Smart Exam Administration System

Progress Report for

ENGR 497 Global Design Project I

Ali Emre ÖZ

Fatih Çağatay GÜLMEZ

Muhammed Yasin YILDIRIM

Ali ÇAKMAK, Assistant Professor

College of Engineering

Istanbul Şehir University

December 30, 2017

Abstract

Smart Exam Administration System, also known as SEAS, develops a computer-based examination system for avoiding paper usage, providing fast and reliable evaluation processes, preventing cheating issues and creating student-friendly environments during exams.

Table of Contents

Introduction ……. 4

Background ……. 5

Body of Report ……. 6

References ……. 9

Introduction

SEAS is a 2-side application which consists of a Server-Side and a Client-Side. In Client-Side, there are two different GUIs for both students and instructors. Within the GUI for students, users are able to see questions, type answers and test them whenever they need. At the same time, instructors have the opportunity to monitor the information gathered during the exam. Server-Side is responsible for compiling or testing the answers and analyzing collected data to reveal various problems such as cheating.

The objectives that the project team consider are providing reliable programming-based exams due to writing and testing codes on a paper leads some problems at the end of the day, more comfortable exam environments for students since the exams are supposed to be a test for their knowledge rather than their attention etc, making evaluation processes easier and less time consuming as well as giving a chance to improve both instructors and students themselves by using analyzed statistics provided by SEAS.

For reaching that objectives, the team's motivation is mainly focused on providing a better evaluation of knowledge at first. Thereby, lectures' efficiency might be increased through more accurate statistics. Besides that, a modern approach to examination might lead to improve brand values of educational institutions and decrease their paper usage which makes archiving become much easier than ever before.

Background

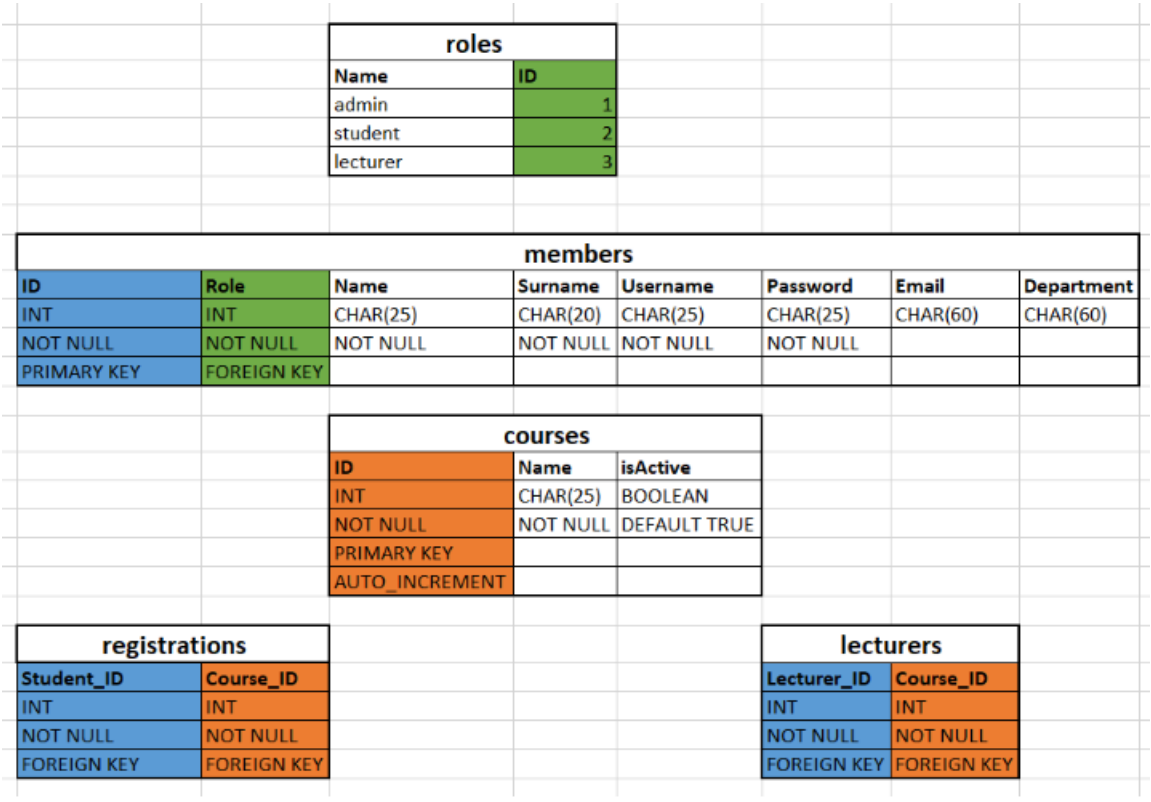
The main idea of the project is the same with one of the previous Global Design Projects that Jareth Conley Moyo, Ibrahim Hamadjoda and Asad Boolani tried to develop. Design of the project is simple as SEAS, a Server-Side and Client-Side app offering various services for both students and instructors. However, this project couldn't get completed and implemented as a working system. Yet it might be a guide for SEAS when machine-learning-related works get in, it contains some algorithms to prevent cheating as planned for SEAS.

Another similar system is Codio which provides a learning experience, that is why it authorizes teachers and students with its technology and learning resources. Codio believes that students and teachers are individuals, teaching and learning styles are different. It offers to enable teachers and faculty to support and master the skills and achievements of students at their own pace by spending 20% more time teaching and 20% less time administering using the provided grading tools. It's integrated with some LMS platforms such as Canvas, Blackboard or Moodle. Besides, Codio claims that it fits to any programming language, any version, any database, any software component to use on its online IDE.

Lastly, there is that Inginious which is also similar to SEAS in some terms. It basically provides a coding ground for testing and evaluating code pieces. However, it is not designed for real-time exams. Thereby, it is not compatible for academic usage. The lack of security and statistics is also an important issue which SEAS overcomes. Inginious, like Codio, is a web-based application and contains Moodle & edX integrations.

Body of Report

Server-Side corresponds to 2 different parts connected to each other, MySQL Database and Flask RESTFul API. The database runs on localhost of the server meaning that database is not connected to the internet directly. The server connects database to its clients in terms of some rules and provided APIs. Database elements are organized as schemas and tables. Each organization has its own schema and this schema is a container for tables of that organization. The tables are members, courses, lecturers, registrations, exams. The members table holds all individuals' information; the columns are respectively ID, Role, Name, Surname, Username, Password, Email, and Department. The courses table holds all information related to courses that uses SEAS; the columns are ID, Name, isActive. The registrations table holds the registration information for each student; the columns are StudentID and CourseID. Lastly, the lecturers table holds the lecturer-course information; the columns are LecturerID and CourseID. The picture below shows the relation and structure of database as explained so far. The shared column colors show foreign key constraints.



Picture 1: Database Structure

In API level, SEAS uses Flask Microservice for Python 2.7. This API provides some key functions to users. API is basically able to handle Organization Sign Up, User Sign Up for a registered organization, User Sign In / Out and so on.

The main tool used is Flask-Security module for a session-based security, defining roles of each user is the main roadmap for the security part of the server.

The most important topic in the security part is that, students can use their computers during exams. So, SEAS must consider what they are able to do on their computer and corresponding enhancements to limit them. Within this regard, more than one method used in the security part and improvements made to prevent cheating at the highest possible level.

As the team, we tried to block the students' access to internet by using very basic networking methods in order to prevent them surf and cheat during exams. However, as the result of our research, we realized that it is even easier to overcome this issue. By keeping the software always on the top while exams go on, SEAS prevents the use of any external program other than itself. Apart from this approach, SEAS is also able to maximize the security by blocking some basic operating system shortcuts (e.g. alt + tab, windows key). In addition to that, we decided to send warning messages first when students try to use these shortcuts or even try to open any external program. Lecturers get informed on such actions taken as well.

SEAS also stores records of students' pressed-key logs for keeping track of what they do during exams. With some enhancements, the software gives accurate feedbacks to lecturers on live-exam-security by using various machine learning approaches analyzing those logs.

Since SEAS aims to run on different operating systems, the libraries that we used for developing is compatible for various systems. Within this regard, GUI is designed to have a dynamic and responsive layout. Although the programming language used in SEAS is Python, Kivy open-source module is the base of GUI.

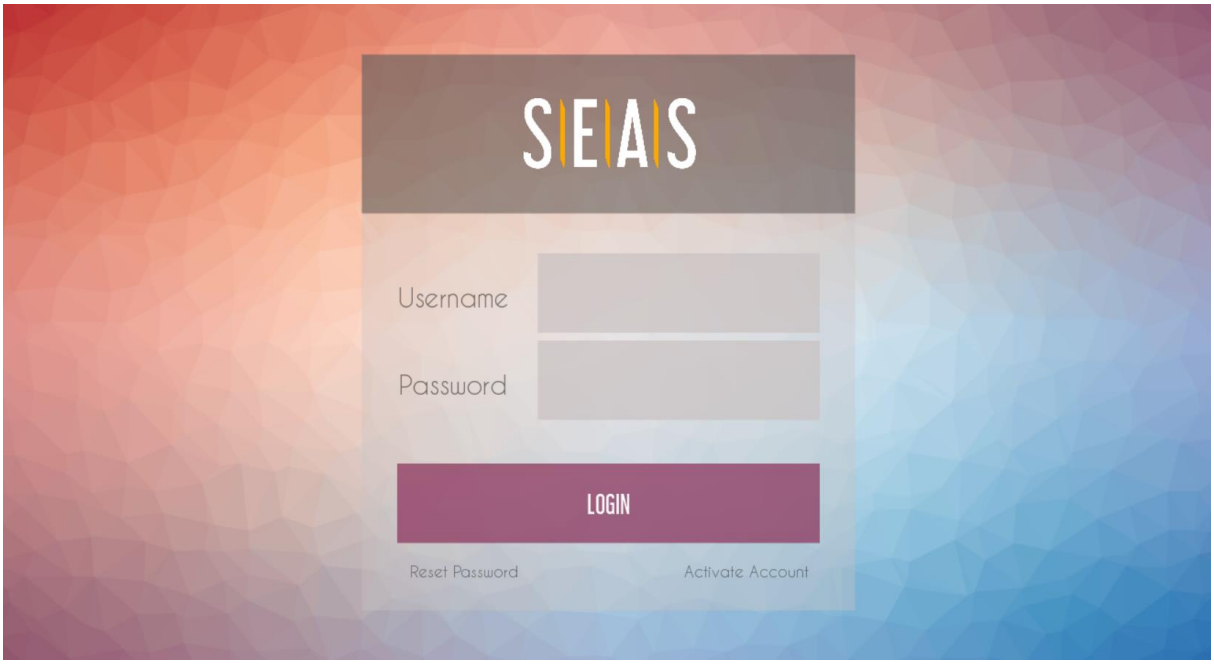
GUI consists of separate screens such as Login Page, Start Page, Lectures Page and so on. Each of these pages are compatible for all operating systems even including mobile ones such as Android, iOS etc.

Design of the pages are done by 2 files, a ".py" file that contains required functions and a ".seas" file that determines where to place what and how to look. Each page is connected to the main file of GUI, which is called "gui.py" and located under the GUI directory.

Relation between GUI and security-provider files is established through threading. When a lecturer starts an exam and students start to join, threading gets active and runs security-provider files to prevent specific unwanted actions as stated above. Besides, GUI also contains some security stuff like warning when ESC pressed or logging if the current screen is left somehow.

Communication between GUI and Server-Side is established through provided APIs. For instance, when a student wants to sign in, GUI takes the username & password inputs and send them to Server-Side via the API. According to the returned value, student gets in or warned. Those kind of returned values get stored in local ".seas" files temporarily created for each action such as sign in, registered courses etc.

Additionally, GUI enables implementation of different libraries, formats or even languages such as Matplotlib, Json and so on. Within this scope, Matplotlib is used for graph-drawing even though Kivy has its own feature called Garden.



Picture 2: Login Page

References

- https://codio.com/docs/
- http://inginious.readthedocs.io/en/v0.4/
- https://kivy.org/docs/api-kivy.html
- https://dev.mysql.com/doc/