CASE STUDY

SUBJECT : ESIOT

STUDENT ATTENDANCE SYSTEM

CLASS : TE-B (2021 BATCH)

GROUP MEMBERS

Atharva Borekar	39012
Atharva Chincholkar	39019
Hardika Doshi	39028
Tejaswini Thakur	39076

TITLE:
A case study of automated online student attendance system

PURPOSE:
The purpose of creating this automated online student attendance system is to aid the teachers and faculty in maintaining attendance records of students. The primary motive of this system is to reduce the time taken by teachers and faculty members to create digital records. Hard copies of attendance need high maintenance. While digital records can be easily maintained and don't require much space as such. This system reduces the efforts needed in generation of digital records from hard copies of attendance.

SCOPE:
This study focuses on the maintenance of digital attendance records of students. For the current state this system is in, it only concerns the teachers and faculty members. The students wouldn't have any use of this system as it maintains the attendance records of students. This is due to the fact that only the faculty members will have access to this system and interact with it.

This system can be transformed into various other systems such as employee management systems, log maintenance systems, etc.

REQUIREMENTS:
- **Hardware Requirements:**
  - Raspberry Pi 3 single-board computer
  - Raspberry Pi Camera Module
  - Internet Connection
- **Software Requirements:**
  - Linux OS / Raspbian OS
  - Python 2.7/3+
  - OpenCV 2.4/3+
  - Moodle API

INTRODUCTION:

Each teacher teaches several students from different classes and fields. Maintaining an attendance record of these many students is a very tedious job to do. These also need to be maintained in a digital format. There is a prominent amount of human error possible in doing so. This system focuses on providing aid to the teachers in this context.

The teachers need to maintain a physical copy of the attendance sheet. The system takes a photo of the sheet with the help of a camera module attached to the Raspberry Pi. This image is given as an input to a program stored on the Raspberry Pi. Then the program extracts information for the current attendance sheet such as:

- Subject Name
- Students' Roll Numbers
- Attendance of each student

The output of this step is an attendance record in csv format which is then mapped onto the Moodle API. This information is then used by a module that then uploads the attendance generated on respective Moodle site(s).

The most prominent feature about this system is that it's automated. The faculty member only needs to place the attendance sheet and scan it. All the other tasks such as retrieving data, manipulating it for use, performing operations on it, storing it and then uploading it on the site, is all automated and done by the Raspberry Pi.

DETAILED EXPLANATION

I. Image Capture

We run a python script to check whether a button is pressed. If a button press is detected, we use the picamera library on the raspberry pi to capture the image through an external camera module. The button should be pressed when keeping the attendance sheet from PES MCOE below the camera on a contrasting surface (preferably black) in a well lit environment.

The image is then saved in a specific directory “input” which is the input directory for the next module. After the image is saved, we move on to the next step in the program.

II. Document Detection

The image is opened in grayscale mode of opencv. We apply Gaussian blur to the image to allow the soft edges to disappear and detect the entire document.

Code

cv2.GaussianBlur(gray, (5, 5), 0) #Here (5, 5) is the kernel size

###

After applying the gaussian blur filter, we use an edge detection algorithm (Canny edge detection) which is available through python opencv library.

Canny Edge Detection:

Canny Edge Detection is a popular edge detection algorithm. It is a multi-stage algorithm.

a) Noise Reduction

Since edge detection is susceptible to noise in the image, the first step is to remove the noise in the image with a 5x5 Gaussian filter. We have already seen this in previous chapters.

b) Finding Intensity Gradient of the Image:

Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (G_x) and vertical direction (G_y).

c) Non-maximum Suppression:

After getting gradient magnitude and direction, a full scan of the image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, a pixel is checked if it is a local maximum in its neighborhood in the direction of gradient.

d) Hysteresis Thresholding:

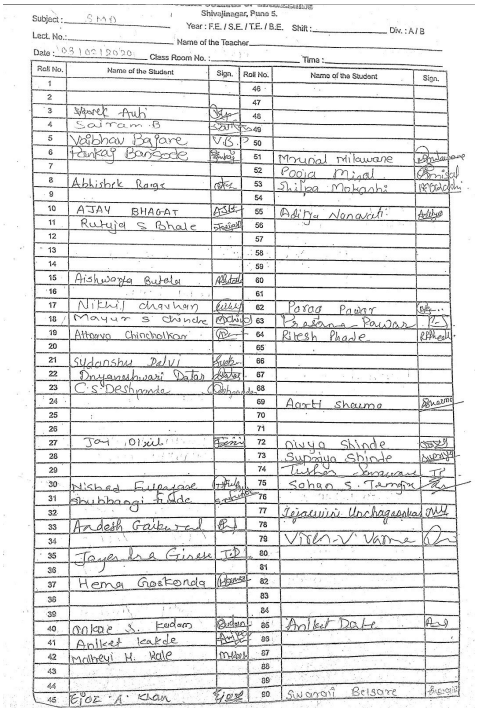
This stage decides which all edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. **Thresholding:**

The resultant code gives us the coordinates of the edges of the document. We warp the image to flatten it. After flatten operation, we use a thresholding algorithm from opencv to increase the contrast in the image.### Code

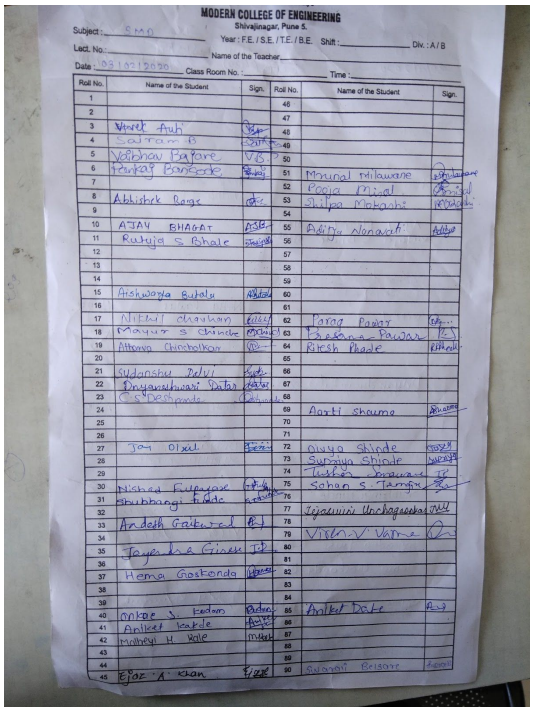
T = threshold_local(warped, 11, offset = 10, method = "gaussian")

warped_thresh = (warped > T+5).astype("uint8") * 255

Example:



Original



Warped and threshold applied

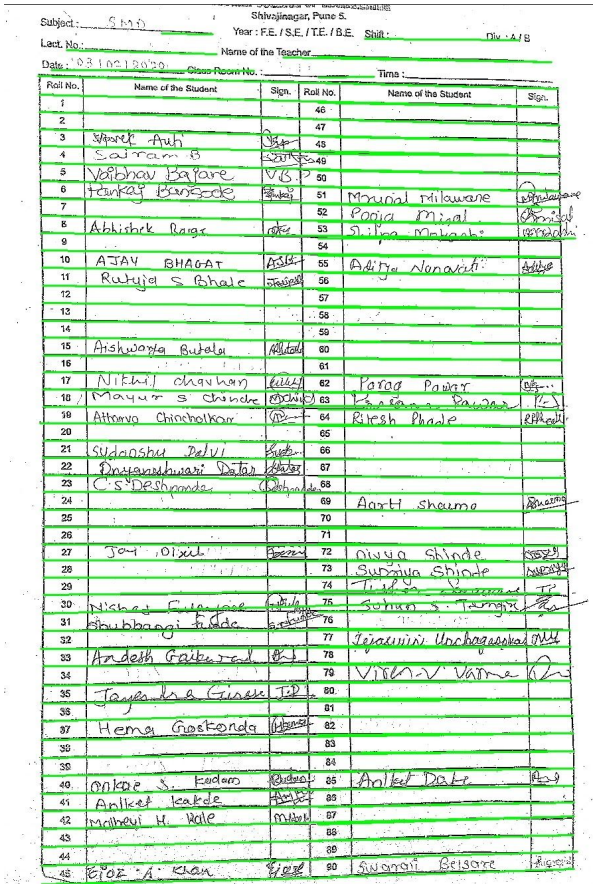
III. Line(s) Detection

The image is now ready for line detection. The attendance sheet has horizontal lines between which the attendance is written. We can exploit this for attendance detection by carrying out the following steps:

1) Detect the straight horizontal lines by checking if there is an adjacent pixel which is dark and progressively move towards the right.
2) Record the starting point and ending point of the line.
3) Store the lines in a list.
4) Remove lines which are very short by comparing x-coordinates.

This has given us the required lines which we can then use for attendance detection.

Detected Lines:

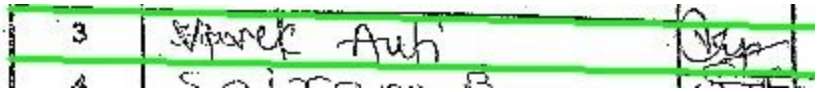


IV. Attendance Detection

As attendance is written between the lines, we can detect the number of dark pixels between two lines to determine whether the student was present. Higher value of dark pixels indicates that something was written in that space.

For achieving this we follow the following steps:

1) For each line pair, calculate their slopes and constants using: y=mx+b line equation and their coordinates. INIT count=0.
2) For each pixel above the minimum y-coordinate of the top line and below the maximum y-coordinate of the bottom line, check if the pixel lies between the lines by using the line equations.
3) If the pixel lies between lines, check its value. If pixel value < 80 (Parameter in function) add pixel to the count.
4) Repeat for all pixels between the two lines.
5) At the end, check if count > threshold_for_present (Parameter in function). If yes, mark as present. Else mark as absent.
6) Return attendance list.
7) Save as csv file and print to stdout.



Pixel Value: 1521

Pixel value: 258

For the attendance sheet, we get the output:

['1:A', '2:A', '3:P', '4:P', '5:P', '6:P', '7:A', '8:P', '9:A', '10:P', '11:P', '12:A', '13:A', '14:A', '15:P', '16:A', '17:P', '18:P', '19:P', '20:A', '21:P', '22:P', '23:P', '24:A', '25:A', '26:A', '27:P', '28:A', '29:A', '30:P', '31:P', '32:A', '33:P', '34:A', '35:P', '36:A', '37:P', '38:A', '39:A', '40:P', '41:P', '42:P', '43:A', '44:A', '45:P', '46:A', '47:A', '48:A', '49:A', '50:A', '51:P', '52:P', '53:P', '54:A', '55:P', '56:A', '57:A', '58:A', '59:A', '60:A', '61:A', '62:P', '63:P', '64:P', '65:A', '66:A', '67:A', '68:A', '69:P', '70:A', '71:A', '72:P', '73:P', '74:P', '75:P', '76:A', '77:P', '78:A', '79:P', '80:A', '81:A', '82:A', '83:A', '84:A', '85:P', '86:A', '87:A', '88:A', '89:A', '90:P']

Which is 100% accurate due to single sample size. We expect accuracy to be about 99% for attendance sheets using recommended conditions.

Code available on : https://github.com/atharvac/image-slicer

V. Subject Detection

The subject written on the attendance sheet will be in human handwriting. We can recognise handwritten words in an image but the computer can't. Thus we need to make the computer learn how to recognise the handwritten words. This is where neural networks come in.

We will use a HTR(Handwritten Text Recognition) model which will recognise the text contained in the images. We will be using a model that is based on SimpleHTR developed by Harald Scheidl with a couple of changes. The principle input and output of a HTR is depicted below:

The HTR takes an image as input and guesses what the word is based on the logic it has formed during the training of the model. The implementation depends on numpy, cv2 and tensorflow imports. It consists of 5 CNN layers, 2 RNN (LSTM) layers and the CTC loss and decoding layer. The description of procedure is given below:

- The input image is a gray-scale image and has a size of 128x32.
- 5 CNN layers map the input image to a feature sequence of size 32x256.
- 2 LSTM layers with 256 units propagate information through the sequence and map the sequence to a matrix of size 32x80. Each matrix-element represents a score for one of the 80 characters at one of the 32 time-steps.
- The CTC layer either calculates the loss value given the matrix and the ground-truth text (when training), or it decodes the matrix to the final text with best path decoding or beam search decoding.

VI. Moodle Interfacing

For transferring the data from the csv file to the Moodle server we use the Selenium web automation tool.

Selenium Web Automation-

Selenium is an umbrella project for a range of tools and libraries that enable and support the automation of web browsers.

It provides extensions to emulate user interaction with browsers, a distribution server for scaling browser allocation, and the infrastructure for implementations of the W3C WebDriver specification that lets us write interchangeable code for all major web browsers.

Selenium allows users to simulate common activities performed by end-users; entering text into fields, selecting drop-down values and checking boxes, and clicking links in documents. It also provides many other controls such as mouse movement, arbitrary JavaScript execution, and much more.

Although used primarily for front-end testing of websites, Selenium is at its core a browser user agent library. The interfaces are ubiquitous to their application, which encourages composition with other libraries to suit your purpose.

Moodle Learning Management System-

Moodle is a free and open-source learning management system (LMS) written in PHP and distributed under the GNU General Public License.

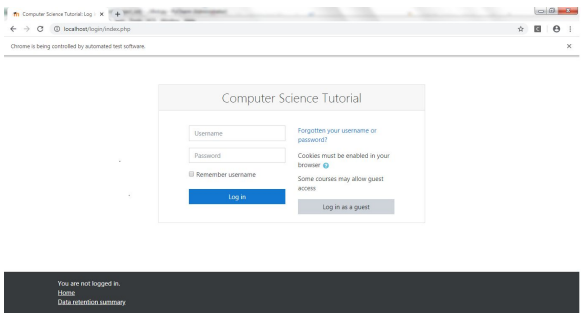
Developed on pedagogical principles, Moodle is used for blended learning, distance education, flipped classroom and other e-learning projects in schools, universities, workplaces and other sectors.

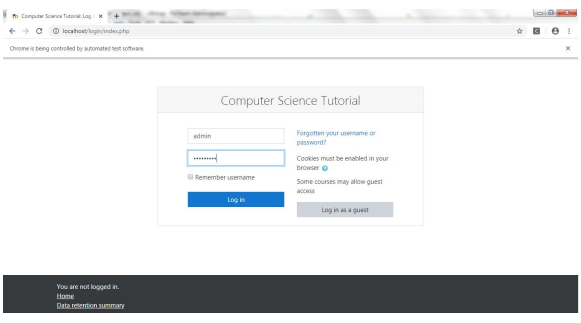
With customizable management features, it is used to create private websites with online courses for educators and trainers to achieve learning goals.

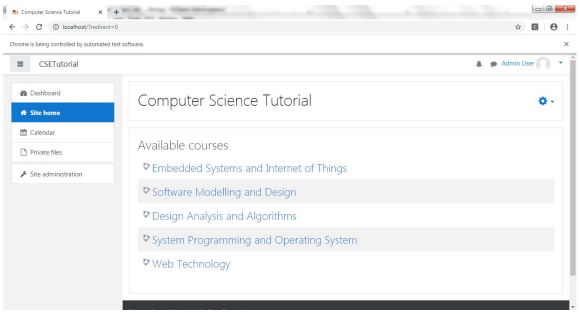
Moodle allows for extending and tailoring learning environments using community-sourced plugins.

The process of automating the software involves -

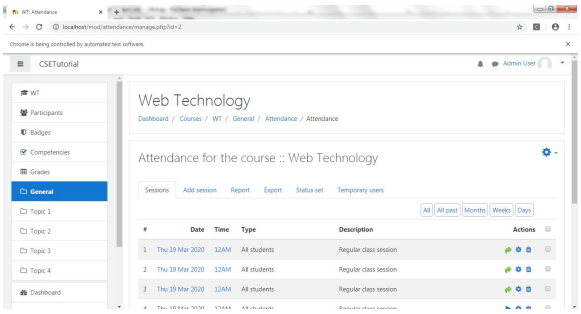
1. **Browsing the Moodle server** on the test browser and logging in into Moodle through the code. Here Chrome web driver is used for accessing the chrome web browser and navigating to the Moodle login page.



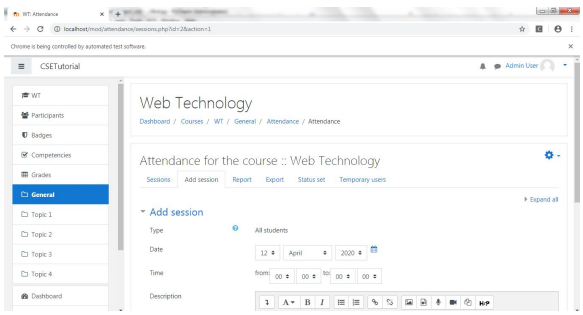


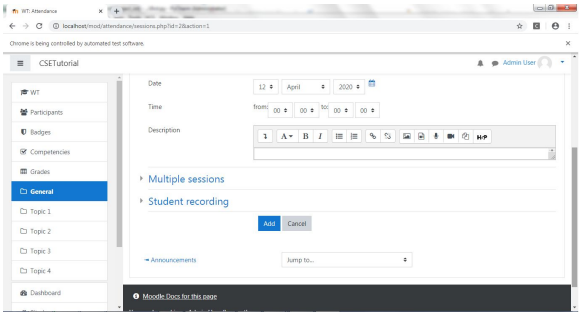


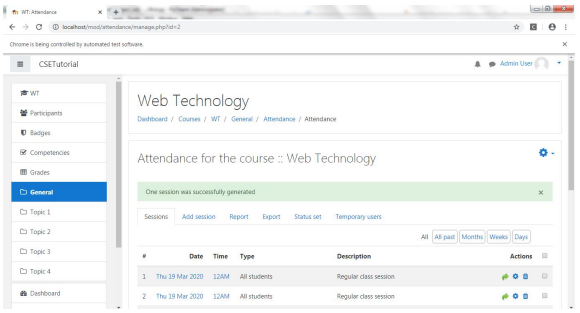
2. **Reading the data** and the subject from the csv file attendance_file.csv and entering the data into Moodle. The csv file contains the subject name and the attendance of the students that has been extracted.



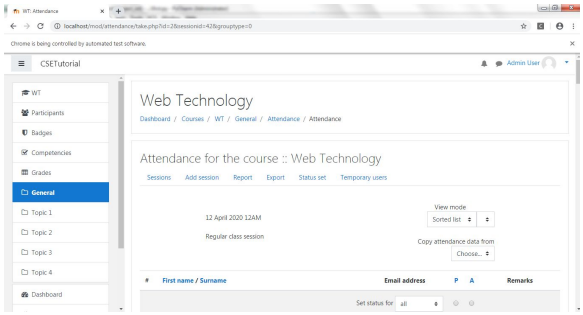
3. **Creating an instance of the attendance sheet** for the respective subject. The date and time of the attendance data entered are recorded.

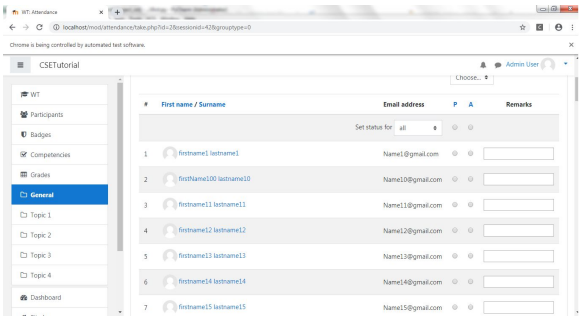


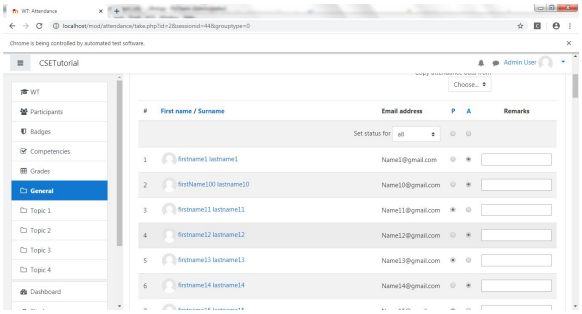


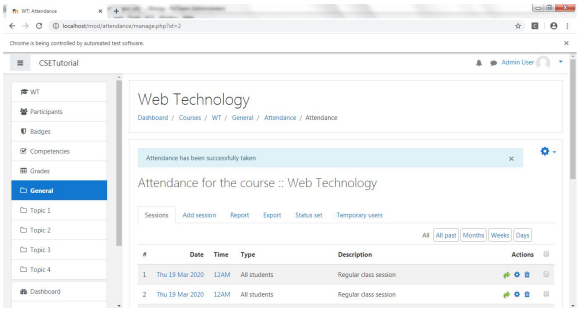


4. **Marking the attendance** of all the students and saving it. The attendance is marked as per the data read from the csv file.

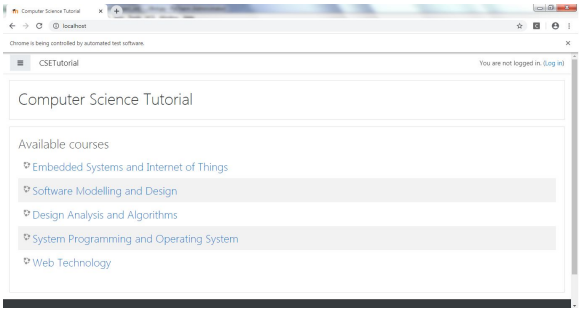








5. **Logging out of Moodle** and closing the browser window. Once the attendance sheet is filled with the data the session is destroyed after logging out of Moodle server.



6.Click on the given link to watch the sample video - Sample Video.mp4

ADVANTAGES

1) It's a simple and efficient way for students who want to keep track of their attendance in various subjects.

2) Students can track their attendance on the college server so students can check their attendance marked for the academic year.

3) If a student forgets to mark his/her attendance on the attendance sheet on a particular day, he/she will know. (In short, students will be able to recheck their attendance)

4) Easy to manage attendance for teachers. Teachers don't need to keep the filled attendance sheets with them throughout the year as all the information will be available for them on our application.

5) This application is very convenient for teachers as everything takes place at a single click of a button.

DISADVANTAGES

1) Students can track their attendance on the college server, attendance tracking outside the college is not possible.

2) Though this project can automate the process of storing the attendance in the database, it is still incapable of eliminating the efforts taken in marking the attendance by the students in the attendance sheet.

CHALLENGES

1) The fixed nature of the attendance sheet meant that we had to work around the problems of mixing of writing strokes on the sheet.

2) The decision to go with Neural Network was also a result of the immutable attendance sheet format, As we could not adopt the MCQ exam strategy of filling in the blank circles for a choice.

3) The document detection only works when there is contrast between the document and its background.

4) The pixel value threshold is difficult to set without knowing the lighting conditions.

5) Due to the unavailability of the hardware, we might need to recalibrate the parameters of the code for best results once deployed on the raspberry pi.

CONCLUSION

We have leveraged the powerful computing capabilities of the raspberry pi to perform complex operations in order to get attendance of students from an image.
The IOT-Readiness of the raspberry pi let us extend its capabilities with a camera module which is essential for the working of this project.

REFERENCES

1. The Selenium Browser Automation Project :: Documentation for Selenium
2. Selenium with Python — Selenium Python Bindings 2 documentation
3. MoodleDocs
4. SimpleHTR by Harald Scheidl
5. Warp image
6. Build a Handwritten Text Recognition System using TensorFlow
7. Document scanner
8. Picamera
9. Sentdex-OpenCV with Python for Image and Video Analysis