



**NANYANG
TECHNOLOGICAL
UNIVERSITY**

A Secure Online Quiz System

Submitted by: Bharti Rahul

Matriculation Number: U1322950K

Supervisor: A/P Chua Hock Chuan

School of Electrical & Electronic Engineering

A final year project report presented to the Nanyang Technological University
in partial fulfilment of the requirements of the degree of
Bachelor of Engineering

2016-17

Table of Contents

Abstract.....	iv
Acknowledgements.....	v
List of Figures.....	vi
List of Tables	vii
Chapter 1 Introduction.....	
1.1 Background.....	1
1.2 Motivations	2
1.3 Objectives and Scope.....	3
1.4 Accomplishments.....	4
1.5 Organisation of Report.....	4
Chapter 2 Literature Review... ..	6
2.1 Previous Quiz Assessment System	6
2.2 Flask.....	8
2.3 Cheating methods: Beyond Plagiarism.....	12
2.4 Tackling issue of Cheating.....	13
Chapter 3 System Overview.. ..	14
3.1 Web Application Architecture.....	14
3.2. System Requirements	15
3.3 Key Stakeholders and Use Cases.....	18
3.4 Database.....	20

Chapter 4 Implementation.....	26
4.1 Quiz Preparation	26
4.2 Quiz Delivery.....	28
4.3 Security.....	29
Chapter 5 Results and Discussion.....	30
Chapter 6 Conclusion and Future Recommendations.....	31
4.1 Conclusions.....	31
4.2 Future Recommendation.....	32
References	33

Abstract

Online Quiz assessment is becoming widely popular as an academic tool, mainly because of the didactic nature and inexpensive means to test students. In order to keep the online assessments fair, any sort of malpractice employed by students needs to be tackled. Although this can be solved using complex plagiarism detection algorithms, the assessment tools majorly rely on dynamic creation of unique assessment for each student, which is as good as the former method.

The aim of the Final Year Project is to design and develop a web-based and secure Online Quiz System (OQS) using one of the python's latest frameworks, flask that targets the following functions:

1. Ability to test students based on four different types of questions, i.e. Multiple Choice, Multiple Choice Multiple Response, Fill-in-the-blank and Short Answer that support parameterized images in the question description
2. Single Page View, minimalistic, and professional looking User Interface that reduces any scope of errors and minimizes the time to learn the system by a new user.
3. Prevention of cheating by making use of a secure browser and employing more secure technologies to host the Quiz for students.

At the end of the project, a prototype was developed that meets all the necessary design and functional requirements. The web application is built on Flask Stack (Linux, Apache, PostgreSQL, Flask(Python)) as part of Client-Server Architecture. The process of development went through all the steps of SDLC (Software Development Life Cycle) that mainly consists of requirement analysis, designing database of 10 tables, UI design, implementing back-end and front-end functions and black-box testing of the Web Application. Results shows OQS's capacity as a reliable and effective assessment tool.

Acknowledgements

It's been a great learning opportunity to work on the Final Year Project, and I would like to thank the School of Electrical and Electronic Engineering of Nanyang Technological University, for giving me this opportunity.

I would like to extend sincere gratitude to my FYP Supervisor, Assoc. Prof Chua Hock Chuan for the mentorship and counselling during the project. The valuable suggestions, the resources and most importantly, the freedom he has given helped me to push myself and develop a strong base in database and web development.

The list of acknowledgements cannot be completed without acknowledging the extensive support provided by my friends and family who have taught me a lot of valuable lessons and been a constant support system throughout my life.

Rahul Bharti

March 2017

List of Figures

Figure 3.1: Web Application Architecture.....	14
Figure 3.2 Example of button element.....	16
Figure 3.3 Use Case Diagram.	19
Figure 3.4 Entity-Relationship Diagram.....	20
Figure 4.1: Work Flow of adding a question.	26
Figure 4.2: Screenshot showing Right-click to add a parameter.	27
Figure 4.3: Screenshot showing Add Answer Choices Table.	27
Figure 4.4: Screenshot of Quiz Delivery to student	28

List of Tables

Table 2-1 Code Snippet for Standalone Flask Web App.	9
Table 2-2 Code Snippet for a sample database class.	10
Table 2-3 Code Snippet for Constructing a sample form.	11
Table 3-1: Table Design: QnA.....	21
Table 3-2: Table Design: MCQMR.....	21
Table 3-3: ques field of MCQMR.....	21
Table 3-4: ans field of MCQMR.....	22
Table 3-5: Table Design: FIB.	22
Table 3-6: ques field of FIB.....	22
Table 3-7: ans field of FIB.....	22
Table 3-8: Table Design: SA.....	23
Table 3-9: ques field of SA.....	23
Table 3-10: ans field of SA.....	23
Table 3-11: Table Design: Courses.	23
Table 3-12: Table Design: CourseGroupUsers.	24
Table 3-13: Table Design: Users.....	24
Table 3-14: Table Design: Submission.....	24
Table 3-15: Table Design: Assessments.....	25
Table 3-16: Table Design: SpecificAssessment.....	26

Chapter 1 Introduction

1.1 Background

One of the most effective ways of providing students with developmental and continuous feedback on their learning growth is the use of formative quizzes. The quizzes help not only the students to understand and test their module content knowledge but also, help the instructors and students in understanding the gaps in students' comprehension. Furthermore, instructors can apply the information attained from students' score trends and pay more attention to specific parts of module where most of the students are lacking.

Keeping up with current technology advancements and trends, the only feasible way to have continuous formative quizzes is to keep the tests online. This way, the test scores can be immediately calculated and thus, the student can get instant feedback. Also, it helps to reduce literally tons of paper wastage, and provide objectivity in test assessment.

Some of the threats to fair test-taking by students, is the scope of copying other students' answers and hacking the system. In order to remove any scope for malpractice, secure browsers will be used so that students will not be able to open any other tabs, or use modifier keys (e.g. Ctrl or Alt). Also, the answer choices are randomized (e.g. in MCQ type questions) and questions will have different variations on the basis of parameters used.

1.2 Motivations

The project is based on the previous work done by the FYP Supervisor, Assoc. Prof. Chua Hock Chuan. The previous online quiz assessment web-app is based on PHP for Back-end processing and MySQL for database needs. Since, PHP doesn't have an extensive library support, there is a lot of boiler plate code to write and thus, the efficiency decreases. On the other hand, using plain SQL queries is also not a good practice since, database can change with time, thus, rendering the code maintenance expensive and tedious.

Thus, there is a need for developing an online quiz assessment tool based on newer technologies and using libraries instead of typing boilerplate code. This could be achieved by using flask microframework of python. Although flask doesn't have any data abstraction layer of its own, Flask supports extensions that can add application features as if they were implemented in Flask itself. Using SQLAlchemy (one of such extensions), which is a Python SQL toolkit and an Object Relational Mapper (ORM) gives a lot of flexibility to work with SQL Databases. It can work with MySQL, PostgreSQL and a variety of other SQL based databases. Therefore, extensive usage of libraries helps with keeping the Lines of Code (LOC) as low as possible while minimizing the possibility of errors. Also, future addition of features is made simpler because of the same.

Currently, the focus is to reduce the boilerplate code and use various security methods to provide a stable web-app that can cater to Instructors and students alike. For instructors, the focus is to provide streamlined question adding procedure, and keep a record of students' submissions and grades. For students, the focus is to keep the assessment-taking as simple as possible while keeping the scope of any malpractice during assessment-taking as low as possible.

1.3 Objectives and Scope

This section properly describes the challenges and problems that the FYP plans to address and furthermore, describe the scope of the project.

1.3.1 Objectives

The aim of the project is to develop a secure online quiz assessment system using Flask stack (Linux, Flask, PostgreSQL, Python), JavaScript, Bootstrap, HTML5, and CSS3. The objective is to keep the UI as streamlined as possible and inculcate Single Page View for question addition. This will help in not only minimising the time taken to create an assessment but also minimise the time it takes for new users to get used to the system. Furthermore, security is one of the topmost priority, which can be kept in check by utilising Flask's common security mechanisms, such as Role management, Password encryption, Basic HTTP authentication etc.

1.3.2 Scope

The scope of the FYP consists of, but is not limited to,

- Smoothly add parameters to all types of questions ranging from Short Answer, Fill-in-the-blank and Multiple Choice Questions.
- Inculcate Single Page View Design in the Web App wherever possible to minimise need for short memory retention for the user.
- Improve Security mechanisms in order to reduce any possibility of malpractice during quiz-taking.

1.4 Accomplishment

By the end of the project, objectives and scope were successfully achieved. The Online Quiz System met all the given design and functional requirements. The quiz system is able to evaluate students on four types of questions, i.e. Multiple Choice Questions, Multiple Choice Multiple Response Questions, Fill-in-the-blanks and Short Answer Questions.

With the help of HTML, JS and Bootstrap CSS, the designed interface did prove to be helpful in adding the questions in a quicker and more efficient way.

1.5 Organisation of Report

The Project report will formally discuss the overall development of Quiz Assessment system, starting from Database schema design to Usability features incorporated to ease-in a new user and reduce scope of errors.

Chapter 1 provides an introduction to the Quiz assessment system including the background knowledge, motivation behind the project, the key objectives and the scope of project, and lastly, accomplishments attained during the course of project.

Chapter 2 comprises of review of past work done in the same field. Additionally, it touches up on the most important technologies being used in designing the web application ranging from flask, the microframework used to host the website and the SQLAlchemy, the Object relational mapper of flask to deal with database. Furthermore, it highlights the common cheating methods that are employed by students in order to trick the 'system' and the new security measures that are being used to make the system less vulnerable to such practices.

Chapter 3 gives the system overview of the Web App infrastructure with the help of schematic diagrams.

Chapter 4 provides the implementation specific information of all the key elements of the web application and gives more information on security measures taken to make the

assessment system more resilient to security vulnerabilities.

Chapter 5 comprises of the results and discusses the performance of the Web Application.

Chapter 6 comprises of conclusions drawn and further recommendations to improve the usability of Quiz System.

Chapter 2 Literature Review

2.1 Previous Quiz Assessment System

The online quiz system previously developed was part of a Final Year Project as well. It was based on PHP for back-end processing and utilized MySQL for database needs. It focused primarily on easy interface design and expediting the process of adding parameters in questions.

It focused primarily on the following aspects of Interface Design [1]:

- i. User Interface Design's Importance
- ii. E-Learning Systems User Interface
- iii. Types of Questions for the quiz assessment system
- iv. Framework for Frontend Development

User Interface Design's Importance:

Specifically, it explained how important it is for quiz system to be usable for different user groups, divided specially based on tech-savviness and the need for UI to be consistent, intuitive, and simple. [2, 3]. Students need to easily comprehend the mechanism to answer questions and instructors need to be able to key in the questions easily.

E-Learning Systems User Interface:

Furthermore, since users focus more about the tasks they need to perform than the user interface, so clear instructions, in concise form need to be provided for every option available on the screen in order to remove any point of confusion. Various aspects such as Typography, Consistency and use of colors can help to convey the information in a clear way.

Types of Questions for the quiz assessment system

Additionally, the project has inculcated four different types of questions that have been selected out of all the possible types. [4] [5] [6] [7] [8]

The type of questions are:

2.3.1 MULTIPLE CHOICE QUESTION (MCQ)

These questions have only one possible correct answer out of all the possible choices. The choices are shown beside the radio buttons and marks are given only if student chooses the correct answer.

2.3.2 MULTIPLE CHOICE MULTIPLE RESPONSE (MCMR)

These are extremely similar to MCQ, except they might have more than one correct answer. Thus, checkboxes are used instead of radio buttons to show multiple option selection capability.

2.3.3 FILL-IN-THE-BLANK (FIB)

Fill-in-the-Blank type of questions, requires student to type in the correct answer. Since, the answer might have a different spelling or rounded off differently, instructor chooses a tolerance level.

2.3.4 SHORT ANSWERS (SA)

Since Short Answers can be multiword, text area is given instead of textbox. Just like FIB, instructor can set up tolerance level.

Framework for Frontend Development

Frontend of a system is what users see and feel. There are a bunch of elements that make up the front end of a webpage. They are stated as below:

1. HTML (Hypertext Markup Language)

HTML is the universal standard markup language which is used to create interface for web pages.

2. CSS (Cascading Style Sheet)

CSS is a style sheet language which is used to describe the appearance of a document written in a HTML.

3. JavaScript

JavaScript is a client-side scripting language which is primarily used to make the UI elements more interactive.

There are several types of frameworks available that make use of the above stated technologies ranging from the most popular one, called Bootstrap to not so popular one, called Gumby Framework.

For the quiz assessment system, Bootstrap was selected primarily due to its popularity and simple nature which in result, has a huge community support. Furthermore, Bootstrap relies on LESS CSS and is compiled via Node.

2.2 Flask

To develop the web application, the author had a choice of selecting from wide variety of frameworks, ranging from non-full-stack frameworks like Flask, Pyramid to full-stack frameworks like Django and web2py.

For the given project, Flask was chosen, primarily because of comparatively simple nature of the small microframework when compared to Django. Also, all the necessary functions could be performed using extensions, such as SQLAlchemy, Uploads etc.

Flask is a micro web framework written in Python. It is based on the Werkzeug toolkit and Jinja2 template engine.

1. Werkzeug: Werkzeug is a Python Web Server Gateway Interface (WSGI)* utility library. It comprises of a Uniform Resource Locator (URL) routing system and full-fledged request and response objects.

*WSGI is an interface between web applications and web servers for Python.

2. Jinja2: a template engine for Python. With its help, Python like expressions can be used in HTML file to generate any markup or source code.

One of the main security features of Jinja template engine is that it helps to prevent Cross-Site Scripting (XSS) by using HTML escaping.

Cross-Site Scripting: It is a kind of computer security vulnerability that has found to be responsible for 84% of website attacks. Through XSS, attackers can inject client-side scripts into other people's web-page views. [14]

Overall standalone Flask provides the following services [9]:

- an interface in compliance with WSGI
- Secure cookies and Sessions.
- a built-in Development Web Server and Debugger.
- Jinja2 templates
- Request dispatching and URL routing.
- Additional security to attacks from use of above mentioned services.

Following is a code snippet which is a standalone Web Application by itself. It can be run by typing, \$python hello.py in terminal. You can check the webpage by typing, http://127.0.0.1:5000/ in address bar of browser window.

Code Snippet : A standalone Flask Web Application to say Hello, World!

```
# -*- coding: UTF-8 -*-
```

```
from flask import Flask
app = Flask(__name__)    # Construct an instance of Flask class for the WebApp

@app.route('/')    # URL '/' to be handled by main() route handler (or view function)
def main():    """Returns hello, world!"""    return 'Hello, world!'

if __name__ == '__main__':    # Condition to check if script is running directly
    app.run()    # Runs the WebApp and hosts it on built-in web server
```

Table 2-1 Code Snippet for Standalone Flask Web App [9]

Flask has various extensions that have been utilized in the project. Some of them are:

2.2.1 SQLAlchemy

SQLAlchemy is the Python SQL toolkit and Object Relational Mapper (ORM) that gives application developers the full power and flexibility of SQL.

SQLAlchemy's Object Relational Mapping capability is one of its key features. In ORM, object classes can be mapped to database tables. Thus, data can be represented and dealt with, in terms of objects. This way, one need not think about data in terms of tables making programming a lot more homogenous.

Furthermore, SQLAlchemy is different from other ORM models, in the way that it doesn't hide the SQL and object relational details behind automated walls, but the processes are completely bare so that developer in charge of how database is organized and how SQL is constructed.

Also, since the programmer doesn't interact with database directly, one of the key advantages of using the SQLAlchemy wrapper is that, further down the line if needed, database can be easily replaced with another one, without much change in code. Thus, it gives great amount of flexibility and power to the programmer.

Code Snippet : A sample class to construct the table QnA in the database

```
# These class variables define the column properties,
# while the instance variables (of the same name) hold the record values
.
class QnA(db.Model):
    __tablename__ = 'QnA'

    questionNo = db.Column(db.Integer, primary_key=True, autoincrement=True)
    questionGroup = db.Column(db.String(64))
    description = db.Column(db.TEXT)
    remarks = db.Column(db.String(2048))
    ques = db.Column(postgresql.ARRAY(db.String(64), dimensions = 2))
    ans = db.Column(postgresql.ARRAY(db.String(64), dimensions=2), default=0)
```

Table 2-2 Code Snippet for a sample database class

The above code snippet if executed would define the class QnA and make a table QnA in the respective database. In case, __tablename__ is not used, the table name is the

lowercase value of the classname.

One can initialize via `db = SQLAlchemy()`, and then access all the available properties via `db` in the form of dot (.) notation.

2.2.1 Flask-WTF

Forms are an important aspect of an HTML page as they help the system to get data from user in real-time. Server side script can extract the data entered by user using either GET or POST method, as appropriate.

Since, Server side script has to recreate the form elements from http request data, thus, essentially form elements are being defined twice, once in HTML and once more in server side script. [10]

That is when WTForms library, can be used to make flexible forms. Form fields can be defined in the python script itself, and rendered using HTML template. Also, form validation can also be applied to the WTF field. [9]

Code Snippet : Constructing a sample form using flask-wtf

```
# Derive a subclass of 'Form' called 'LoginForm', containing Fieldsclass LoginForm(Form):
    username=StringField('User Name:', validators=[InputRequired(), Length(min=3,
max=30)])
    passwd = PasswordField('Password:', validators=[Length(min=4, max=16)])

# Construct an instance of 'LoginForm' called 'form'
form = LoginForm()

# Reference a field via attribute-style
form.username
```

Table 2-3 Code Snippet for Constructing a sample form [9]

2.3 Cheating methods: Beyond Plagiarism

As it's a common belief by students to score the highest marks, some students don't hesitate to use illegal methods. Beyond plagiarism there are a variety of ways used by students to trick the system. Most popular ones are listed as below: [11]

Retaking the assessment:

Most of the times, students are only allowed to take the quiz once in one sitting. There's a clock running with limited time on it. However, sometimes students might interfere with the internet connection and break the connection to server. This way, the student can claim that they had lost power, and thus, need to retake the quiz. Also, by interfering with system clock, the student can get more time than allowed, thus creating element of unfairness in assessment taking.

Getting quiz solutions beforehand:

Since it's difficult to make sure that all students take the quiz at same time, students who took the quiz before may provide solutions to the later students, thus, cheating the system. If the method of using question 'bank' is employed, to generate a unique set of questions for each student, it's difficult to grade the student on fair basis, as it's not possible to guarantee same level of questions for each student.

Identity fraud during the assessment

Sometimes, intelligent students can take the quiz for not-as-proficient ones in order to get them better grades. That's why, it's necessary to ensure the student taking the assessment is one who is actually enrolled for the class. [12] This vulnerability can be tackled by using biometric authentication systems as mentioned in the next section.

2.4 Tackling issue of cheating

Since online quiz assessment is more prone to cheating by students, new technologies have been introduced in the world of quiz-taking. Coursera is a leading company that offers MOOC (Massive Open Online Courses). Coursera uses biometric methods to authenticate student's identity. Two of the methods used during most of its assessments are:

Face Recognition:

When the student is enrolled in Coursera's Signature Track, the student needs to provide a headshot and a picture of Identity Card. This way, before every quiz student's identity can be verified against the biometric profile in the database.

Typing Rhythm Test:

Research shows that every person has a unique typing rhythm. This result can be used to authenticate student's identity by letting the student type a short phrase right before the quiz.

The above-mentioned methods certainly provide some level of effectiveness but still are not full-proof, since a student can authenticate themselves right before the quiz and let someone else answer the questions for them. Thus, the best method is to make the students take the quiz in a slightly controlled environment. Furthermore, by randomizing the order of answer choices and introducing variations through use of parameters, the issue of cheating can be tackled.

Chapter 3 System Overview

3.1 Web Application Architecture

The architectural layout of Web App can be seen below:

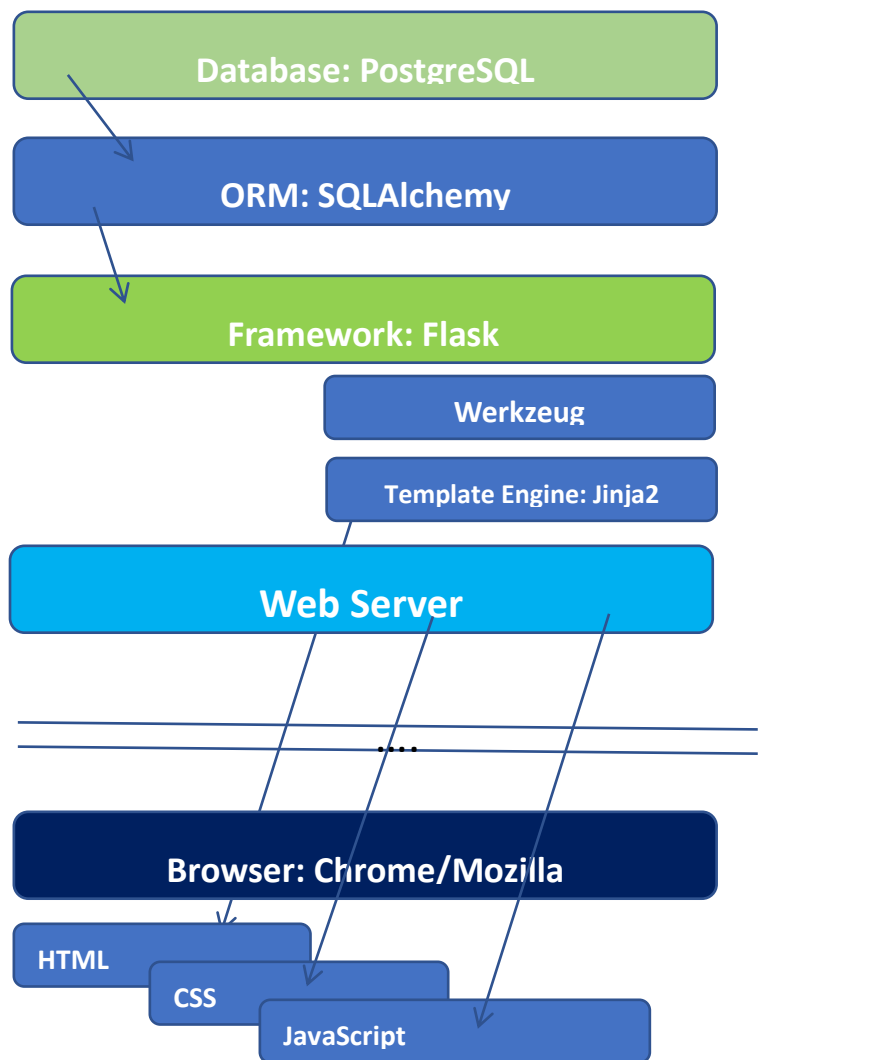


Figure 3.1: Web Application Architecture

In brief, the PostgreSQL is the database used in the Web Application that sits on Server machine. SQLAlchemy acts as a wrapper, that co-ordinates the exchange of data from Flask web Microframework to Database and vice versa. The flask, with the help of template engine and Werkzeug, a universal WSGI interface renders the data received from Database in the viewable format. Since, the flask framework itself cannot deliver the data directly to user on the client side, Web Server helps to send the data to client. The client can use either of the Web Browsers, Mozilla Firefox or Chrome to view the rendered web page. [13]

Note: During development phase, flask's in-built web server can be used, however, once the app goes in production environment, a dedicated web server is required. Apache, or Cherokee are two such common web servers.

3.2. System Requirements

Requirements of system can be grouped in three ways, listed as below:

3.2.1 Design Requirements

Since one of the key objectives of Web App is to make the addition of questions easier to the question bank, special emphasis has been given to the UI's elements so that new users make minimum or no errors. Following requirements need to be catered to do the same: [1]

1. The user interface needs to have minimalistic design that looks professional as well as be powerful enough to cater to all the needs of instructors when it comes to adding questions.
2. The parameters need to have the ability to not only have text but also allow images to be inserted.
3. Navigational and functional buttons need to have locational consistency in the UI.

Following are some key aspects of UI that have been taken in mind, while designing the front-end of system:

1. **Color palette and theme**

Since the design requirements needs the UI to be professional looking, the theme has been kept minimalistic by choosing the white and grey as primary colors. Furthermore, instructors might need to use system for long period of time, thus, the neutral colors such as various levels of blue have been used for functional buttons, to not put strain on eyes while using the system for longer periods.[1]

Below is the screen shot of the button:



Figure 3.2 Example of button element

2. **Typography**

Typography refers to the font type used in a piece of document. Since, bootstrap framework provides a standard family of fonts "Helvetica Neue", "Helvetica", "Arial", and "sans-serif" in its font stack, the same have been used. These fonts are referred to as web fonts and are a universal standard. In case, Helvetica isn't available on a computer, the next font stack will be used instead.

3. **Grid System**

Bootstrap follows a grid system that allows the page to be divided in up to 12 columns. Since, Bootstrap's grid system is responsive in nature, i.e. the size of column rearranges according to screen size. Grid systems not only helps the

programmers stack the data in a clean and neat way, but also helps the user to understand the information at a quick glance.

4. Consistency

By keeping the web page elements such as navigational buttons at same place, element of consistency has been used so that, navigation between the pages is intuitive and thus, addition of questions can be quicker and painless.

3.2.2 Functional Requirements

The quiz assessment system has a number of functional requirements as stated below:

1. Enrollment

Instructor needs to be able to enroll students to the specific course code. Once enrolled, all the students part of the course code can access login the system to access the specific assessment. Thus, there will be two access levels, one is Instructor and one is student.

2. Login

A student needs to be able to log in using the NTU credentials. Although the system currently doesn't have access to student credentials' database, but once put in production, the system will be able to verify the credentials and let student log in.

3. Preparation of Quiz

The system should be able to offer four types of questions, namely Multiple Choice Questions, Multiple Choice Multiple Response Questions, Fill-in-the-blanks and finally Short Answer Questions. Also, parameterization should be allowed in questions, which should allow both text and image to be represented.

4. Assessment Delivery

In order to avoid confusion and overload of information at one page, only one question at one page should be allowed, however, students should be able to navigate freely among questions.

Furthermore, there needs to be a timer which once runs out, should submit the responses to the server and stop the assessment.

5. Result

The system needs to be able to automatically score all four types of questions, and deliver the result to students. Furthermore, partial scores should be given for Multiple Response type of questions in case, student is able to guess at least one of many answers.

3.3 Key Stakeholders and Use Cases

In order to keep things simple, two access levels have been allocated to three types of actors, Administrator, Instructor and Student. Administrator and Instructor have the same privilege level. In future, if the audit trail is implemented, Administrators can be given superior privileges and will be able to access instructors' trail of actions.

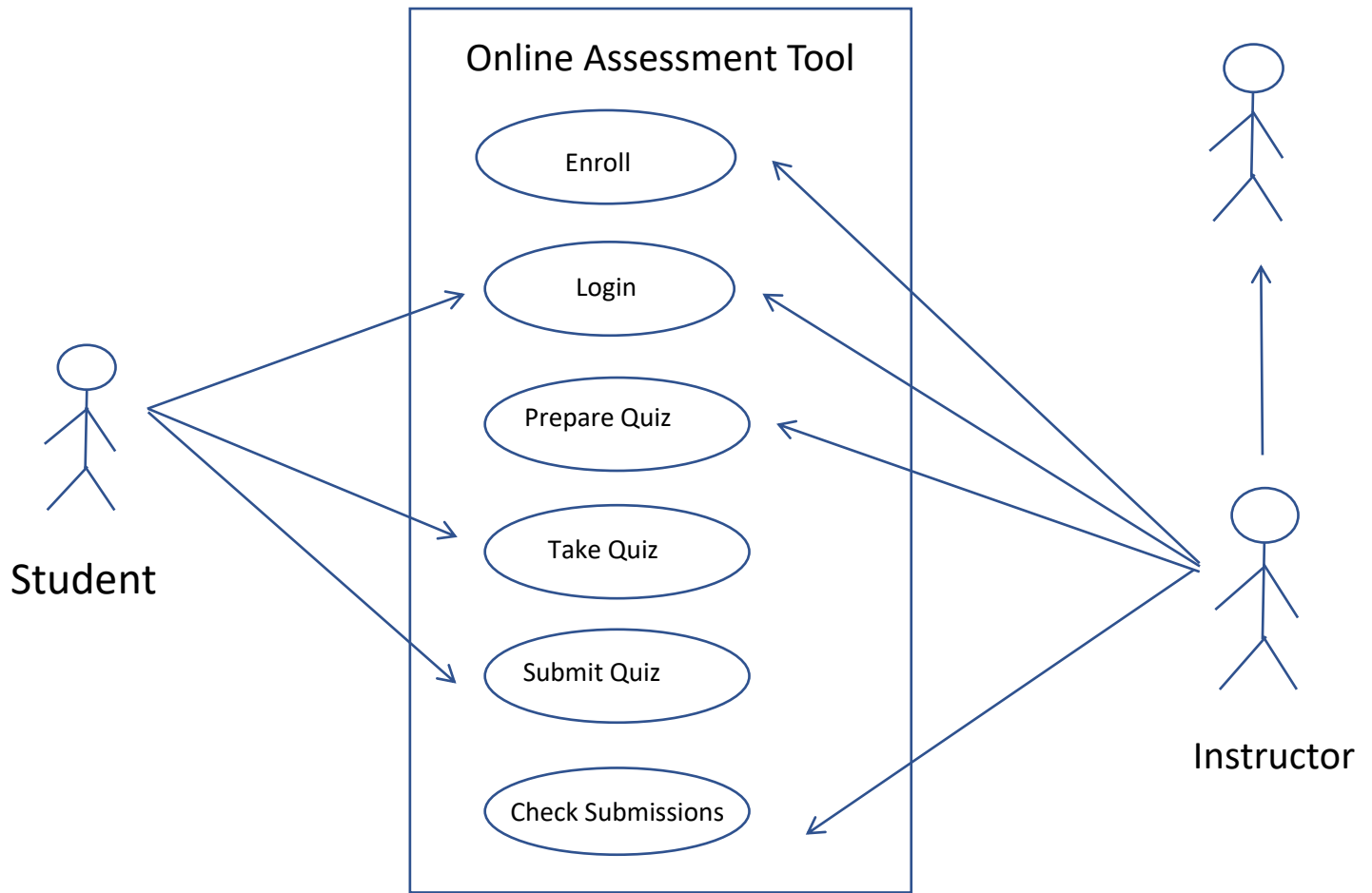


Figure 3.3 Use Case Diagram

3.4 Database

PostgreSQL has been chosen the database for storage needs. Reason for doing so, was to accommodate Array datatype, which is not available in MySQL.

There are in total 10 Tables. The ER Diagram of the same can be referred below:

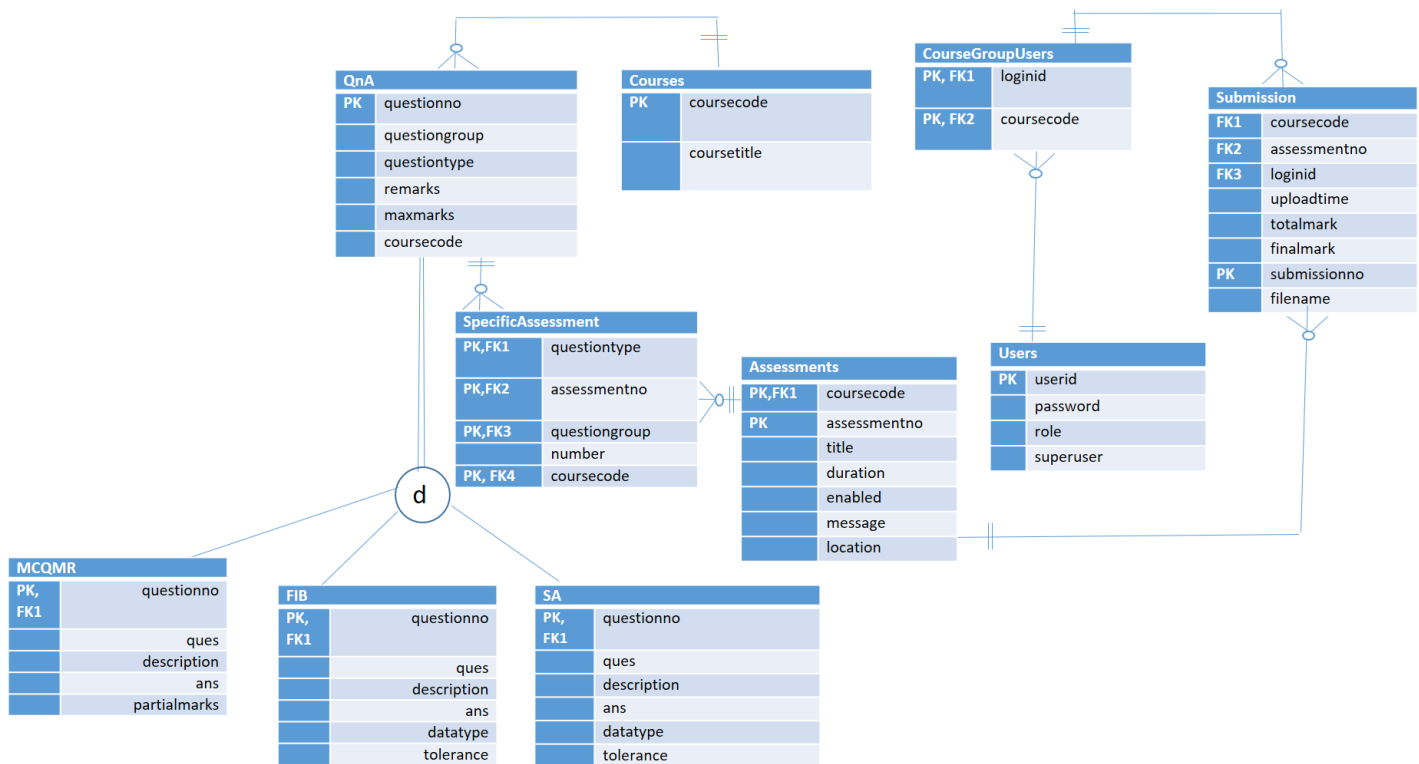


Figure 3.4 Entity-Relationship Diagram

Note: Separate tables have been designed for each type of Question because of different order of values in the 2D array datatypes, ques and ans for each type of question.

Below is the schema for each table:

Field	Explanation
questionno	Primary Key
questiongroup	The group that question belongs to
questiontype	ENUM: Question Type can be MCQ, MCMR, SA, FIB
remarks	Comments about the question for inference
maxmarks	Maximum marks for the question
coursecode	Foreign Key

Table 3-1: Table Design: QnA

Field	Explanation
questionno	Primary and Foreign Key
ques	2D Array explained below
description	Stem of the Question
ans	2D Array explained below
partialmarks	Partial marks for each correct value in Multiple response type questions. Value is 0 if Question Type MCQ

Table 3-2: Table Design: MCQMR

The ques and ans datatypes are 2D arrays with columns as represented below:

Column Name	Explanation
hasparam	1 if question has Parameters, otherwise 0
variationno	Variation Number of question
paramno	Parameter Number of the specific Variation
paramtype	Can be 'text', or 'image'
paramvalue	Value of parameter. Name of image in case parameter is image

Table 3-3: ques field of MCQMR

Column Name	Explanation
variationno	Variation Number of question
choiceno	Choice Number
choicevalue	Value of Choice
isanswer	Value is 1 if the respective choice is correct, otherwise it's 0

Table 3-4: ans field of MCQMR

Field	Explanation
questionno	Primary and Foreign Key
ques	2D Array explained below
description	Stem of the Question
ans	2D Array explained below
datatype	ENUM: Data Type of Answer, can be String, Float or Integer
tolerance	Tolerance of Answer if type is float or Integer

Table 3-5: Table Design: FIB

The ques and ans datatypes are 2D arrays with columns as represented below

Column Name	Explanation
hasparam	1 if question has Parameters, otherwise 0
Variationno	Variation Number of question
paramno	Parameter Number of the specific Variation
Paramtype	Can be 'text', or 'image'
Paramvalue	Value of parameter. Name of image in case parameter is image

Table 3-6: ques field of FIB

Column Name	Explanation
variationno	Variation Number of question
ansno	Answer Number of the specific Variation
answerkey	Value of Answer

Table 3-7: ans field of FIB

Field	Explanation
questionno	Primary and Foreign Key
ques	2D Array explained below
description	Stem of the Question
ans	2D Array explained below
datatype	ENUM: Data Type of Answer, can be String, Float or Integer
tolerance	Tolerance of Answer if type is float or Integer

Table 3-8: Table Design: SA

The ques and ans datatypes are 2D arrays with columns as represented below:

Column Name	Explanation
hasparam	1 if question has Parameters, otherwise 0
variationno	Variation Number of question
paramno	Parameter Number of the specific Variation
paramtype	Can be 'text', or 'image'
paramvalue	Value of parameter. Name of image in case parameter is image

Table 3-9: ques field of SA

Column Name	Explanation
variationno	Variation Number of question
ansno	Answer Number of the specific Variation
answerkey	Value of Answer

Table 3-10: ques field of SA

Field	Explanation
coursecode	Primary Key, Course Code
coursetitle	Title of Course

Table 3-11: Table Design: Courses

Field	Explanation
loginid	Primary and Foreign Key
coursecode	Primary and Foreign Key

Table 3-12: Table Design: CourseGroupUsers

Field	Explanation
userid	Login ID of Users
password	Encrypted password
role	ENUM, role can be student, instructor or admin
superuser	Value is yes, if Role is instructor or Admin

Table 3-13: Table Design: Users

Field	Explanation
coursecode	Foreign Key
assessmentno	Foreign Key
loginid	Foreign Key
uploadtime	Timestamp at which assessment was submitted
totalmark	Total mark that is allotted to the assessment; Calculated by summing up max marks of each question
finalmark	Final marks obtained by student
submissionno	Primary Key
filename	Name of the file submitted by student

Table 3-14: Table Design: Submission

The submitted file has the name: <coursecode>_<assessmentno>_<loginid>_<uploadtime>
This is saved both on local machine and the server, which can be later accessed if necessary.

Field	Explanation
coursecode	Primary and Foreign Key
assessmentno	Primary and Foreign Key
title	Title of assessment
duration	Total Duration in minutes
enabled	Value is 1, if assessment is active
message	Instruction or Message that can if required
location	ENUM: Location of message can be beginning or end

Table 3-15: Table Design: Assessments

Field	Explanation
questiontype	Primary and Foreign Key
assessmentno	Primary and Foreign Key
questiongroup	Primary and Foreign Key
number	Number of Questions to pick
coursecode	Primary and Foreign Key

Table 3-16: Table Design: SpecificAssessment

Chapter 4 Implementation

4.1 Quiz Preparation

One of the key objectives is to make the process of adding questions smooth and pain free. With the help of screenshots and schematic diagrams the process of adding questions is explained below:

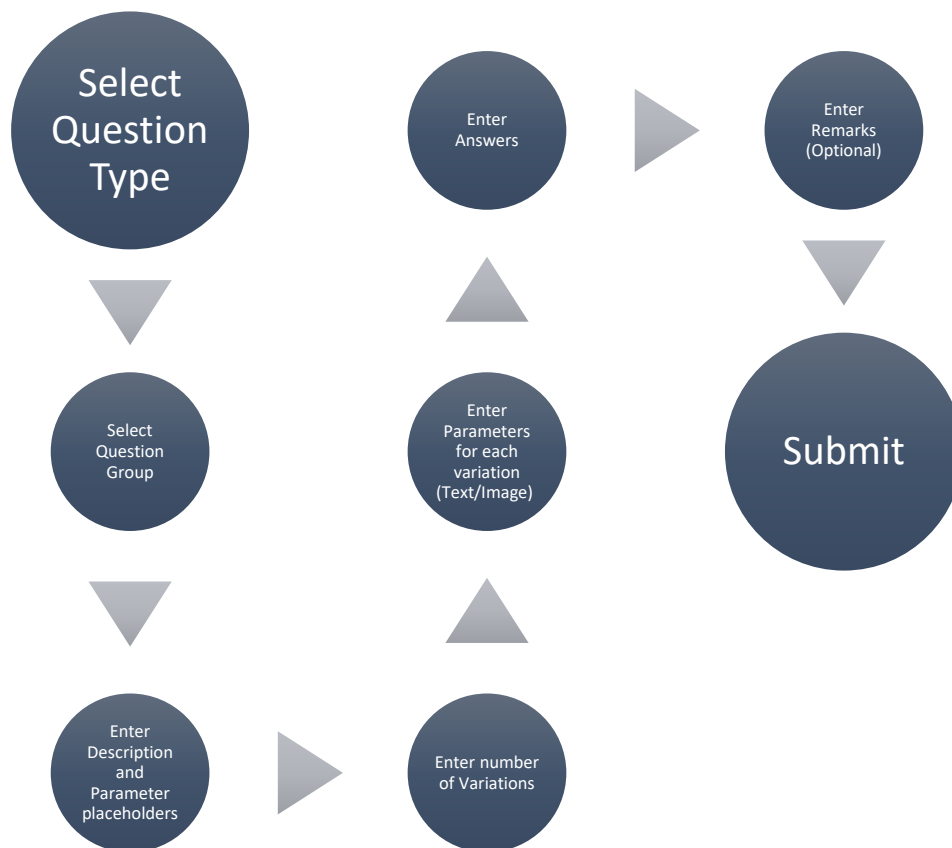


Figure 4.1: Work Flow of adding a question

Numerous times, instructors need to add parameters to a question, this can be done by right-clicking the mouse, and clicking add parameter. This will add %%P%% to the

question description and increase the counter of question parameters by 1. The reason to add such a functionality in right-clicking is to allow less technical-efficient instructors to add parameters without encountering back-end operations of system. For Fill-in-the blanks, right clicking also has Add Answer parameter option.

Add New Question

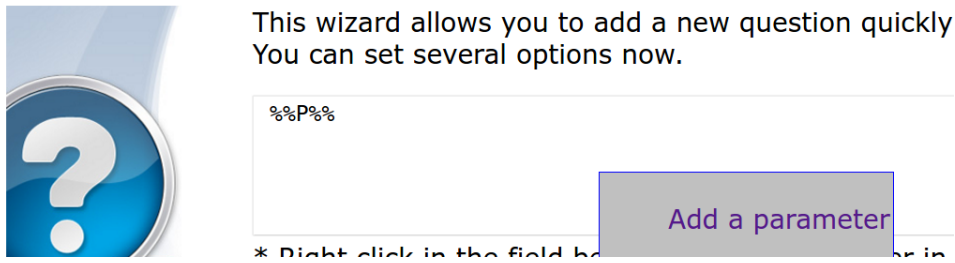



Figure 4.2: Screenshot showing Right-click to add a parameter

For Multiple choice questions, Answers can be added in the table. Since each question can have a different number of choices, 'Add More Choices' button can be clicked to add more number of choices.



Answer Table for Variation 1 :

	Answer	Right Answer		
3	<input type="text"/>	<input type="radio"/>	Delete	Add More Cho
5	<input type="text"/>	<input type="radio"/>	Delete	Add More Cho
3	<input type="text"/>	<input checked="" type="radio"/>	Delete	Add More Cho

Figure 4.3: Screenshot showing Add Answer Choices Table

To add images as parameters, one can simply click Upload Image button in order to upload images. This facility can be used in the instance of having circuit diagram(s) in the question.

For fill-in-the blank type of questions, there are additional fields such as Data Type and Tolerance. Instructor has the option to choose from three data types, 'String', 'Integer' and

'Float'. In case the data type is Float or Integer, Tolerance can be added in Percent Value. For instance, if the tolerance is 5%, then answer of 98 will be marked as correct against the given correct answer of 100 since it's under tolerance limit.

4.2 Quiz Delivery

Once, the student has logged in, the student will be greeted with a optional message. On clicking next minimalistic window telling Total Time Duration of Quiz, Title of Quiz and Start Button will appear. Once the student hits start button, Timer will start, and Question 1 will be presented by default. Student can navigate to other questions by clicking next or clicking question number button in navigation bar.

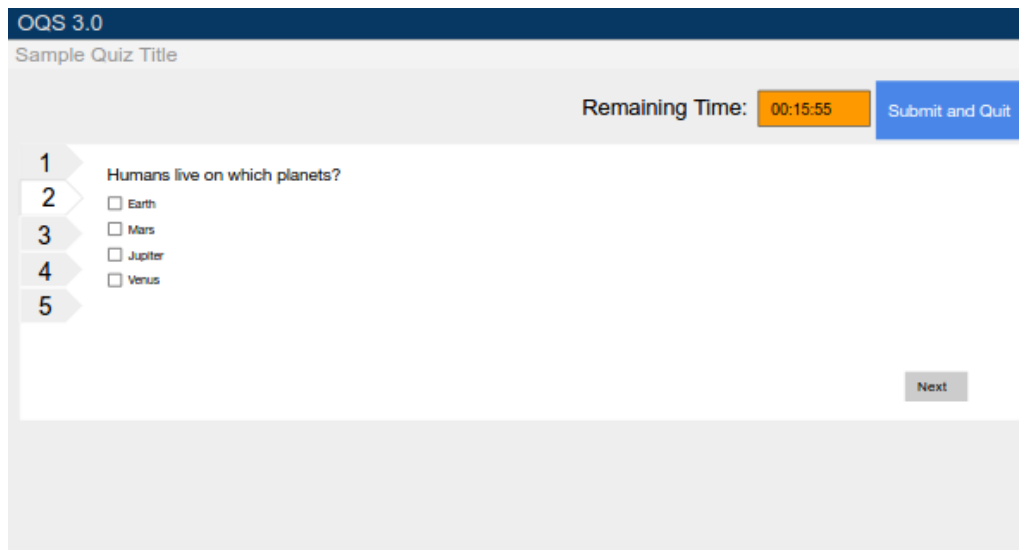


Figure 4.4: Screenshot of Quiz Delivery to student

During navigation, if the student has completed the question, the navigation tab will turn green which implies, student has completed the question. Also, there's a timer running with orange background to inform how much time is left for the quiz to end.

Also, once the timer runs out, or student clicks submit button, the quiz will end and a copy of quiz will be saved both on local machine and server, just in case of network failure.

4.3 Security

In order to keep the quiz system resilient to threats, various techniques have been used.

1. **Secure Browser**

A secure browser has been used that keeps the window on full screen so that address bar and tabs are not visible. Also, the modifier keys (e.g. Ctrl) are disabled.

2. **Flask's Extension Support**

With the help of security measures available in Flask's extensions, such as Session based authentication, Role management, Password encryption, secure cookies etc further protection is provided against Cross-site scripting.

REFER: <https://pythonhosted.org/Flask-Security/>

3. **Form Validation**

Forms have client side validation using JavaScript to make sure user only enters the needed info and doesn't upload any scripts to the server through forms.

Chapter 5 Results and Discussion

The Online Quiz System has been successfully developed. The requirements mentioned in previous chapters (both design and functional) were satisfied, and the bugs have been rectified. The system was tested on Chrome Browser, Version 57.0.2987.98 running on 64-bit Ubuntu Operating System.

The common cheating methods were tried, which failed due to the techniques employed, thus proving system's reliability. Since, the system hasn't been put in production environment and tested with wider range of users on a secure browser, more bugs might be expected. Furthermore, load balancing capability of system also needs to be tested by putting the system under stress testing. In addition, the system needs to be white box tested to further reduce potential risk.

Chapter 6

Conclusion and Future Recommendations

6.1 Conclusion

Due to growing trend of iBT (Internet Based Testing) in today's world, there are variety of assessment tools that are available now. The Online Quiz assessment developed as part of Final Year Project is one such system. With the help of a secure browser, the following objectives have been achieved:

4. Four different types of questions, i.e. Multiple Choice, Multiple Choice Multiple Response, Fill-in-the-blank and Short Answer are supported along with parameterized image addition capability.
5. Simple, minimalistic, and professional looking User Interface that reduces any scope of errors and minimizes time to learn by a new user.
6. Prevention of cheating by making use of a secure browser and employing more secure technologies to host the Quiz for students.

With the help of Flask's libraries and extensions, the Online Quiz System can function as a standalone Web Application that utilizes PostgreSQL hosted on server. The database is well-defined and consists of 10 tables.

The blackbox testing conducted shows that quiz system is free of most of the bugs, is functioning smoothly and prevents any chances of cheating.

6.2 Future Recommendations

Due to limited time, there is still scope of further improvement. Developers in future can work on creating an audit trail of instructors' and students' actions that can only be viewed by Administrator.

Furthermore, grading the quiz can be made more flexible by using Natural Language Processing that not only tests SA and FIB type questions' answers as absolute words but also checks the alternative meaning of the entered answer.

Also, feature of creating questions by uploading a file in required format, will be able to reduce instructor's time. This way, questions will be automatically created and instructor doesn't have to go through all the steps of adding questions.

Since, Formative Assessments are used to test concepts, and give instant feedback, facility of showing comments in case a student get an answer wrong can be provided so that students do not just know that they are incorrect but they also see an explanation of how to improve.

REFERENCES

- [1] E. Emilie Kueh, "A Secure Online Quiz System", pp. 8-11.
- [2] Webster, J. (2015, January 26). Designing an E-Learning Graphical User Interface (GUI). Retrieved March 25, 2017, from [http://www.trainingindustry.com/contentdevelopment/articles/designing-an-e-learning-graphical-user-interface-\(gui\).aspx](http://www.trainingindustry.com/contentdevelopment/articles/designing-an-e-learning-graphical-user-interface-(gui).aspx)
- [3] Legault, N. (n.d.). User Interface Design: 3 Things E-Learning Designers Need to Know. Retrieved March 25, 2017, from <https://community.articulate.com/articles/3-things-every-elearning-designer-needs-to-know-about-user-interface-ui-design>
- [4] Socrative User Guide. (n.d.). Retrieved March 25, 2017, from <http://www.socrative.com/materials/SocrativeUserGuide.pdf>
- [5] Quiz Formatting Guidelines. (n.d.). Retrieved March 25, 2017, from <http://online.ucf.edu/teach-online/develop/assessments/quiz-formatting-guidelines/>
- [6] What Types of Questions Are on a Quiz? (n.d.). Retrieved March 25, 2017, from <https://guides.instructure.com/m/4212/1/50757-what-types-of-questions-are-on-a-quiz>
- [7] Question types. (n.d.). Retrieved March 25, 2017, from https://docs.moodle.org/24/en/Question_types
- [8] Online Quiz Question Types. (n.d.). Retrieved March 25, 2017, from <http://www.wpi.edu/academics/ATC/Collaboratory/HowTo/MyWPI/Bb9/question-types.html>
- [9] H. Chua, "Developing Python Web Applications with Flask", *Ntu.edu.sg*. [Online]. Retrieved March 25, 2017, from http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/Python3_Flask.html.

- [10] "Flask WTF", *www.tutorialspoint.com*, 2017. [Online]. Available: https://www.tutorialspoint.com/flask/flask_wtf.htm. [Accessed: 25- Mar- 2017].
- [11] Neil C. Rowe, "Cheating in Online Student Assessment: Beyond Plagiarism," *Online Journal of Distance Learning Administration*, vol. 7, no. 2, 2004.
- [12] Rob Phillips and Kate Lowe, "Issues associated with the equivalence of traditional and online assessment," in *Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE)*, Adelaide, 2003, pp. 419-431
- [13] H. Karahan, "Flask Guide for Web Application Development", *Webapp-dev.blogspot.sg*. [Online]. Available: <http://webapp-dev.blogspot.sg/2012/07/flask-guide-for-web-application.html>. [Accessed: 25- Mar- 2017].
- [14] "Symantec Internet Security Threat Report", *symantec.com*, 2008. [Online]. Available: http://eval.symantec.com/mktginfo/enterprise/white_papers/b-whitepaper_exec_summary_internet_security_threat_report_xiii_04-2008.en-us.pdf. [Accessed: 29- Mar- 2017].