucial. It would not be acceptable for them that the computer has failed and hence, it has given them a lower grade.
- The system's scores might not be legally defensible: Ford wondered if a student' exam automatically assessed would be legally considered.

2.3 Role of Automated Evaluators in Educational Field

Teachers all over the world spend a great deal of time just marking students' works. Hence, they have to cut down the time they can devote to their other duties. Even doing that, sometimes they do not have enough time to properly assess the big number of students they have. Therefore, many authors believe that this situation has to be solved and some of them have presented the computer as a new assessing tool. These authors do not attempt to substitute the teacher with the computer, but to help the teachers with the computer software.

In conformity with Whittingdon and Hunt, the automated assessment of students' essays is regarded by many as the Holy Grail of computer automated assessment. The technical approaches that these tools are undertaking are very different, but the goal and concepts underlying are just the same for all of them.

On the other hand there has always been hard critics about the idea of a computer grading human essays. Nowadays there are still some sceptical researchers that do not consider the automatic grading possible. However, the advances in NLP, machine learning and neural network techniques, the lack of time to give them appropriate feedback (despite the general assumption of its importance) and the conviction that MCQs are a poor assessment method are favouring a change in this situation.

Automatic assessment of students' texts could be seen as the higher level of a hierarchy in which two subcategories could be identified: the automatic assessment of short answers and the automatic assessment of essays. Sometimes the same tool can evaluate both pieces of writing, but, in general, the boundaries between the two tasks are clear and most computer automated assessment tools only evaluate either essays or short answers. This work focuses in the first subcategory, that is, the automatic assessment of short answers. Thus, semi-automated computer based essay marking systems, systems that assess the student ability to summarize or systems to improve the student writing skills are not going to be considered as they are out of the limits of this dissertation.

Concerning the input method, although some researchers describe systems that allow hand-written input, this approach is still too challenging and achieves very low results. Besides, some students are worried because of their bad calligraphy and thus, they are grateful that they can use the keyboard to enter data. Therefore, in this work, keyboard input is the only input method contemplated.

But the key question remains and it is how the computer can effectively measure the student knowledge. Page has made a dichotomy between evaluating content and style. Content would refer to what the essay says, and style refers to syntax, mechanics, diction and other features of the writing. Some researchers are strongly against this classification because they think that, in order to assess the text, both content and style are important, and one should not consider the former without the latter. In order to grade the style, these Computer automated assessment software tools look for direct features in the text, such as word number, word lengths or use of adjectives, and translate them into more abstract measures such as variety, fluency or quality. Although there are some critics to this procedure, it is widely accepted and it can provide good results.

Concerning essay content evaluation, several automated assessment techniques have appeared recently, and some of them are even commercially available. Besides, several traditional tests such as the Graduate Management Admissions Test (GMAT), the Test of English as a Foreign Language (TOEFL) or the Graduate Record Examination (GRE) are including open ended questions with a computer-based delivery, which may support the use of automated scoring methods. Thus, this is a growing field with interests from researchers, educators and businessmen, and with a promising future.

2.4 Natural Language Processing

Natural language processing (NLP) is a field of computer science, artificial intelligence and linguistics concerned with the interactions between the computers and human(natural) languages i.e., it is focused on developing systems that allow computers to communicate with people using everyday language.

Features of NLP

- **Morphological Analysis:** Individual words are analyzed into their components and non word tokens, such as punctuation are separated from the words.
- **Syntactic Analysis:** Linear sequences of words are transformed into structures that show how the words relate each other. Some word sequences may be rejected if they violate the languages rules for how words may be combined.
- **Semantic Analysis:** The structures created by the syntactic analyzer are assigned meanings.

- **Discourse Integration:** The meaning of an individual sentences may depend on the sentences that precede it and may influence the meanings of the sentence(may depend on the sentences that precede it) that follow it.
- **Pragmatic Analysis:** The structure representing what was said is reinterpreted to determine that what was actually meant. For example, the sentence “Do know what time it is?” should be interpreted as a request to be told the time.

The following is a list of some of the most commonly researched tasks in NLP.

Parts of speech tagging

Given a sentence, determine the parts of speech for each word. Many words, especially common ones, can serve as multiple parts of speech. For example, "book" can be a noun ("the book on the table") or verb ("to book a flight"); "set" can be a noun, verb or adjective; and "out" can be any of at least five different parts of speech. Some languages have more such ambiguity than others. Languages with little inflectional morphology, such as english are particularly prone to such ambiguity. Chinese is prone to such ambiguity because it is a tonal language during verbalization. Such inflection is not readily conveyed via the entities employed within the orthography to convey intended meaning.

Parsing

Determine the parse tree(grammatical analysis) of a given sentence. The grammar for natural languages is ambiguous and typical sentences have multiple possible analyses. In fact, perhaps surprisingly, for a typical sentence there may be thousands of potential parses (most of which will seem completely nonsensical to a human).

Question answering

Given a human-language question, determine its answer. Typical questions have a specific right answer (such as "What is the capital of Canada?"), but sometimes open-ended questions are also considered (such as "What is the meaning of life?"). Recent works have looked at even more complex questions.

2.5 Tools Integrated

2.5.1 Stanford Log-linear Part-Of-Speech Tagger

- A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads text in some language and assigns parts of speech to each word (and other token), such as noun, verb, adjective, etc., although generally computational applications use more fine-grained POS tags like 'noun-plural'. The tagger was originally written by Kristina Toutanova. Since that time, Dan Klein, Christopher Manning, William Morgan, Anna Rafferty, Michel Galley, and John Bauer have improved its speed, performance, usability, and support for other languages.
- The system requires Java 1.8+ to be installed. Depending on whether 're running 32 or 64 bit Java and the complexity of the tagger model, 'll need somewhere between 60 and 200 MB of memory to run a trained tagger (i.e., may need to give java an option like `java -mx200m`). Plenty of memory is needed to train a tagger. It again depends on the complexity of the model but at least 1GB is usually needed, often more. Several downloads are available. The basic download contains two trained tagger models for English. The full download contains three trained English tagger models, an Arabic tagger model, a Chinese tagger model, a French tagger model, and a German tagger model. Both versions include the same source and other required files. The tagger can be retrained on any language, given POS-annotated training text for the language.
- NLP tasks of The Stanford POS Tagger

Query

My dog also likes eating sausage.

Tagging

My/PRP\$
dog/NN
also/RB
likes/VBZ
eating/VBG
sausage/NN

2.5.2 Microsoft Azure

Microsoft Azure [9]/'æʒər/ is a cloud computing platform and infrastructure created by Microsoft for building, deploying, and managing applications and services through a global network of Microsoft-managed data centres.

It provides both PaaS and IaaS services and supports many different programming languages, tools and frameworks, including both Microsoft-specific and third-party software and systems.

Microsoft Azure uses a specialized operating system, called Microsoft Azure, to run its "fabric layer":[citation needed] a cluster hosted at Microsoft's data centers that manages computing and storage resources of the computers and provisions the resources (or a subset of them) to applications running on top of Microsoft Azure. Microsoft Azure has been described as a "cloud layer" on top of a number of Windows Server systems, which use Windows Server 2008 and a customized version of Hyper-V, known as the Microsoft Azure Hypervisor to provide virtualization of services.

Scaling and reliability are controlled by the Microsoft Azure Fabric Controller[citation needed] so the services and environment do not crash if one of the servers crashes within the Microsoft data center and provides the management of the user's web application like memory resources and load balancing.

Azure provides an API built on REST, HTTP, and XML that allows a developer to interact with the services provided by Microsoft Azure. Microsoft also provides a client-side managed class library which encapsulates the functions of interacting with the services. It also integrates with Microsoft Visual Studio, Git, and Eclipse.

In addition to interacting with services via API, users can manage Azure services using the web-based Azure Portal, which reached General Availability. The portal allows users to browse active resources, modify settings, launch new resources, and view basic monitoring data from active virtual machines and services.

2.5.3 Wordnet

WordNet is a lexical database for the English language. It groups English words into sets of synonyms called synsets, provides short definitions and usage examples, and records a number of relations among these synonym sets or their members. WordNet

can thus be seen as a combination of dictionary and thesaurus. While it is accessible to human users via a web browser, its primary use is in automatic text analysis and artificial intelligence applications. The database and software tools have been released under a BSD style license and are freely available for download from the WordNet website. Both the lexicographic data (lexicographer files) and the compiler (called grind) for producing the distributed database are available.

2.5.4 RiTa Wordnet[7]

A software toolkit for computational literature designed to support the creation of new works of computational literature, the RiTa library provides tools for artists and writers working with natural language in programmable media. The library is designed to be simple while still enabling a range of powerful features, from grammar and Markov-based generation to text-mining, to feature-analysis (part-of-speech, phonemes, stresses, etc). All RiTa functions are heuristic and do not require training data, thus making the core library quite compact. RiTa can also be integrated with its own user-customisable lexicon, or with the WordNet database. RiTa is implemented in both Java and JavaScript, is free/libre and open-source, and runs in a number of popular programming environments including Android, Processing, Node, and p5.js.

2.6. Bootstrap Framework[8]

Bootstrap is a free and open-source front-end library for creating websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. It aims to ease the development of websites and web applications. Bootstrap is a front end web framework, that is, an interface for the user, unlike the server-side code which resides on the "back end" or server. Bootstrap is the second most-starred project on GitHub, with over 95K stars and more than 40K forks.

2.6.1 Stylesheets

Bootstrap provides a set of stylesheets that provide basic style definitions for all key HTML components. These provide a uniform, modern appearance for formatting text, tables and form elements.

2.6.2 Re-Usable Components

In addition to the regular HTML elements, Bootstrap contains other commonly used interface elements. These include buttons with advanced features (e.g. grouping of buttons or buttons with drop-down option, make and navigation lists, horizontal and vertical tabs, navigation, breadcrumb navigation, pagination, etc.), labels, advanced typographic capabilities, thumbnails, warning messages and a progress bar. The components are implemented as CSS classes, which must be applied to certain HTML elements in a page.

2.6.3 JavaScript components

Bootstrap comes with several JavaScript components in the form of jQuery plugins. They provide additional user interface elements such as dialog boxes, tooltips, and carousels. They also extend the functionality of some existing interface elements, including for example an auto-complete function for input fields. In version 2.0, the following JavaScript plugins are supported: Modal, Dropdown, Scrollspy, Tab, Tooltip, Popover, Alert, Button, Collapse, Carousel and Type ahead.

CHAPTER – III

PROBLEM SPECIFICATION

3.1 Problem

Each student answer a question in different ways. It is difficult for a teacher to correct and allot marks. Evaluator may go wrong while counting marks of the students.

The project aims to create an application for evaluating short answers overcoming the tedious conventional evaluation process. Teachers can save their time that was previously spent in marking question papers. They can do their other tasks. This technique is closer to Student and Teacher expectation. An example of a short-answer objective question is: “What is the main difference between structures and arrays?” The correct answer is: “Arrays can only hold multiple data items of the same type, but structures can hold multiple data items of different data types.” Correct student responses are expected to be paraphrases of this concept, and therefore, the primary task of the automated marking system is to recognize which answers are paraphrases of the correct concept and which are not.

Short answer assessment systems compare students’ responses to questions with manually defined target responses or answer keys in order to judge the appropriateness of the responses, or in order to automatically assign a grade

3.2 Input Specification

The inputs that are to be given are:

- Questions by the admin that are to be answered.
- Model answer to be given by admin for each question.
- User gives the answer for the question displayed.
- Limitations
 - The system has been developed for the English language, and will therefore be based on English language part of speech elements.

- Primary assumption in this system is that all student responses as well as supplied correct answers are entered by the end users using proper spelling and grammar using complete sentences.
- The system requires that the supplied answers as well as student responses be written in a direct manner. That is, all answers must not use analogies, slang or examples.
- This system focuses on single-sentence responses.

3.3 Output Specification

Once user answers a question and submits, the following outputs will be generated

- Marks for the student answer

3.4 System requirements

3.4.1 Software requirements

- JDK (1.8)
- NetBeans IDE
- Apache Tomcat Server
- Xampp Servers—Apache,MySQL
- MySQL and SQL Server JDBC Drivers
- SQL Server Management Studio
- SQL Azure Migration Wizard
- Web Browser
- Operating system- Windows

3.4.2 Hardware requirements

- Processor - Pentium IV
- Hard Disk - 80GB
- RAM - 2GB
- Speed - 1.1 GHz
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Internet connectivity

3.4.3 Technologies Used

- HTML5
- CSS3
- JSP-Java Server Faces
- Javascript
- SQL
- Photoshop CS6

CHAPTER – IV

SYSTEM DESIGN

4.1 Project Flow

Project flow describes the operations which we are going to perform during the project, diagrammatically. This project consists of two phases, one is admin and another is user who is going to take the exam.

4.1.1 Examiner

Examiner decides the questions he want to pose for the students. Examiner enters the questions and corresponding answers in data base .

- Creates test
- Store questions and corresponding model answer.

(1) Creates test

Based on the requirements of the students examiner creates a test and adds questions for the students .Any number of questions can be added. Evaluator stores questions in SQL tables.

(2) Store questions and corresponding model answers

The admin stores the questions and the related model answer for the question. Different synonyms for subject object verb are taken from wordnet.

4.1.2 User

User takes the exam prepared by examiner. User does the following things

- Starts the test.
- Get question
- Type answer and submit
- Get marks

1 .Starts test

The application starts with start page .When ever student clicks the Take test button exam starts.

2.Get question

After starting the exam first question is displayed.

3.Type answer and submit

User type the answer and submit the answer. User can also clear his answer if he/she thought that the answer might be wrong.

4.2 Analysis and Design

4.2.1 Use case Diagrams

- A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
- The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted
- A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors.

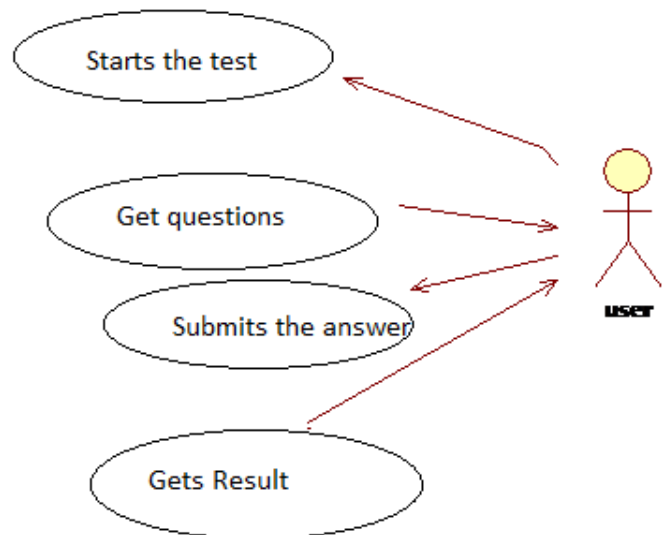
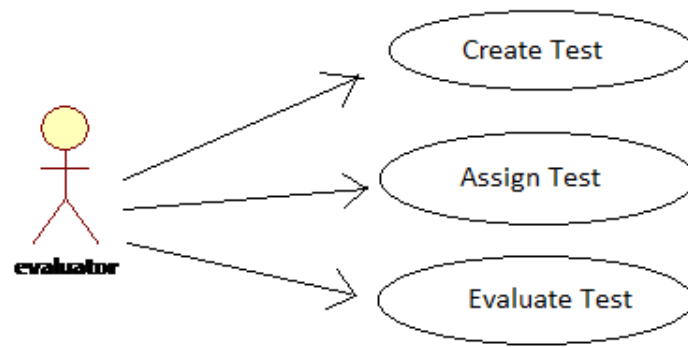


Fig.4.1 use case Diagram

In the Fig.4.1, Two actors are there 1.Evaluator,2.Student. Evaluator add questions and answers. Add regions and possibilities for every answer. Allot marks for every answer and possibility. Student starts he test and submit the answer. After submitting the answer he will get marks.

4.2.2 Sequence diagrams

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

Sequence diagrams demonstrate the behaviour of objects in a use case by describing the objects and the messages they pass. The diagrams are read left to right and descending.

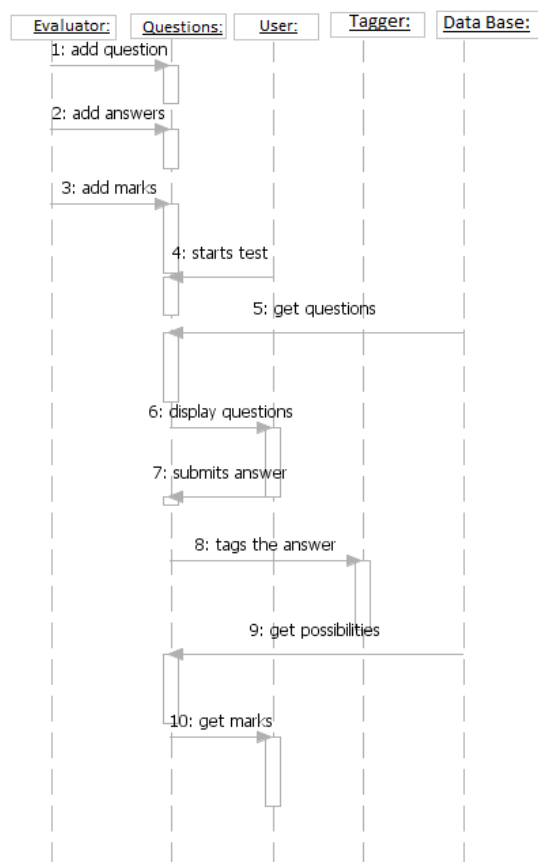


Fig.4.2 Sequence Diagram

In the Fig.4.2 evaluator adds questions answers and assign marks. User starts test and submits answers to the displayed questions. Tagger tags submitted answer. User answers are compared with possibilities stored in XML document which are parsed through parsing.

4.2.3 Collaboration diagrams

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modelling Language (UML). The concept is more than a decade old although it has been refined as modelling paradigms have evolved.

A collaboration diagram resembles a flowchart that portrays the roles, functionality and behaviour of individual objects as well as the overall operation of the system in real time. Objects are shown as rectangles with naming labels inside. These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles. The messages between objects are shown as arrows connecting the relevant rectangles along with labels that define the message sequencing.

Collaboration diagrams are best suited to the portrayal of simple interactions among relatively small numbers of objects. As the number of objects and messages grows, a collaboration diagram can become difficult to read. Several vendors offer software for creating and editing collaboration diagrams.

In the Fig.4.3 the evaluator object and question object are interacted through messages those are adding question and answers and adding marks and user object is interacted with question object through messages those are submit answer and get marks, while questions object is interacted with user object through start test and start test messages. Whereas parsing object is interacted with question object with messages such as get possibilities and get questions.

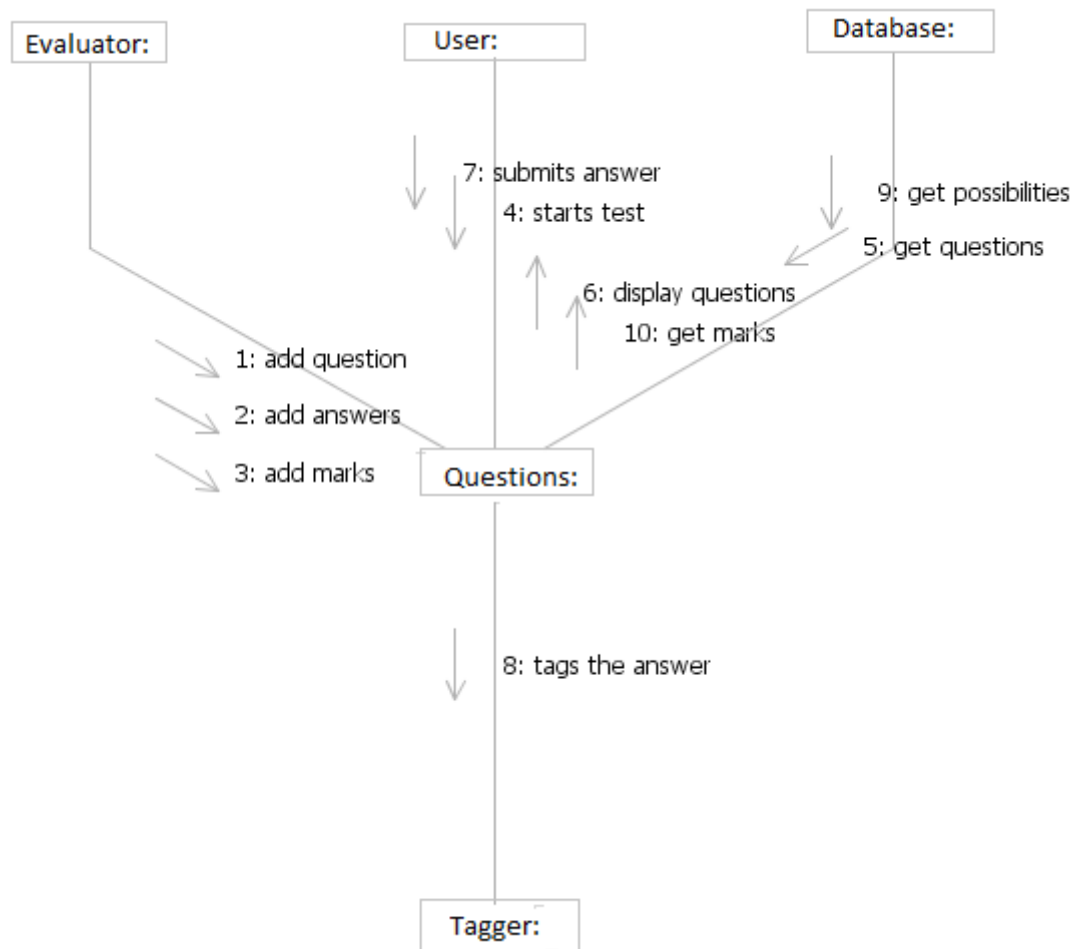


Fig.4.3 Collaboration Diagram

4.2.4 Activity Diagrams

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams are intended to model both computational and organisational processes (i.e. workflows). Activity diagrams show the overall flow of control.

Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

- rounded rectangles represent actions;
- diamonds represent decisions;
- bars represent the start (split) or end (join) of concurrent activities;
- a black circle represents the start (initial state) of the workflow;
- an encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen.

Hence they can be regarded as a form of flowchart. Typical flowchart techniques lack constructs for expressing concurrency. However, the join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions or loops.

In the FIG.4.4 User activities are submitting answer, getting marks. Whereas Evaluator activities are adding question and answer, and assigning weights to the edges.

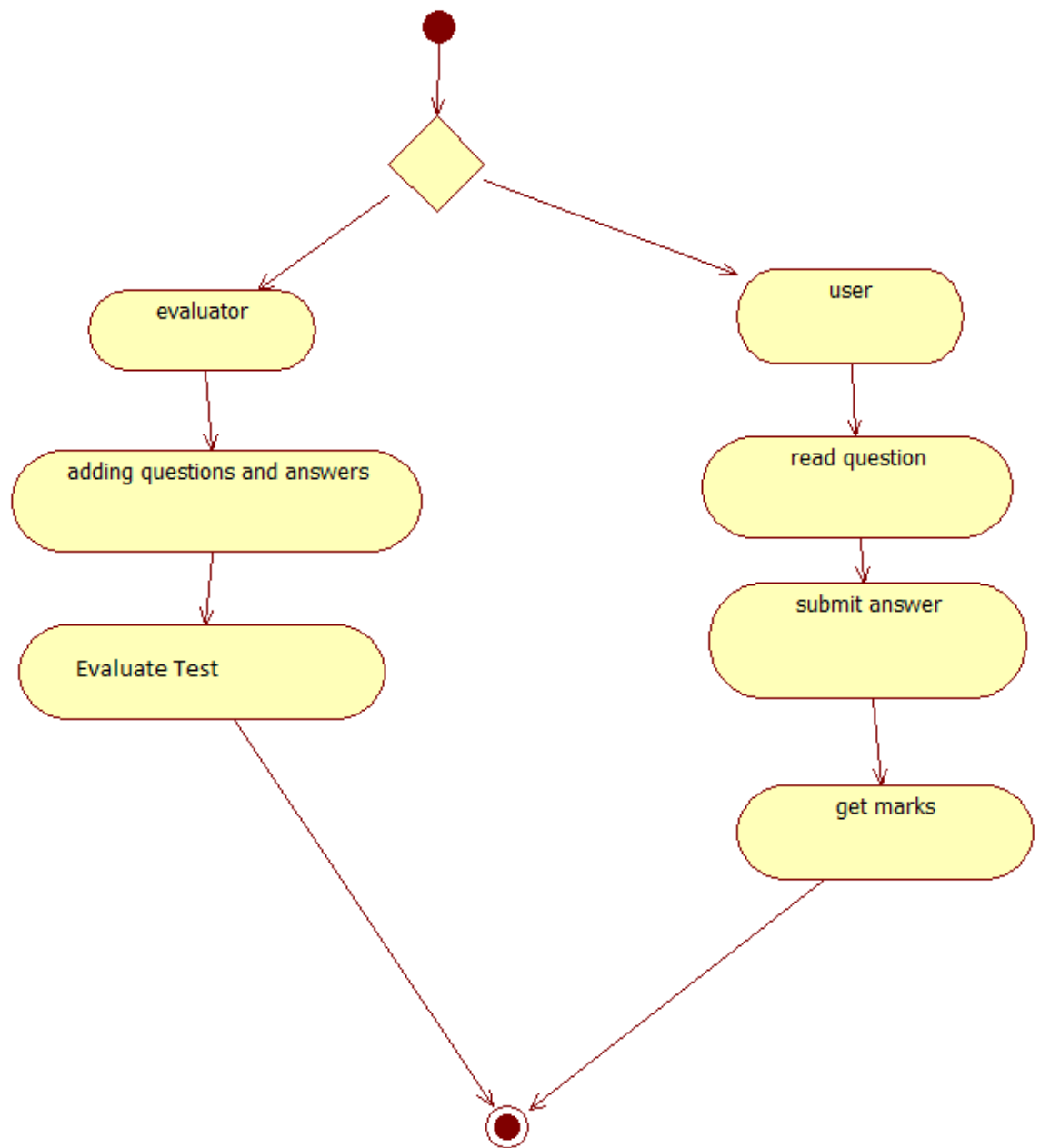


Fig.4.4 Activity Diagram

4.2.5 State chart diagram

In the Fig.4.5 the state transitions of user are starting the test , entering the answers and finally getting marks. State transitions in the evaluator are between states such as adding questions and answers, adding possibilities, assigning marks to each possibility.

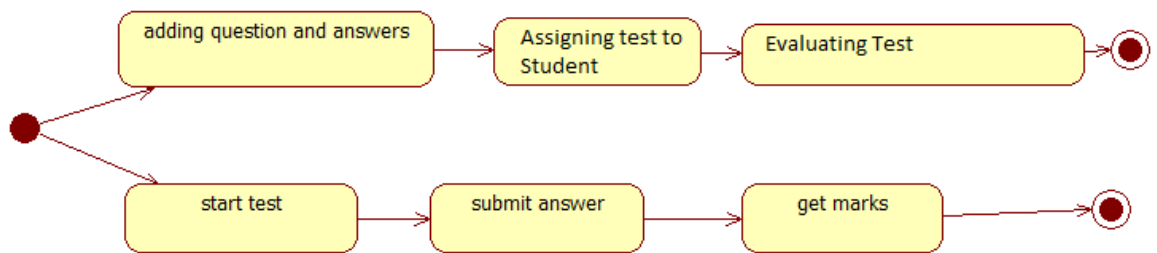


Fig.4.5 State Chart diagram

4.2.6 Component Diagram

Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model physical aspects of a system. The Fig.4.6 gives the components of the student evaluator system.

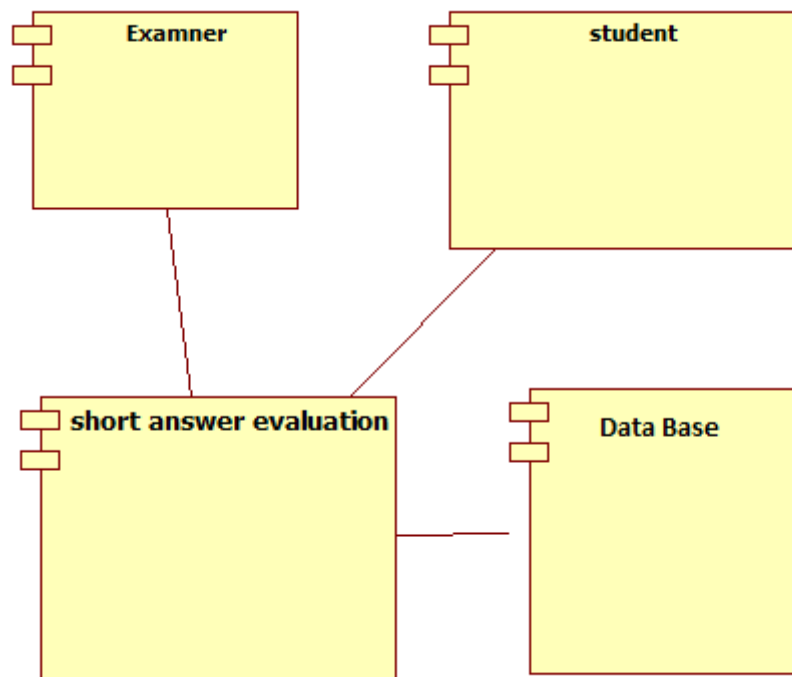


Fig.4.6 Component Diagram

CHAPTER – V

IMPLEMENTATION

5.1 Download the Apache Tomcat 8 server

We are going to create a Tomcat server locally to use for testing purposes, so need to download the relevant binary distribution core zip of Apache Tomcat 8.

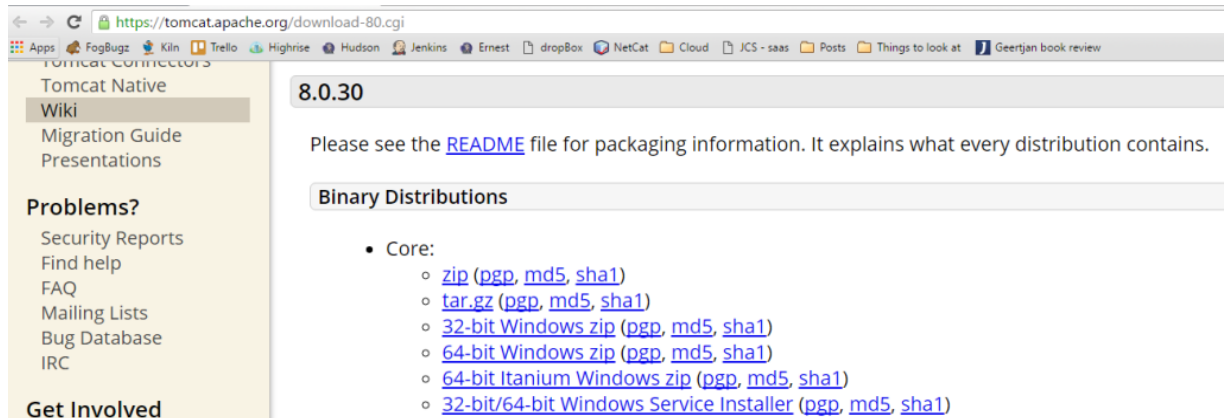


Fig.5.1 Tomcat Download

Fig.5.1 Extract the contents from downloaded zip and save in Program Files directory or optionally in another directory.

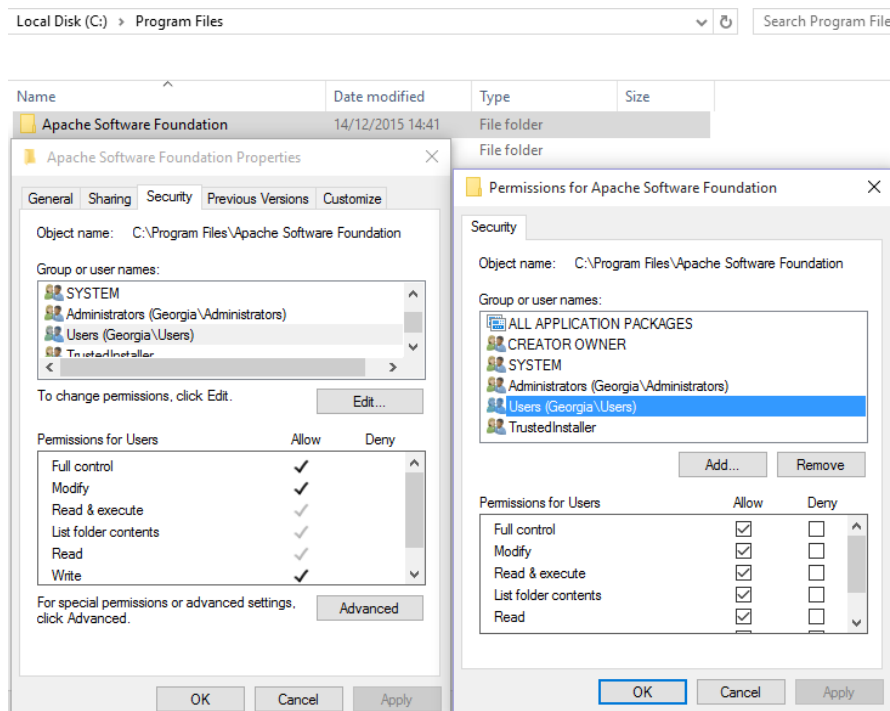


Fig.5.2 Tomcat Setup

Fig.5.2 To make sure we have the correct permissions, right-click top-most Tomcat folder in the Program Files directory and select properties. In the properties window go to the Security tab. Then select user account name and make sure all of permissions are set to Allow. If they aren't go to Edit and tick them and click OK. If these aren't set then Tomcat may not run correctly.

5.2 Add the Tomcat server to NetBeans

Once opened NetBeans go to Services window. Then right click on Servers node and select Add Server. When prompted select Apache Tomcat or TomEE and can name server at the bottom. Click Next.

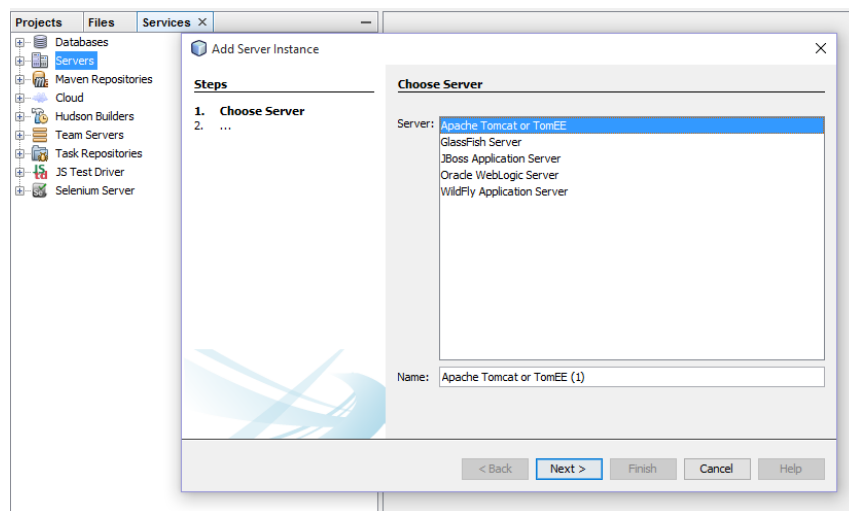


Fig.5.3 Connect Tomcat & NetBeans

Fig.5.3 Server location should be the folder containing all the Tomcat files including the bin, conf and lib folders. Next we need to create a user account for this server so that we can allocate permissions that allow to run, deploy projects, etc. To do this we add a username and password and leave Create user if it does not exist ticked which will create the account for . Click Finish.

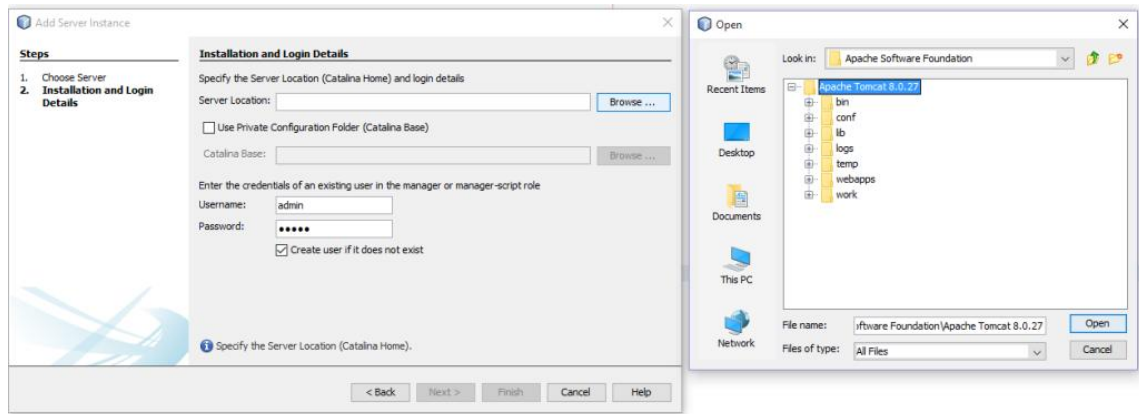


Fig.5.4 Start Tomcat

Fig.5.4 Now that we have created a Tomcat server instance that appears under the Servers node. Right click the server and select Start.

Not every time but running Tomcat on Windows may give a *"127.0.0.1 is not recognized as an internal or external command error"*. To fix this need to go to C:\ServerLocation\bin\catalina.bat and need to edit this file so open it up in a text editor. Search for the below two snippets and remove the speech-marks.

```
:noJuliConfig
"JAVA_OPTS=%JAVA_OPTS% %LOGGING_CONFIG%"

:noJuliManager
"JAVA_OPTS=%JAVA_OPTS% %LOGGING_MANAGER%"
```

Now save changes. Changing this requires to delete the server instance just created on NetBeans because have changed the servers configuration and will have to restart this step again in that case.

Another common error is that NetBeans may not create server login details correctly. To check this open this file in a text editor :C:\ServerLocation\conf\tomcat-users.xml. If this is not set correctly, then when we go to deploy an application locally, it will not accept the login credentials so we had to edit tomcat-users section of this file to resemble this:

```
<tomcat-users xmlns="http://tomcat.apache.org/xml" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.0" xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd">
```

```
<role rolename="manager-gui"/>
<user password="Password" roles="manager-gui,manager-script,admin" username="UserName"/>
</tomcat-users>
```

Again we need to delete NetBeans instance and add it to NetBeans again and start the server. One more issue may be if we want to run Tomcat when another process is running on port 8080. Port can be changed in C:\ServerLocation\conf\server.xml file. Just search for 8080 and change every instance to a different port number e.g. 8085.

5.3 Create a Web Application

To create a web application go to the Projects window, right click and select Create New Project. Then in the Java Web category select Web Application. On the next screen name the project. When come to the Server and Settings page set the Server to the Tomcat server we created in the above steps. The latest Java EE version supported in version 7. Click Finish and project will be created.

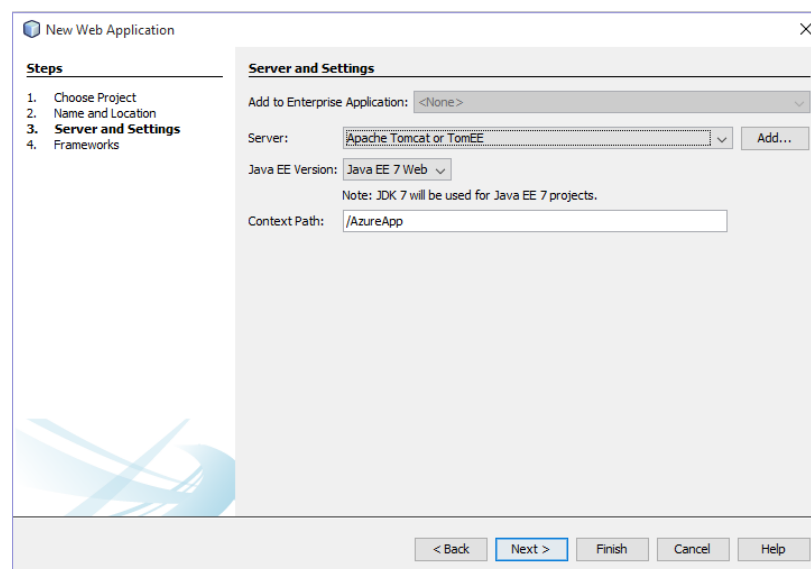


Fig.5.5 Create web application

Fig.5.5 depicts the process of creating a web application using NetBeans IDE and Tomcat Server.

5.4 Database Schemas

For local testing we used MySQL phpmyadmin of XAMPP as our backend database management system and later on the project database is exported to SQL Server Management Studio and then migrated as Azure SQL data instance.

Questions Table

Column	Type	Null
Question	varchar(1000)	Yes
Answer	varchar(1000)	Yes
Id	int(11)	No

Table 5.1 Questions Table

Test Table

Column	Type	Null
Question	varchar(1000)	Yes
Answer	varchar(1000)	Yes
Score	Float	Yes
Testname	varchar(30)	Yes
Username	varchar(255)	Yes
Id	int(11)	No

Table 5.2 Test Table

User Table

Column	Type	Null
Username	varchar(255)	Yes
Password	varchar(255)	Yes
Role	varchar(10)	Yes
Id	int(11)	No

Table 5.3 User Table

Status Table

Column	Type	Null
Username	varchar(255)	Yes
Status	varchar(10)	Yes
Id	int(11)	No

Table 5.4 Status Table

5.5 Evaluator Module

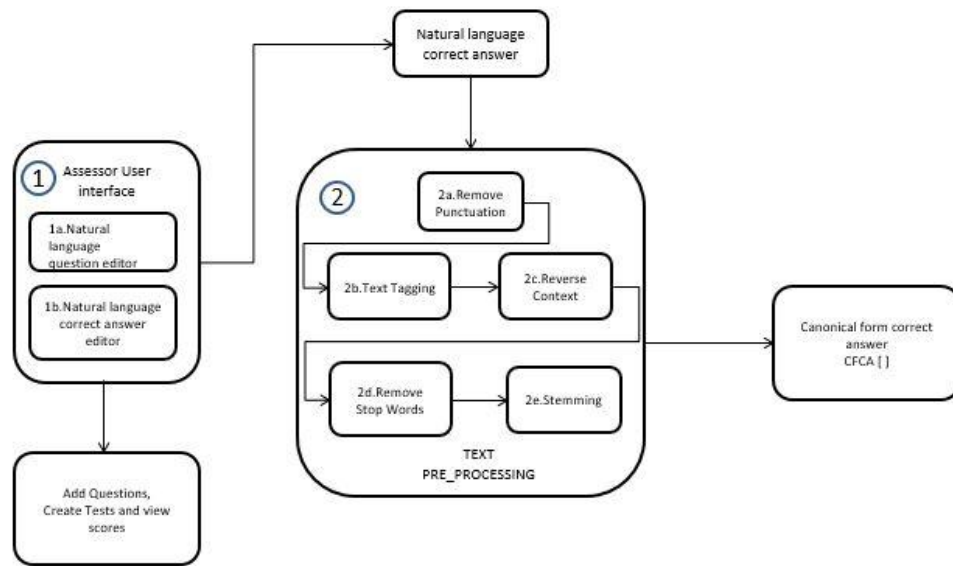


Fig.5.6 Evaluator Module

The evaluator module Fig.5.6 consists of the following

- Create test
- Add question
- Evaluate test
- Reports
- Add Users

5.5.1 Create test

The evaluator has to provide the test name, select the questions and assign multiple students who are supposed to take test. The questions and the student names are retrieved from the database. A table is created with the provided test name to keep track of student's status on completing the test and the test details are stored in the 'Test' table Table 5.2.

5.5.2 Add questions

The evaluator can add a question and corresponding model answer which are stored in the 'Questions' table Table 5.1 with the pre-processed answer. The question added by the evaluator is can be used at time when the evaluator creates a test.

5.5.3 Evaluate test

The evaluator has to select one of the test that he has created which is retrieved from 'Test' table Table 5.2 to view the test details and ensure complete status of all the assigned students for that particular test from 'Status' table Table 5.4 and then evaluate the test . Each student answer text goes through evaluation process i.e. Text Pre-processing, matching and marks calculation and updated in the 'Test' table. Finally the respective result is available to each student of that test.

5.5.4 Reports

Here the evaluator can view the details of all the tests and scores of each student with corresponding questions that were assigned to him i.e everything that is in the 'Test' table Table 5.2 .

5.5.5 Add Users

The evaluator can view the current users along with their role Admin/Student and can enter a new user entry into the 'Users' table Table 5.3.

5.6 User module

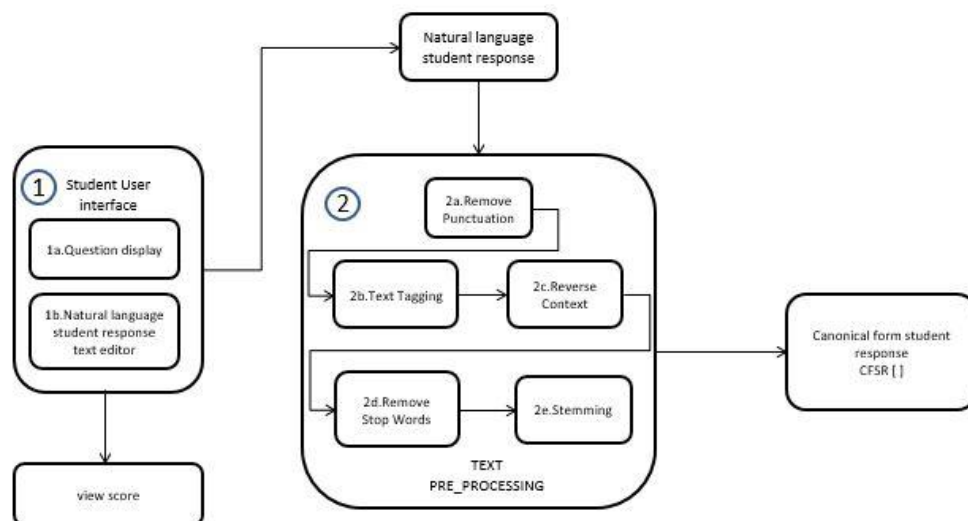


Fig.5.7 User Module

The following are the steps in User module Fig.5.7:

- Reading questions
- Submitting answers
- View result

5.6.1 Reading questions

The student has to select one of the test that he was assigned with and take that test where he will be displayed with questions which are retrieved from 'Test' table along with text areas to write the answers.

5.6.2 Submitting answers

The student after answering all the questions submits the test which in turn updates the 'Test' table and 'Status' table. Student is also provided clear button to clear the answer when needed.

5.6.3 View result

After submitting the answer, the student has to wait until the evaluator has evaluated the answers to view the result.

5.7 Text-Pre-processing

Both the model answer and student answer are first pre-processed before they could be scored. Following are the steps in pre-processing phase:

- Remove punctuations
- Text tagging
- Reverse Context
- Stemming
- Remove stop words
- Remove duplicates

5.8 Deploying into Cloud

The following are the steps to deploy the java web application into Microsoft Azure cloud:

- Sign up for an Azure free trail account

- Migrate the local database to SQL azure
- Change the database connection string in application
- Create a Tomcat server instance in Azure interface
- Upload the WAR file

5.8.1 Sign up for an Azure free trail account

The trial is available for 30 days with no option to extend the trial like some other cloud providers offer, on the other hand though receive \$125 worth of credit to use over all of the cloud services. The AWS trial was quite restricted by the functionality could access but this trial offers the whole package. After the trial expires can optionally look into using the pay-as--go option if the 30 days weren't enough, They also have a cool Pricing Calculator to help estimate the pay-as--go expenses.

We can access the portal by the link <https://portal.azure.com/> and signing into the account created when signing up for the Azure free trial. Below is a picture of the portal. have the dashboard in the main section which can pin any frequently used utilities or applications.

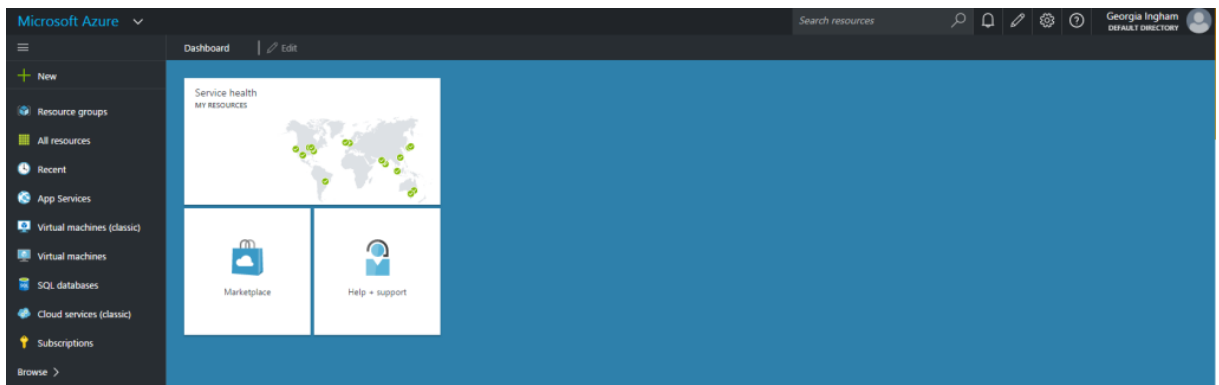


Fig.5.8 Azure Portal Dashboard

5.8.2 Migrate the local database to SQL azure

1. Download the SQL Azure Migration Wizard: <http://sqlazuremw.codeplex.com/>.
2. Launch **SQLAzureMW.exe**.
3. Under **Analyze and Migrate**, select **SQL Database** and click Next.
4. Connect to local server.

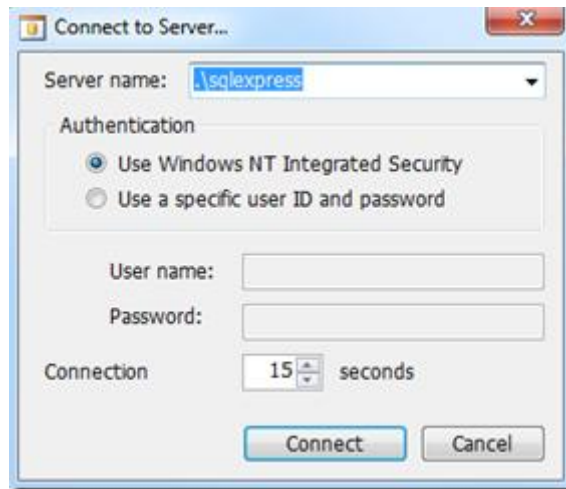


Fig.5.9 Source Database

5. Select the database we wish to migrate (say Northwind).
6. Select Script all database objects.
7. In the **Script Wizard Summary** pane, click **Next**.
8. Click **Yes** when asked if are ready to generate the script. The wizard will create a script that we will then execute to create a database (with its data in tact) in SQL Azure.
9. When the wizard finishes creating the script, click **Next**.
10. In the next pane, will connect to SQL Azure server. will need to replace SERVER with server ID, replace UserName with the user name used when creating the server, and supply the password for that user.

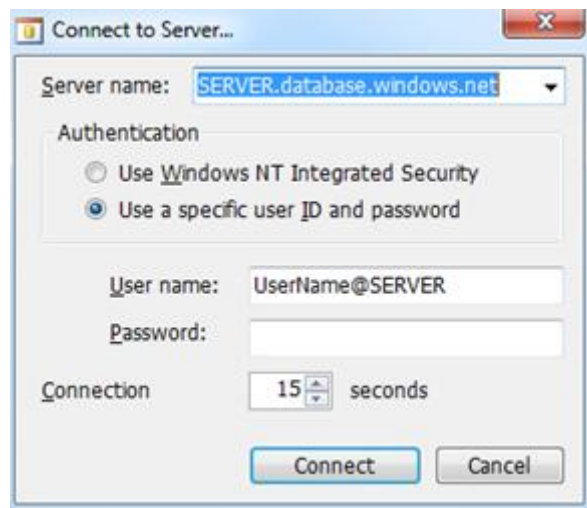


Fig.5.10 Target Database

11. After have connected to the server, click **Create Database** and supply the name of the database want created.
12. Select the newly created database, and click **Next**.

13. Click **Yes** when asked to execute the script against the destination server.

5.8.3 Change the database Connection String in application

Once have moved local database to SQL Azure, only have to change the connection string in application to use SQL Azure as data store. In my case (using the Northwind database), this meant changing this...

```
String connectionString = "jdbc:sqlserver://serverName\\sqlexpress;"
                          + "database=Northwind;"
                          + "user=UserName;"
                          + "password=Password";
```

...to this...

```
String connectionString = "jdbc:sqlserver://xxxxxxxxxx.database.windows.net;"
                          + "database=Northwind;"
                          + "user=UserName@xxxxxxxxxx;"
                          + "password=Password";
```

(where xxxxxxxxxxxx is SQL Azure server ID).

5.8.4 Create tomcat server instance in Azure interface

We need to create a Tomcat server on Azure which we will deploy our application to. To do this go to New and in the search bar at the top of the new window input Tomcat 8 and hit enter.

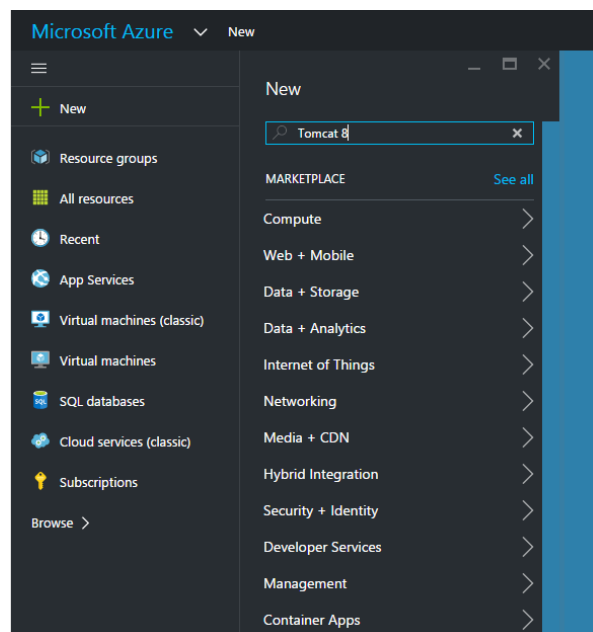


Fig.5.11 Azure Marketplace

Select the Apache Tomcat 8 option, which will open a new window and click Create and another window opens! Where we can enter the applications name here. The name has to be globally unique so might have to put a slightly odd name in. Once we have done that make sure Pin to dashboard is ticked and click Create.

Fig.5.12 Azure App Name

Fig.5.13 App Settings

access it by going to <http://AppName.azurewebsites.net/>

5.8.5 Upload the WAR file

To create a WAR file go to **File** view. Right click **build.xml** and go to **Run Target** and select **dist**. This will create a dist directory with a WAR file for application.

Going back to the Azure Portal now. Select web application that created earlier (can find it under the App Services menu item on the left if have closed it). Now in web applications menu select Tools which will open another window. Under the DEVELOP menu select Kudu (which opens yet another window) and click Go. Kudo is a tool that lets us view and modify servers file system. We are going to use it to upload the WAR

CHAPTER – VI

RESULTS

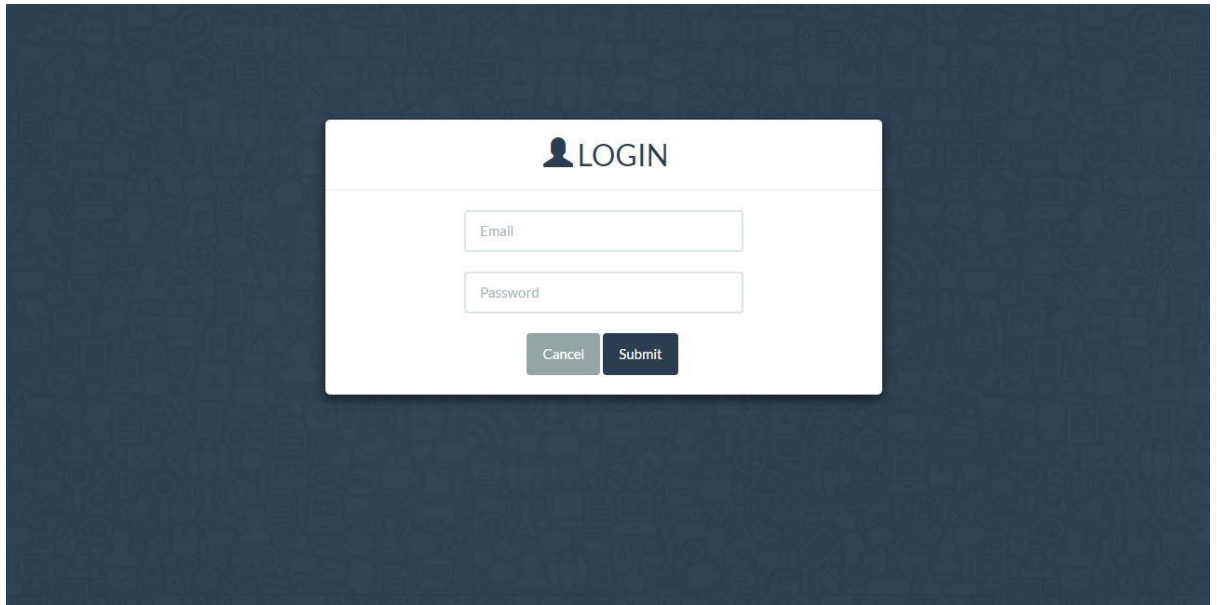


Fig.6.1 Default Login Page

Fig.6.1 shows us the basic login page. Depending upon the type of login Admin/Student the user is redirected to the appropriate page.

EVALUATOR INTERFACE

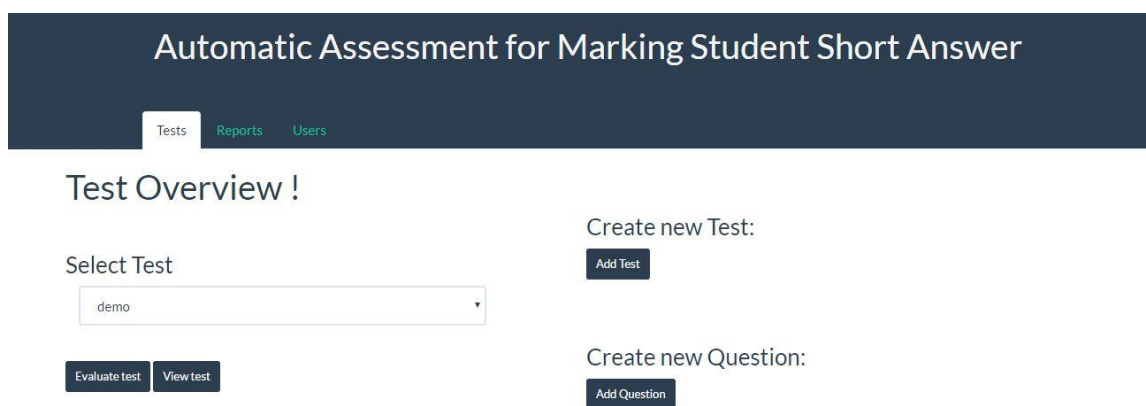


Fig.6.2 Evaluator UI

The Admin interface has 3 tabs above Tests, Reports, Users where Tests tab includes the core functionalities like adding question, crating test and evaluating the tests.

Automatic Assessment for Marking Student Short Answer				Logout
Tests	Reports	Users		
user1				
Testname	Question	Answer	Score	
Test1	What is a variable ?	A variable is a area in memory that contains values.	3.71429	
Test1	What is Recursion ?	Something calling itself is called recursion.	0	
Test3	What is a variable ?	A variable is a area in memory that contains values	3.71429	
Test3	What is a Pointer ?	A pointer is a element that keeps the address of another variable.	4.77778	
user3				
Testname	Question	Answer	Score	
Test1	What is a variable ?	A location in memory where data can be stored and retrieved.	3.33333	
Test1	What is Recursion ?	A function inside a function.	0	
user2				
Testname	Question	Answer	Score	

Fig.6.3 Report Tab

This is the sample report which shows each user's result for each question to the respective test name.

Automatic Assessment for Marking Student Short Answer			Logout
Tests	Reports	Users	
user1	pass1	Student	
user2	pass2	Student	
user3	pass3	Student	
user4	pass4	Student	
user5	pass5	Student	
user6	pass6	Student	
<input type="text"/>	<input type="text"/>	Admin	
Add New User			

Fig.6.4 Users Tab

In the users tab of the Admin interface current users can be viewed and new users can be added.

Fig.6.5 Evaluator Add test Screen

Interface for test creation which is a part of Test tab where interacting with all the fields is a must before saving it.

Fig.6.6 Evaluator storing question

Fig.6.6 shows the interface of the evaluator storing the question. Evaluator can add any number of questions which are stored in the database for later on use in test creation.

STUDENT INTERFACE



Automatic Assessment for Marking Student Short Answer

Tests

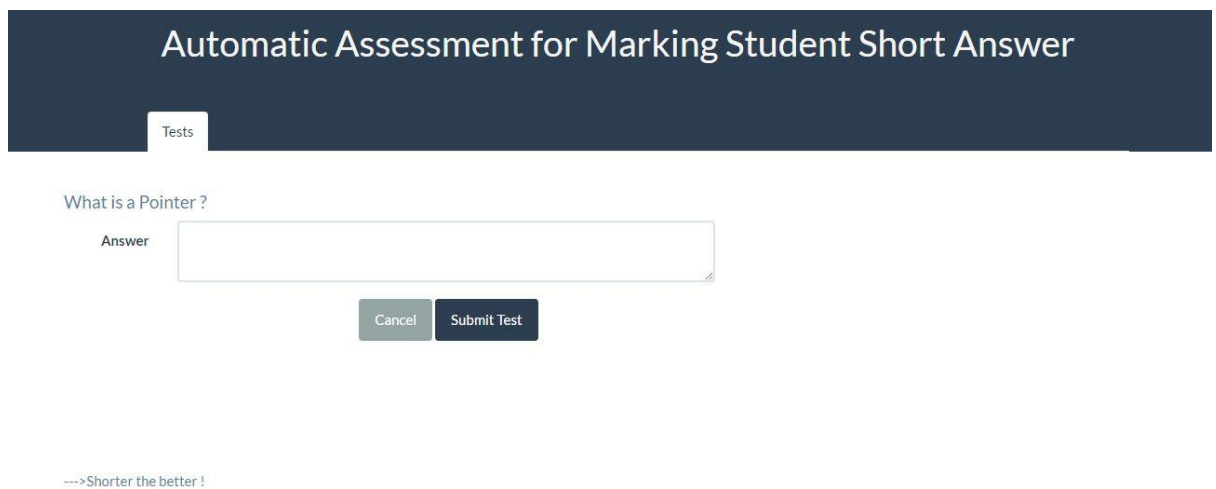
Select Test

demo

Take test View result

Fig.6.7 Student UI

Fig.6.7 is the default login page of the user. The student has to select the test name that he hasn't attempted in the list of tests that are list box. He can attempt the test or view result.



Automatic Assessment for Marking Student Short Answer

Tests

What is a Pointer ?

Answer

Cancel Submit Test

--->Shorter the better !

Fig.6.8 Take Test

Fig.6.8 shows the Student test interface that he was assigned with by logging in with his credentials.

Automatic Assessment for Marking Student Short Answer			
		Change password	Logout
Testname	Question	Answer	Score
Test2	What is function signature ?	A function signature is the functions parameters and their type and the name of the function.	4.83333
Test2	What is function pointer ?	A pointer to a function itself, contains the address of the function and can be used to call that function.	4

Fig.6.9 View Result

Fig.6.9 shows the Score that is made available after the evaluator has evaluated the answer in evaluator interface

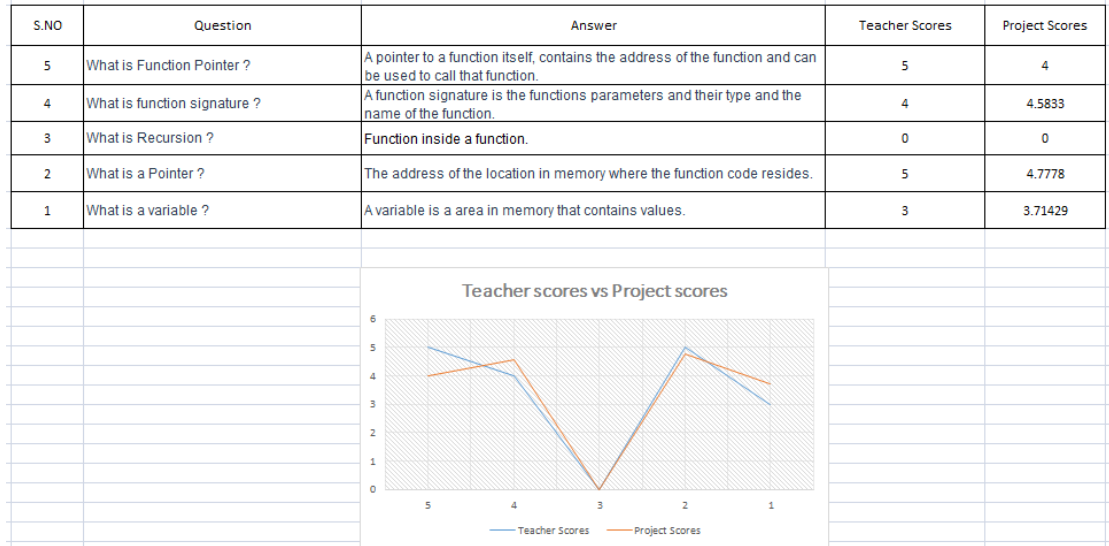


Fig.6.10 Graphical Anaysis

Fig.6.10 depicts the graphical analysis of the project scores vs the evaluator scores. As the graph depicts the difference between the expert answer and software given answer is less hence the evaluator can be relied on.

MOBILE SCREENSHOTS

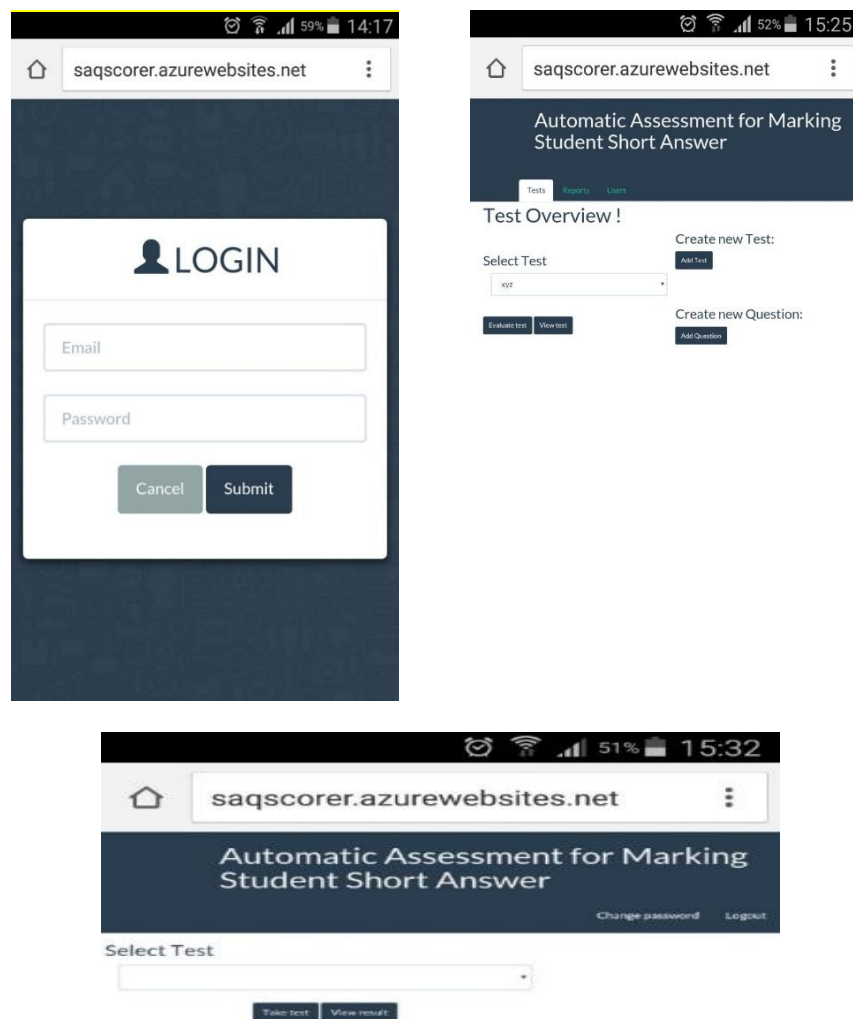


Fig.6.10 Mobile Interactions

In the Fig.6.10 as the app was deployed into cloud it could be accessed from anywhere with internet connectivity and also with the bootstrap framework which was used while developing this cloud-based web application made it responsive to adjust the component sizes with respect to smartphones.

CHAPTER – VII

CONCLUSION AND FUTURE WORK

Research proposes the combination of different techniques to produce an approach for automatic short answer marking. In recent years, a number of automated marking systems for program texts, i.e., texts written in a programming language, have been developed. However, these systems cannot mark short answers expressed in natural language. Short-answer questions provide a very useful means of testing theoretical concepts associated with a programming course.

The salient feature of our project (from the pedagogical perspective) is that it can provide tests and immediate marks to students regardless of the size of the class. The lecturer has to initially spend some time developing and validating the required structures when a test is conducted for the first time. But, once the required structures are finalized, the same practice test may be repeated wherever the same course material is taught. The lecturer conducting the test, after the first test, does not need to spend any time manually marking student's tests. The feedback feature which specifies exactly where the user has gone wrong in answer and spell checking feature while user writing his question can be added to improve the efficiency of our application. This system has been developed for the evaluation of English Language only. Looking forward to implement in other languages also developing a common methodology for evaluation.

REFERENCES

- [1] L. Cutrone, M. Chang, and Kinshuk, "Auto-Assessor: Computerized Assessment System for Marking Student's Short-Answers Automatically," Proceedings IEEE International Conference on Technology for Education (T4E 2011), July 14-16, 2011.
- [2] Leacock C., Chodorow M. C-rater: "Automated Scoring of Short-Answer Questions".Computers and Humanities 37, 2003.
- [3] L. Cutrone and M. Chang, "Automarking: Automatic Assessment of Open Questions," Proceedings IEEE International Conference on Advanced Learning Technologies (ICALT 2010), July 5-7, 2010, pp. 143-147
- [4] WordNet.NET, Computer Software, Ebswift, available online at <http://opensource.ebswift.com/WordNet.Net/>
- [5] Tom Mitchell, Nicola Aldrige, and Peter Broomhead.2003." Computerized marking of short-answer free-text responses". Paper presented at the 29th annual conference of the International Association for Educational Assessment (IAEA), Manchester, UK.
- [6] Larkey, L. S. (1998), "Automatic essay grading using text categorization techniques" , in 'Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval'.
- [7] <http://rednoise.org/rita>
- [8] <http://getbootstrap.com>
- [9] http://en.wikipedia.org/wiki/microsoft_azure

APPENDIX

Text Pre Processing code :

```
String[] stopwords={"the", "is", "are"}; //SO ON ADDED AROUND 300 STOP  
WORDS
```

```
String[] penn={ "cc","cd","dt"}; //SO ON ADDED ALL THE TAGS OF PENN  
TREEBANK
```

```
Boolean reverse;
```

```
public final String[] preproc(String str)  
{  
    int k=0;  
    str=str.toLowerCase();  
    str=RiTa.stripPunctuation(str); //REMOVE ANY PUNCTUATION  
    if(str.contains("was")||str.contains("by")){reverse=true;}  
    //str=RiTa.getPosTagsInline(str); //TEXT TAGGING
```

```
    MaxentTagger tagger = new MaxentTagger("C:/Users/SAI  
CHAITANYA/Downloads/stanford-postagger-2015-01-30/models/english-  
left3words-distsim.tagger");
```

```
    str=tagger.tagString(str);  
    str=str.toLowerCase();  
    String regex,cfsr[];  
    ArrayList <String>a=new ArrayList();  
    ArrayList <String>b=new ArrayList();  
    ArrayList <String>c=new ArrayList();
```

```
    //REMOVE STOPWORDS
```

```
    for(String wrd: stopwords)  
    {
```

```

for(String pen:penn)
{
    regex = "\\s*\\b"+wrd+"_"+pen+"\\b\\s*";
    str=str.replaceAll(regex, " ");
    str=str.replace("$", "");
}
}

```

//STEMMING

```

cfsr=RiTa.tokenize(str);
String temp[];
for(String i:cfsr)
{
    temp=i.split("_");
    if(temp[1].startsWith("j"))
    {
        temp[1]="a";
    }
    else if(temp[1].startsWith("n"))
    {
        temp[1]="n";
    }
    else if(temp[1].startsWith("v"))
    {
        temp[1]="v";
    }
    if(temp[0].endsWith("y")||temp[0].endsWith("e")||temp[0].endsWith("ion")){
        i=temp[0];
    }
}

```

```

else{
    i=RiTa.stem(temp[0]);
}
i=i+":"+temp[1];
cfsr[k++]=i;
}

// SWAPPING SUBJECT AND OBJECT
if(reverse)
{
    int i=-1;
    for(String w:cfsr)
    {
        i++;
        if(w.contains("v")){break;}
    }
    for(int j=0;j<i;j++)
    {
        a.add(cfsr[j]);
    }
    for(int j=i+1;j<cfsr.length;j++)
    {
        b.add(cfsr[j]);
    }
    for (String b1 : b) {
        c.add(b1);
    }
    c.add(cfsr[i]);
    for (String a1 : a) {

```

```

        c.add(a1);
    }
    cfsr=c.toArray(new String[0]);
}
//REMOVE DUPLICATES
cfsr=new TreeSet<String>(Arrays.asList(cfsr)).toArray(new String[0]);
return cfsr;
}

```

Sample Input: “The QUIck brown fox! jumped over@ the, lazy Lazy dog”

Output Canonical form array: [brown:a, dog:n, fox:n, jump:v, lazy:a, quick:a]

Note: v=verb a=adjective n=noun

The model answer and student answer is first pre processed like above and are compared for an exact match. An exact match is determined based on:

- a) *A matching part-of-speech tag*
- b) *A word match*