

# ***Question paper Generation system***

## ***Project Demonstration***

-Haritha.R

### **Speciality of the software:**

Generates unique question papers with different questions.

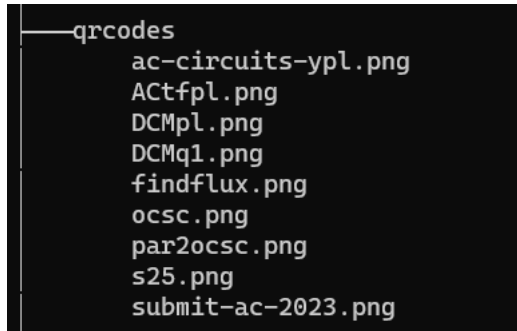
### **Architecture:**

We have 2 files “exam-randomizechoices.sty” and “run.py” and 3 folders

```
C:\Users\Ravivarma\Downloads\Question_paper_generation_software>tree /f
Folder PATH listing for volume OS
Volume serial number is B6C9-0379
C:..
  exam-randomizechoices.sty
  run.py
  qrcodes
    ac-circuits-ypl.png
    Actfpl.png
    DCMpl.png
    DCMq1.png
    findflux.png
    ocsc.png
    par2ocsc.png
    s25.png
    submit-ac-2023.png
  questionbank
    AAABB
      AAABB.json
      AAABB.py
      AAABB.tex
    AURTX
      AURTX.json
      AURTX.py
      AURTX.tex
    BBCDD
      BBCDD.json
      BBCDD.py
      BBCDD.tex
    BBHOZ
      BBHOZ.json
      BBHOZ.py
      BBHOZ.tex
    JNJMQ
      JNJMQ.json
      JNJMQ.py
      JNJMQ.tex
  sections
    Electric Circuits
      q_map.csv
    Power Systems
      q_map.csv
```

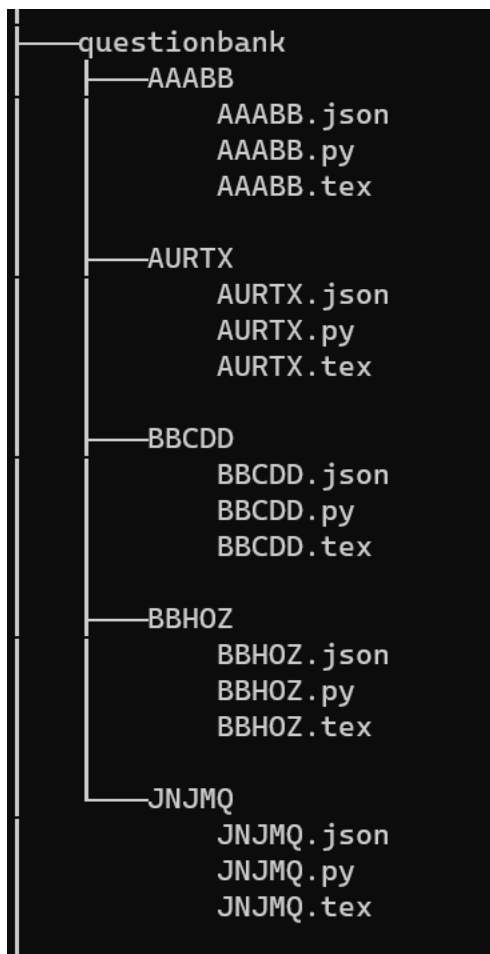
### Qrcodes Folder:

Contains images of QR-codes



### Question Bank Folder:

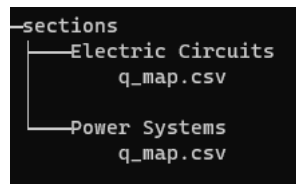
Contains a list a folders . Each folder has the name of a 5digit unique question code. Each folder is responsible for the generation of a question with that question code.



### Sections Folder:

The sections folder contains list of topics. Here “Electric Circuits” and “Power Systems”.

Each topic folder contains a “q\_map.csv” file which contains list of question codes.



```
C:\Users\Ravivarma\Downloads\Question_paper_generation_software\sections\Electric Circuits>type q_map.csv
AAABB
BBCDD
BBCDD
```

```
C:\Users\Ravivarma\Downloads\Question_paper_generation_software\sections\Power Systems>type q_map.csv
BBHOZ
AURTX
JNJMQ
JNJMQ
JNJMQ
BBHOZ
```

### Illustration of a Question:

Let us take the example of the folder “AAABB”

“AAABB.tex”: contains latex code for the question with variable question choices and values

The i,a,q1cc,q1wc1,q1wc2,q1wc3 are variables.

```
C:\Users\Ravivarma\Downloads\Question_paper_generation_software\questionbank\AAABB>type AAABB.tex
\question
What will be the current density of metal if a current of --i--A is passed through a cross-sectional area of --a--m^2$?

\begin{randomizechoices}
\correctchoice --q1cc-- A/$m^2$
\choice --q1wc1-- A/$m^2$
\choice --q1wc2-- A/$m^2$
\choice --q1wc3-- A/$m^2$
\end{randomizechoices}
```

“AAABB.py”:

the code generates random values for the current and area parameters of a question. It then calculates different options based on those values and stores them along with the parameters in a JSON file named "AAABB.json".

10 random values are generated .

```

C:\Users\Ravivarma\Downloads\Question_paper_generation_software\questionbank\AAABB>type AAABB.py
# python file to generate the json options for the question
import random, math, cmath, json

q_json_file = open("AAABB.json","w")      #

def rr(mn, mx, st):
    return random.randrange(mn, mx, st)

def f2s(no,d):
    s1 = "{z:0."+str(d)+"f}"
    return s1.format(z=no)

def get_ckt_values():
    # given parameters for the question
    i= rr(1, 50, 1) # current
    a= (rr( 1, 20 ,1))/20 #area
    return [i,a]

q_json={}
q_json["values"]=[]

for x in range(10):
    [i,a] = get_ckt_values()

    qlcc = f2s(i/a,4)
    qlwc1 = f2s(a/i,4)
    qlwc2 = f2s((i*100)/a,4)
    qlwc3 = f2s(i/(a*100),4)

    q_json["values"].append([i,a,
                             qlcc, qlwc1, qlwc2, qlwc3])

q_json["parameters"]=["i","a",
                      "qlcc","qlwc1","qlwc2","qlwc3"]

json.dump(q_json, q_json_file)
q_json_file.close()

```

“AAABB.json”: file generated by running “AAABB.py”

```

C:\Users\Ravivarma\Downloads\Question_paper_generation_software\questionbank\AAABB>type AAABB.json
{"values": [[19, 0.8, "23.7500", "0.0421", "2375.0000", "0.2375"], [9, 0.75, "12.0000", "0.0833", "1200.0000", "0.1200"], [35, 0.9, "38.8889", "0.0257", "3888.8889", "0.3889"], [8, 0.75, "10.6667", "0.0938", "1066.6667", "0.1067"], [38, 0.2, "190.0000", "0.0053", "19000.0000", "1.9000"], [22, 0.5, "44.0000", "0.0227", "4400.0000", "0.4400"], [28, 0.75, "37.3333", "0.0268", "3733.3333", "0.3733"], [21, 0.6, "35.0000", "0.0286", "3500.0000", "0.3500"], [49, 0.65, "75.3846", "0.0133", "7538.4615", "0.7538"], [1, 0.25, "4.0000", "0.2500", "400.0000", "0.0400"]], "parameters": ["i", "a", "qlcc", "qlwc1", "qlwc2", "qlwc3"]}

```

## Working of “run.py”:

This Python code generates question papers for an exam in PDF format. Here are the main results produced by the code:

### 1.Generation of Question Papers:

- The code generates a random alphanumeric code for each question paper using the `gen_code()` function.
- It creates a LaTeX file for each question paper (`exam_<code>.tex`) based on a template defined in the `start_text` and `end_text` variables.
- The code selects random questions from different sections and includes them in the question papers.
- For each selected question, it replaces the placeholder values in the LaTeX template with actual values obtained from corresponding JSON files.
- The generated question papers are saved in the current directory as PDF files (`exam_<code>.pdf`).

### 2.Generation of Answer Keys:

- The code creates an answer key for each question paper.
- It generates a LaTeX file for each answer key (`exam_key_<code>.tex`) based on a template defined in the `start_text_key` and `end_text_key` variables.
- The answer keys contain the correct answers for each question.
- The generated answer keys are saved in the current directory as PDF files (`exam_key_<code>.pdf`).

### 3.JSON Files:

The code creates several JSON files to store related data:

- `keys.json`: Contains the answer keys for each question paper.
- `questions.json`: Contains the list of selected questions for each question paper.
- `question_allocation.json`: Contains the mapping of questions to question papers.

### 4.File Management:

- The code organizes the generated files by creating a separate directory for each question paper using the generated code.
- It moves the corresponding LaTeX files, JSON files, and PDF files to their respective directories.

Please note that the specific results and file names may vary depending on the input parameters and the structure of the question bank directory.









## Results:

Each time run.py is executed a unique Question paper with a 10 digit unique is generated.









The Answer key for the question paper is also generated.

- In the Question paper generated under the Electric Circuits section there are 2 questions of BBCDD and 1 question of AAABB
- In the Question paper generated under the Powe Systems section there are 2 questions of BBHOZ, 3 questions of JNJMQ and 1 question of AURTX

Files generated by executing run.py one time:

 exam\_TJKZMLAZFS.pdf  
 exam\_key\_TJKZMLAZFS.pdf  
 exam\_key\_TJKZMLAZFS.tex  
 exam\_TJKZMLAZFS.tex  
 key\_TJKZMLAZFS.json  
 keys.json  
 question\_allocation.json  
 questions.json

Files generated by executing run.py another time:

 exam\_ZTQUQZQEYY.pdf  
 exam\_key\_ZTQUQZQEYY.pdf  
 exam\_key\_ZTQUQZQEYY.tex  
 exam\_ZTQUQZQEYY.tex  
 key\_ZTQUQZQEYY.json  
 keys.json  
 question\_allocation.json  
 questions.json

Each time different files are generated by executing “run.py”

**Key Features:**

1. **Randomized Question Selection:** The project randomly selects questions from a question bank, ensuring each exam paper is unique.
2. **Shuffling of Questions:** The project shuffles the order of questions within each section, increasing the randomness and fairness of the generated exam papers.
3. **Parameterized Questions:** The project supports parameterized questions, allowing different numerical values or parameters to be assigned to each exam paper. This enables personalized assessments and variability in the generated exams.
4. **Generation of Key:** The project generates an answer key for each exam paper, providing a reference for evaluating the students' answers.
5. **File Management:** The project organizes the generated exam papers, answer keys, and associated files into separate directories for easy access and distribution.

**Specialty:** This project specializes in automating the process of creating exam papers. By leveraging randomization and parameterization, it ensures that each student receives a unique set of questions, maintaining the integrity and fairness of the assessment process. The ability to shuffle questions and assign different parameter values adds an element of surprise and customization to the exams.

**Impact:**

1. **Time-saving:** The project automates the manual process of creating exam papers, saving significant time for instructors or exam administrators.
2. **Customization:** The parameterized questions feature allows for personalized assessments, catering to individual student needs and providing a tailored learning experience.
3. **Fairness and Integrity:** The randomization and shuffling of questions ensure fairness in the exam papers by minimizing the possibility of cheating or sharing answers among students.
4. **Resource Utilization:** By utilizing a question bank, The project optimizes the use of available resources, making it easier to create and manage a large number of exams.
5. **Standardization:** The generated exam papers follow a consistent format, layout, and structure, promoting standardization and clarity in the assessment process.

Overall, This project streamlines the exam paper generation process, enhances fairness, and provides flexibility and customization in assessments, thereby positively impacting the efficiency and effectiveness of the examination system.

**Future Extension:**

- Integrating with app
- Integrating with web
- Generating more questions on different topics