International Institute of
Information Technology, Bangalore

Automated Quiz Generation
And
Evaluation

Mentor: Dr Sujit Kumar Chakrabarti

Keshav Gupta
MT2019053

Eshita Sharma
MT2019037

Lubaina Machinewala
MT2019057

INTRODUCTION

This application also known as EvalObj is an automated quiz generation and evaluation. Since every field is moving towards distant learning so automated evaluation has become the need of an hour. It can generate quizzes unique to each candidate minimizing the risk of mass cheating and making the examination process more smooth.

There are many existing systems, doing the same thing, but all of them have few drawbacks which are considered in this application. Few of the drawbacks of existing system are:

1. Need for Internet access during administering quizzes - All of the existing system needs an internet connection for the assessment. But the majority of people live in rural regions and good internet connection is a far fetched dream. EvalOBj need not require the user to be connected to the internet throughout the test. Instead a student needs internet access only twice, once while downloading the quiz packages, next while uploading the solution file.

2. Use of Database for storing data - Most of the systems use an internal database to store all of their data. The data is often locked into the system. When it comes to repairing or analysing the data, it becomes quite hard to do so on these databases. EvalObj uses a file system to store all its data. The entire data is available with the instructor at all points in time. This saves a lot of hassle when you want to analyse or repair the data.

3. Lack of flexibility - Most of the systems are rigid in adopting the instructor's view of assessment and evaluation. They have a set algorithm for creating quiz and quiz type and don't have ways to allow new ways of assessment and evaluation. EvalObj is flexible in its assessment and evaluation algorithm. It’s design pattern allows users to add new methods according to the instructor without much difference in code.

4. Cheating in exams - According to the studies by Erasmus Magazine , texting during examinations is the way of cheating. The number of suspected cases of cheating during online exams is on the rise. But this could be reduced to a great extent if each student is given a different question paper. Communicating with other students to find the right answer will be significantly more difficult in such scenarios. EvalObj has a large set of item banks, so it’s possible to make totally new question papers for students to reduce cheating.

Hence EvalObj handles the problem of existing systems and provides a better application for assessment and evaluation.

IMPLEMENTATION

EvalObj system is divided into three parts:

1. Creating question papers
2. Carrying out the examination
3. Evaluating the responses

Creating Question Papers:

There are two types of questions that are included in the question papers: - As of now, two question type classes have been implemented:

- multiple choice questions (MCQ)
    - An MCQ is a question with two or more possible options out of which one or more may be correct choices. For example:
      Question - Name two dynamically typed programming languages:
      1. Java
      2. Python
      3. Haskell
      4. Ruby
      In the above, options 2 and 4 are correct answers.

- Match the following (MTF)
    - Match the following programming languages on the left column with properties on the right.

	A. Static typing
1. Python	B. Dynamic typing
2. Java	C. Untyped
3. C++	D. Implicit typing
4. OCaml	E. Explicit typing
	F. Functional
	G. Object oriented

In the above example, the matches are as follows

1. Python matches with B., D., F. and G.
2. Java matches with A., E. and G.
3. C++ matches with A., E. and G.
4. OCaml matches with A., D., F., and G.

A few points to note here:

- The mapping can be many to many, even allowing some options on either side to have no matches at all in MTF type questions.
- There is no restriction on the number of options on either columns.
- The code is flexible enough to accomodate paper to answer, which contains questions sampled out of a larger item bank and jumbled.any other question type needed for the instructor. The instructor just needs to add his/her question type class.

This software supports two types of objective question quizzes:

- Simple quiz - In this quiz, the students are supposed to answer all questions. All students answer the same question paper
- Jumbled quiz - In this form of quiz, each student gets a different question

Simple Quiz

The question paper creation for a simple quiz is manual. All students solve an identical question paper. The evaluation step directly runs on the student responses using the SimpleEvaluator module.

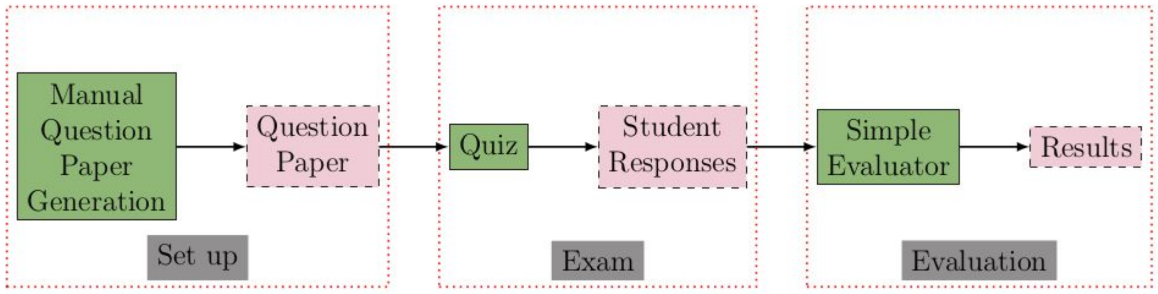


Fig 1: *Simple Quiz Workflow*

Setup

As mentioned above, in a simple quiz, all students solve the same question paper. Such a quiz is ideal when there is small class and there is assurance that cheating is not a possibility. Of course, all questions are assumed to be either multiple choice questions (MCQ) or match the following questions (MTF). The assessment project can be set up conveniently using the setup utility. The setup utility automatically generates the skeletal infrastructure for conducting such a quiz.

Jumbled Quiz

The question paper creation step uses the genAIs (generate assessment items) and genQPs (generate question papers) modules. Once the question papers are generated, a quiz is conducted. Finally, the automated evaluation process takes place using the JumbledEvaluator module.

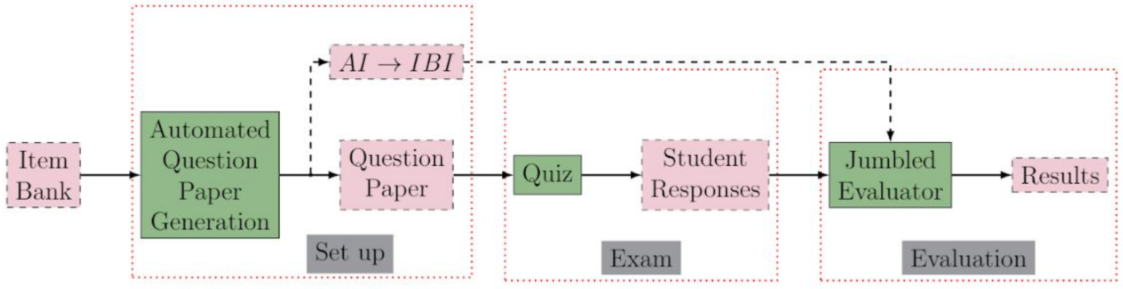
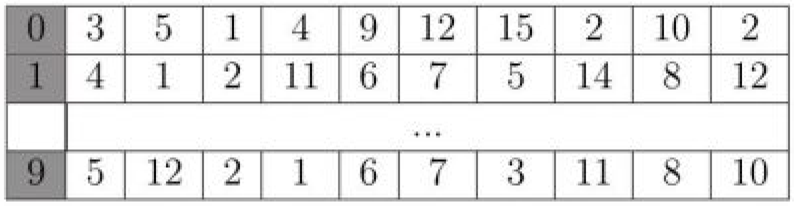


Fig 2: *Jumbled Quiz Workflow*

Python is used to construct this software and Pdf Latex module for generating the question papers. To discourage cheating in the class, we generate a set of question papers by randomly selecting n questions out of an item-bank of N questions. A set K of distinct assessment instruments are generated, numbered 0, 1, ..., |K − 1|, we call them assessment instruments. The question paper generator module G generates a set C of codes each of which can be mapped to any one of the assessment instruments of K. Each of the code c in C is finally mapped to one distinct question paper with c printed on it. This way, the students will not be able to identify which assessment instrument k ∈ K their copy of the question paper belongs to. Each question paper will have an empty table called the response table on page one which will be used by the student to fill in his responses. G also generates a map from assessment instrument to question order. This tells us the original question number of each item in the given assessment instrument.



0	3	5	1	4	9	12	15	2	10	2
1	4	1	2	11	6	7	5	14	8	12
	...									
9	5	12	2	1	6	7	3	11	8	10

Assessment Instrument Item to Item Bank Item map AII→ IBI

For example: Figure shows a possible mapping from assessment instruments to item bank items. The table can be interpreted as follows: There are 10 assessment instruments numbered 0 through 9. For each assessment instrument AI ∈ K (here |K| = 10), there is a row in the table. Each cell in that row has the item bank item number for that item. For instance, for AI = K[0], AI[0] = 3, AI[1] = 5 and so on. This module will generate a map – called QP → AI between question paper code to assessment instrument. For example: QP → AI = [0, 1, 2, ..., 9, 0, 1, 2, ...] could be one such mapping. It says that QP→ AI[0] = 0 (i.e. the question paper with code 0 maps to assessment instrument number 0). Similarly, QP → AI[11] = 1 (i.e. the question paper with code 11 maps to assessment instrument number 1) and so on.

Carrying out the examination
At the time of administering the quiz, these question papers can be shared with the students using an appropriate external method, e.g. email, LMS or a shared drive. This is outside the preview of this system.
Tkinter package of python is used to create the graphical user interface for python.Student is allowed to review or change his/her answer for a particular question any number of times. A response csv file will be created after the user's final submission.This csv file can then be shared with the administrator which can then be used for further evaluation.

Evaluating the responses
The response rearranger refers to the RN → QP and QP → AI map to extract the assessment instrument for each roll number. Using this, the evaluator rearranges the responses in the order as per the item bank to create a rearranged response for the roll number n, R' (n). This is given to the evaluator for final automated evaluation This software was used to conduct a Python Quiz for M.Tech 2020 Batch in IIIT Bangalore. We created an item bank of 55 questions and each assessment item was mapped with random 20 questions from the item banks. Question paper and required files such as instructions, python script for graphical user interface (to record the response) were generated for each of 200 students, later the responses were recorded from the student and evaluated after fetching the previous mapping of assessment items.

In case of jumbled quiz the response rearranger refers to the RN → AI map to extract the assessment instrument for each roll number. Using this, the evaluator rearranges the responses in the order as per the item bank to create a rearranged response for the roll number n, R' (n). This is given to the evaluator for final automated evaluation.

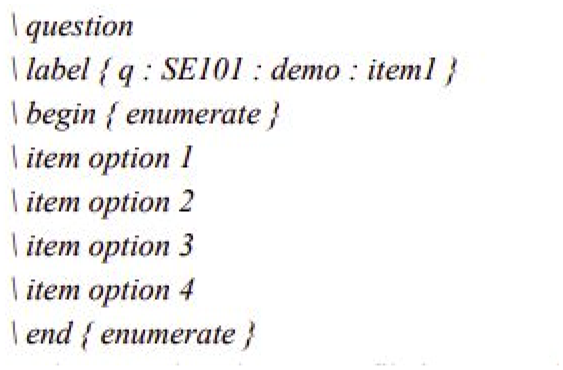
This system has been used to conduct numerous quizzes in IIIT Bangalore ranging from very small scale(<10 Questions,<20 Students), to medium to large scale(>20 Questions, >150 Students).The system creates an item Bank of all the questions and each assessment item is mapped with the desired number of questions in the final question paper from the item banks. Question paper and required files such as instructions, python script for graphical user interface (to record the response) is generated for each student, later the responses are recorded from the student and evaluated after fetching the previous mapping of assessment items.

DEMONSTRATION

1. **Generate the quiz**

   This generates all the necessary files for the quiz. Let's take a look at the contents of the directory:

   - Assessment-instruments:
     This is the directory created – currently empty – to contain the question papers once they are generated.
   - config.py:
     This is the file which contains the Python translation of the contents of the config.json file.
   - Evaluation:
     This is the evaluation directory. The evaluate.py file is the evaluator file which will be run once the quiz is conducted and submissions have come in.
   - gen.py:
     This script needs to be run next to generate the quiz.
   - item-bank:
     This contains the item bank, in the form of stubs of LATEX text. These need to be filled up to create the question papers. Note that there are five item stubs created, since that's the number of items mentioned in config.json.
     For example, the contents of the item1.tex is:



```
\ question
\ label { q : SE101 : demo : item1 }
\ begin { enumerate }
\ item option 1
\ item option 2
\ item option 3
\ item option 4
\ end { enumerate }
```

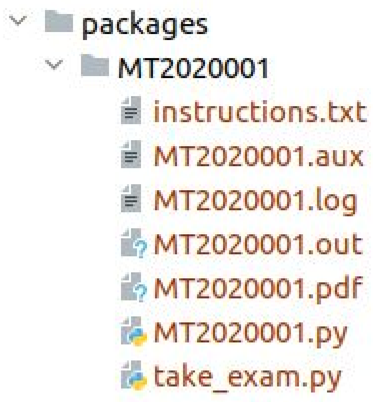
   - Packages: When the gen.py file is executed, it is this directory – currently empty where the question papers will be generated.

2. **Generate the Question Papers**

   The main effect of running this script would be that the packages directory is no more empty. It contains one directory for each roll number. In there, among other files, you will find the question paper for that roll number:

packages/rn1/rn1.pdf
packages/rn2/rn2.pdf
packages/rn3/rn3.pdf
packages/rn4/rn4.pdf

Each package will contain a question paper for that roll number in the pdf form, along with a python file take_exam.py. When take_exam is executed, a graphical user interface application starts running. This application can be used to record the answers while giving the quiz. When the test is complete, a csv file is generated automatically from the GUI that has all the answers given by the individual. Along with these files, many other helpful files will be there in the package, like instructions.txt.



```
packages
  MT2020001
    instructions.txt
    MT2020001.aux
    MT2020001.log
    MT2020001.out
    MT2020001.pdf
    MT2020001.py
    take_exam.py
```

3. **Administer the Quiz**

At the time of administering the quiz, these question papers can be shared with the students using an appropriate external method, e.g. email, LMS or a shared drive. This is outside the preview of this system.
At the end of administering the quiz, the submitted answers are provided in the submissions directory in a particular format. In this demo, we simulate the process of answer submission by running the backup script provided.

4. **Evaluate the Quiz**

There is a separate module for evaluation. It contains evaluation for MCQ and MTF questions. It is flexible to accommodate any other question class evaluation in it. Marking scheme can be adjusted as per the instructor's wish.
Evaluation process can be done in the following manner:

- Students csv answer sheets generated by the GUI program are kept in a folder named submission in the evaluation folder itself.

- Along with students' answer sheets, the correct answer of every question in the question band also needs to be provided in the csv format. This file is also present in the evaluate folder itself.
- Finally the evaluate.py file is executed, which calls the evaluator.py file from the source directory. This file has all the classes for evaluation.
- After calculating marks for all the students, a csv file is generated containing marks of all the roll numbers.

CONCLUSION

It is an automated quiz generation and evaluation tool which is used to conduct a quiz with different formats such as match the following, multiple choice questions etc. Instructors can input all the details in one json file to set the quiz for the entire class and folders can be generated for each student as specified by the instructor. This software includes generation of assessment instrument from item bank mapped randomly for each student in pdf format, graphical user interface developed to input responses from the student which can be done without using any external server and automated evaluation of quiz by fetching the previous mapping and generating the report. Hence this tool is used to conduct offline exams regardless of the need for the students to be connected to an uninterrupted internet connection. And also it provides an extensible design that allows us easy extension of the tool to accommodate many other question types without having the need to go through all the code for rigorous changes

REFERENCES

[1] https://docs.python.org/3/library/tkinter.html

[2] https://docs.python.org/3/library/csv.html

[3] Evaluation system like DomJudge, HackerRank, Moodle