Auto Grading

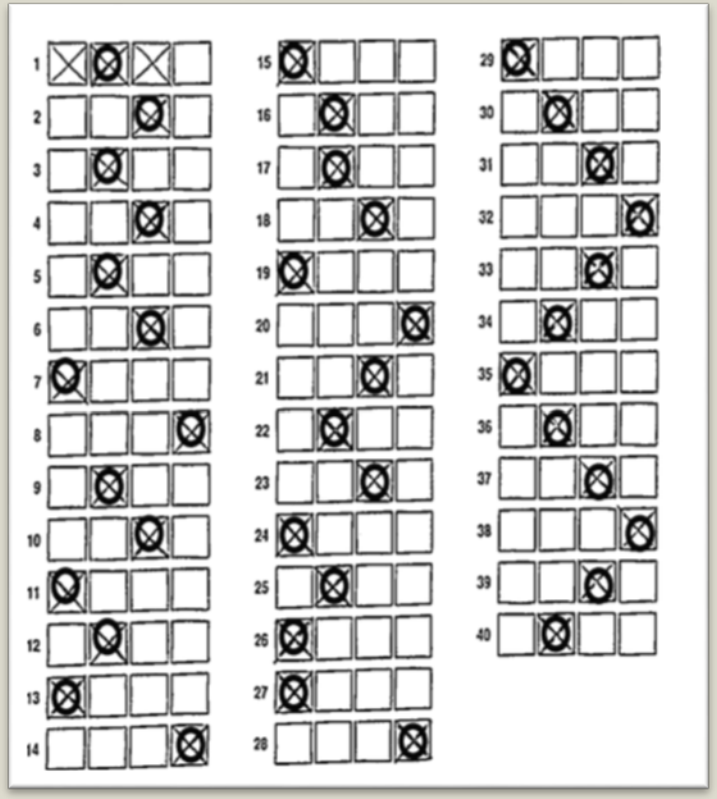
Ruijie Geng

March 9, 2018

This semester:

- Teachers are the main clients (large scale of grading)
- Code reconstruction (divide different functionality into different module)
- Supporting system (automation)
- More functionality (web, database, handwriting recognition, file monitoring)

Overview



File Monitor

- Upload local files
- Check monitored files and call Auto Grading API

Grading

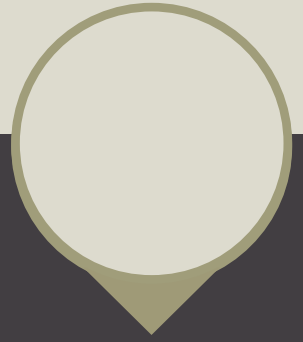
- Grading One Template (IBDP)
- Handwriting recognition

Web

- User login
- Back end
- Front end
- One table database for user login and scores storing

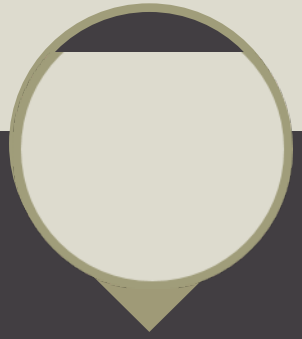
Milestone

Web



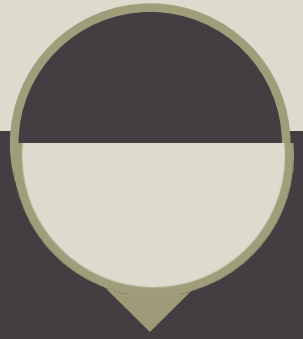
Completed

- Improve existing web page
- Setup database system
- Discuss the frontend



Improvement needed

- Setup user login



In process

- Implement frontend

Not started yet

- Implement user interface for mobile web

Database

Key	Name	Password	Level	Score	Exam name

- Login
- Get score
- Store score

User Login

Achieved Goals

01 The function is able to let users to login to our system if he can offers correct username and password.

02 The registered users may store information on the system.

- As a student, you can check your grades on different tests.
- As a professor, you can store grades and answer sheets of all your students.

Future Goals

03 User registration function with email confirmation.

04 Safety protection such as password hashing and Cross-site Request Forgery

DEMO 1
(Web & Database)

Milestone Grading

- Complete second template
  - (Give up --><--)
- Edge distribution
  (Complete)
- Explore how to use number recognition code to recognize the number in the answer sheet
  (Use handwriting to replace)

March

- Implement function: find the number of questions
  (Complete)
- Implement function: find the layout of the template
  (Complete)
- Combine grading part with number recognition
  (Complete)

CODE RECONSTRUCTION

ORIGINAL

- app.py
- grading.py
- helperfunction.py
- test.py
- static file for front end

NEW

- sample
  - grading
    - grading.py
    - Box.py
    - answerSheet.py
  - web
    - app.py
    - static file
    - database
  - File monitor
    - Server.py
    - Client.py
- test.py

Handwriting recognition

- Use open source library pytesseract to recognize
- Approximate 60% accuracy compare with correct name
- Get handwriting names from specific region in answer sheet, and compare the result with name in database to find most close one.

BUT

- Library just has few functions, so we can not improve accuracy

File Monitor

File Monitor

- Monitoring the changes in the file list
- IO operations based on Python
- Automatically calls the Grading API
- Automatically update/move file

DEMO2
(Grading & Server)

Future

- Improve code
  - Database
  - User login
  - Grading algorithm
- Handle different input image
  - Huge image quality difference between mobile cameras and scanners
- Handle more template
- Find a better library to recognize the handwriting information

THANKS

- Professor Moorthy
- Professor Turner
- Professor Goldschmidt
- The Mentors
- Red Hat
- The RCOS Program
- Python
- OpenCV