Researching an Attendance System

Sponsored by the WPI Choral Association

An Interactive Qualifying Project

Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
In partial fulfillment of the requirements for the
Degree of Bachelor of Science

By:

_____

Sebastian Bellisario, Mechanical Engineering/Humanities and Arts

_____

Andrew Smith, Chemistry

May 2013

Approved:

_____

John Delorey, Advisor

_____

Vincent Manzo, Advisor

Table of Contents

Abstract

The purpose of this project was to provide the Choral Association a recommendation for attendance tracking software which met a variety of criteria. The software had to be dynamic enough to suit the needs of the organization for the coming years. Several products and apps were researched and were found unable to meet the requirements. The project group then began coding a custom attendance software which suited the needs of the Choral Association.

Introduction

Recording attendance for an organization is an important administrative task providing information about the members as well as determining factors within the organization. However, taking attendance at large meetings often leads to wasted time and inaccuracies. Several of the choral organizations on campus suffer from these shortcomings. Festival Choir, Glee Club, and Alden Voices hold rehearsals of over 100 people, and have a vested interested in conserving wasted rehearsal time. The groups use their attendance to determine eligibility to sing at concerts and grades for those receiving academic credit. The responsibility of the task currently falls to the officers, specifically the Secretary and Stage Manager. To streamline their attendance process, Professor Delory and Professor Manzo sponsored an IQP. The goal was to research an accurate and efficient manner to take and record attendance, and to recommend a solution to our sponsors.

A set of design specifications were created listing potential features and ranking their importance to the sponsor. One important criterion was the use of RFID or Coded Magnetic Strips to electronically determine attendance. This requirement would allow for a simplistic and effective process where each member need only to swipe or tap their ID to have their attendance recorded with very high accuracy. The system needed to be robust and dynamic, allowing for definitive and accurate records while leaving room for future changes to meet new expectations. The IQP team used these guidelines to research potential solutions, either through implementation of existing programs or through design of a prototype.

There are several existing attendance programs which could potentially be used. The team investigated Attendance app, TimeTrex Time and Attendance Software, The Attendance

App, Check-In Easy, Attendatron by Dan, and iAttendance but none of them addressed the full range of parameters. The team began designing a prototype system. Conceptual designs were developed for interface organization and generated reports. The end result is a fairly mature prototype system which gathers attendance quickly through the use of RFID chips, and presents the information in appropriate formats.

The development was done with a greater context included. The users have no need to understand how the system is functioning but it is imperative that they are able to use the system correctly. Streamlining the user interface was integral to the use of the program. Carefully organizing the code allows the prototype to be enhanced and to have new features added to the system in the future.

Background and Research

The WPI Choral Association is made up of the WPI Men's Glee Club, Alden Voices, their associated A Cappella groups, WPI Festival Chorus, Vocal Performance Lab, and other choral ensembles. As part of the Music Division of the Department of Humanities and Arts, the Glee Club and Alden Voices are both extra-curricular activities as well as academic classes. Not all members of these two choral groups are enrolled for academic credit; approximately 30% of the members of Glee Club and Alden Voices for credit each year. In addition, accurate attendance is necessary as members of the A Cappella groups are required to maintain good standing. Lastly, scholarships are offered to individuals who maintain good standing and need financial assistance to participate in tour events.

Students who participate in Glee Club or Alden Voices for academic credit must attend all rehearsals and concerts unless excused by the Director. Students in Glee Club and Alden Voices have the opportunity to audition for the A Cappella groups Simple Harmonic Motion and Technicords respectfully. Members of the A Cappella groups Simple Harmonic Motion and Technichords, are expected to maintain good attendance. Glee Club and Alden Voices each rehearse Tuesday evening with a Thursday evening combined rehearsal. Members are allowed to miss two rehearsals in a semester and must attend a make-up rehearsal for each additional missed rehearsal to remain in good standing.

In previous years, the secretary of each group used a clipboard to mark attendance at each rehearsal. Though accurate in keeping attendance, previous secretaries have noted it as a time consuming task that often caused the secretary to miss most of the rehearsal. This information was also not readily available because the secretary had to either copy the attendance sheet to

provide to the Director after each rehearsal or enter attendance into a spreadsheet each night, consuming additional time and resources. More often than not, these attendance records were not brought to the director in a timely manner, creating a backlog of missing attendance records. In 2010, Daniel Lettiere and Amanda Eaton began work on an automated attendance system which would scan Student ID cards at each rehearsal, maintain records on each student's attendance, and build reports based on standing to provide to the Director. This system, known as the Attendatron, relieved much of the workload and time commitment from the secretaries by tracking attendance in CSV style spreadsheets and a Microsoft Access database. However, after only a couple of years of successful use, both creators graduated and did not leave enough information behind to maintain the system. With no knowledge of how to maintain and continue using the Attendatron, its usefulness diminished and attendance returned to markings on a clipboard.

Throughout the process of this IQP, other potential systems were researched that other groups have used to find a solution to a lack of proper attendance tracking systems in the Choral Association:

iAttendance by Pavel Braun is an app made for iPhone and iPad and works with a group of preregistered individuals to keep track of attendance. This product does not allow individual attendance reports, which was required by the sponsors.

TimeTrex Time and Attendance Software is by far the closest product to the sponsor's parameters. It collects individual attendance at various events through a variety of methods, such as bar codes, proximity scanning, card swipe, and more. Unfortunately, the cost of the software far outweighs its usefulness for the number of members in the Choral Association.

Check-In Easy by Charity Happenings, LLC is an iPhone/iPad app and works exceptionally as a way to collect attendance for a large number of people at an event. The app does not keep records of multiple events and cannot meet the requirements of multi-event tracking without external software.

Attendance by David Reed is another app for iPhone/iPad. It keeps track of attendance for a large number of individuals in multiple groups for many, many different events. The app does not have support for card swiping, proximity readers, or bar code scanning and thus does not meet the requirements of the sponsors.

The Attendance App by Prosjektlab is an app for iPhone and iPad which can track attendance for many individuals in many groups. The app does not allow for any particular information about individuals to be stored, such as voice part, standing, and other information required by the sponsors.

The feasibility of creating a new system was also investigated. The system would address the needs of the organization, while maintaining support after the project ends. Our initial concerns with creating a new system was its complexity and whether or not an effective system could be created within the scope of the project. As the programs that were investigated were either too general or overly complex, a new prototype system was seriously considered to address the specific needs of the Choral Organization. Through researching Python, MySQL, and other systems on the WPI servers, the team determined a prototype was feasible. Ultimately the IQP recommended a new system be developed, and with the sponsors approval began implementation.

Methodology

The sponsors had many ideas for what features the Attendance System should provide. The sponsors demanded that the system took accurate attendance in a timely, efficient manner and not interfere with rehearsal. The system would also act as a way to store information about each student, such as graduation year, A Cappella groups, voice part, and more as determined by the sponsors. The system needed to have open event creation by demand and allow for a variety of different event types, including implementation of make-up rehearsals and concerts. The sponsors also asked for emails after each rehearsal with reports containing information about attendees and absences. Finally, the sponsors wanted a system which could be updated and changed in future years to adapt to changing expectations and needs.

With an outline of overall system functionality, the team soon discovered the necessity for a system that focused on student data and how different events affected this data. Our next step was to meet with our sponsors to build a list of data to include for each student. During a sponsor meeting a list of 12 parameters was generated. The sponsors also gave feedback on make-up rehearsals and attendance reports. As a result, the system needed to have fluidity in scheduling rehearsals to account for unscheduled make up rehearsals. The reports needed to be accessible to the officers but would only directly be sent to the Director and the Secretary. The secretary would then be responsible for contacting individual members. Finally, the team met to discuss the list of parameters. While all twelve we're relevant to our sponsors, a focus on prioritizing and selecting the most important parameters was necessary. Five parameters were ultimately selected: Standing, Voice Part, Class Year, Choral Groups, and Concert Roster.

With a firm understanding of the sponsor's expectations, the team began development of the prototype.

Results

Throughout the work on this IQP, there were many specific uses for an Attendance System in the Choral Association. While the main function of the program was to take attendance via a swipe of a Student ID, the system also needed to maintain logs, send reports, hold student information, and keep track of large amounts of data while remaining open for future developments.

The proposed Attendance System in this project uses the WPI MySQL server to store information about Students, Events, Excuses, and Reports, maintaining accurate accounts of which members were at which rehearsals and which members were excused. The program uses an HTML/CSS display through the WPI CCC server to be accessible via anywhere with a stable internet connection. Using Python coding, the web display accesses, modifies, and maintains the MySQL database while still being adaptable and fluid.

The System is split into four main sections: Students, Events, Excuses, and Reports. The Students section houses the "Add Student" and "List Students" menus that are both used to create, modify, and delete students from the database. These menus also display information about each student, allowing Secretaries, Stage Managers, and other Officers to quickly ascertain certain details about a particular individual. In the Events section, events are created at the beginning of a rehearsal, concert, make-up rehearsal, or optional rehearsal for the various groups and members sign in using their Student ID cards by tapping them against the attached RFID reader. This section is the most important part of the system because it is where attendance is collected for each rehearsal. After closing, the system sends a report to the Secretary and the Director via email and displays a Rehearsal Roster on the screen. The system interface can be

viewed in the appendix. A test version can be viewed at

http://users.wpi.edu/~amsmith514/attendance-IQP/

Password: Attendance

Conclusions

While many systems for electronic attendance exist, most are designed for an alternate purpose. Our sponsors needed a system to address their specific needs. The Attendance System proposed by our IQP group was universally accepted by the Officers of the Glee Club and Alden Voices. Though the system is only a prototype, it solved many of the problems that had been encountered by previous solutions and systems. The ease of setup combined with the ability to use the system anywhere was greatly admired by the groups. The speed of the system allows for instant access to the attendance without any additional work for the Secretary and the RFID reader quickly records each student's attendance. The prototype has successfully been used for a semester, and continues to be the primary method of attendance for the Choral Association.

Further Research and Development

Although the Attendance System fits the critical parameters indicated by our Sponsors, there are some improvements and developments which would make the system better. As it is now, the system does not distinguish between different types of standing. For example, there are some concerts which are required only for those enrolled in the courses for Glee Club and Alden Voices. These concerts and the rehearsals for them, which are not required for Officers and A Capella singers, may affect their standing in ways they should not. In addition, there is currently no way to add a "blanket" excuse for recurring commitments such as ID 2050 and evening graduate classes. This forces both the Secretary and the member in question to continuously communicate the same excuse for the duration of the term, adding a layer of redundancy which is both unnecessary and time consuming. Finally, there is not yet an archive for members who are no longer students at WPI or no longer members of the choral groups. Thus, whenever someone must be deleted for these reasons, there is no way to return their data if they return or if their information is needed.

The Attendance System can be expanded in many ways to accommodate different organizations and needs of the Director. Typically both graduation year and major are displayed on the Roster of active members. While the system currently tracks graduation year, it does not track majors nor does it display either in the Roster output. This would be a very simple problem to solve with the system as it is, though it would involve data collection from nearly 100 individuals which can be a daunting task. In addition to majors, the system could be expanded to work with Choral Folders to check them in and out at concerts or rehearsals. The system could

be adapted into a standalone-software which does not require an internet connection. It could also be more generalized to work with other choral groups in the nation.

The Attendance System must be properly maintained by the Secretaries of the Choral Association. Each year when new members join, they must be added to the system through the "Add Student" menu. Current members must also have their system data changed if their information changes, such as if they receive a new ID card or if they join one of the A Cappella groups. At the end of each semester, the system must be cleared of data from the semester to prepare for the next semester via the "New Semester" menu. Altogether, this system, like the Attendatron before it, will lose its usefulness if it is not properly maintained at recurring intervals of time.

Appendix:

A. Pictures of the user interface

B. Code Database

C. External References

Event Type:	Rehearsal	Group:	Glee Club	Date:	2013-01-15

Attendance	Groups	Credit	Officer
	None	Yes	No
	None	No	Yes
	Simple Harmonic Motion	No	No
	Simple Harmonic Motion	No	No
	None	Yes	No
	None	Yes	No
	Simple Harmonic Motion	No	No
	None	Yes	No
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	No
	Simple Harmonic Motion	No	No
	None	Yes	No
	Audiophiles	No	No
	None	No	No
	None	No	Yes
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	No
	None	No	Yes
	None	Yes	No
	None	No	No
	None	Yes	No
	Simple Harmonic Motion	No	No
	None	Yes	Yes
	Audiophiles	Yes	No
	None	Yes	Yes
	Simple Harmonic Motion	Yes	No
	None	No	No
	Audiophiles	No	No
	Simple Harmonic Motion, Audiophiles	Yes	No
	Simple Harmonic Motion	Yes	No
	Audiophiles	Yes	Yes
	None	No	No
	None	No	No
	None	No	No
	None	No	Yes
	None	No	No
	None	Yes	Yes
	Simple Harmonic Motion	No	No
	Audiophiles	No	No
	None	No	No
	None	Yes	No
	None	Yes	No

Absent	Groups	Credit	Officer
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	No
	None	No	No
	None	Yes	No
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	No

Late	Groups	Credit	Officer
Excused	Groups	Credit	Officer
	None	No	No
	None	Yes	No
	None	No	Yes
	None	No	No
	None	Yes	No

Figure 1: Rehearsal Report

Currently Not in Good Standing			
Name	**Groups**	**Credit**	**Officer**
	None	Yes	Yes
	None	Yes	No
	None	No	No
	Simple Harmonic Motion	No	No
	Simple Harmonic Motion	No	No
	Simple Harmonic Motion	No	No
	Simple Harmonic Motion	No	No
	None	Yes	No
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	No
	Simple Harmonic Motion	No	No
	Audiophiles	No	No
	None	No	No
	None	No	Yes
	Simple Harmonic Motion	No	No
	None	No	No
	None	No	Yes
	None	No	No
	None	No	No
	None	No	No
	None	No	No
	None	No	No
	Simple Harmonic Motion	No	No
	None	Yes	Yes
	Simple Harmonic Motion	Yes	No
	Simple Harmonic Motion	No	No
	None	No	No
	Audiophiles	No	No
	Simple Harmonic Motion, Audiophiles	Yes	No
	Simple Harmonic Motion	Yes	No
	Audiophiles	Yes	Yes
	None	No	No
	None	No	No
	None	No	No
	None	No	Yes
	None	No	No
	Simple Harmonic Motion	No	No
	Audiophiles	No	No

Figure 2: Standing Report

Name	Voice Part	Graduation Year
	Tenor 2	2013
	Tenor 2	2016
	Tenor 2	2015
	Bass 1	2014
	Bass 1	2014
	Bass 1	2016
	Bass 1	2015
	Bass 1	2015
	Bass 1	2016
	Bass 2	2016
	Bass 2	2014
	Bass 2	2015
	Bass 2	2016
	Bass 2	2013
	Bass 2	2015
Total:	15 Students	

Back Reports Menu

Figure 3: Roster Report

Code Database

The following scripts are used throughout the program for formatting and common programs.
engine.py code:

```
import datetime
import MySQLdb
import os


def __log(error):
    l = open('error.log','a')
    l.write(datetime.datetime.now().strftime('At %H:%M:%S On %m-%d-%y :: '))
    pwd = os.path.dirname(__file__)
    l.write(pwd)
    l.write(' :: ')
    l.write(str(error))
    l.write('\n')
    l.close()


def __get_conn():
    try:
        return MySQLdb.connect (host='mysql.wpi.edu',user='choralattnd',passwd='[redacted]',db='attendance')
    except Exception, e:
        __log(e)
        return None


def printheader(title):
    p = open('header','r')
    print 'Content-Type: text/html\n'
    for i in p.readlines():
        n = i.find('<!-- TITLE -->')
        if n != -1:
            print i[0:n]+title+i[n+14:]
        else:
            print i
    p.close()


def printfoot():
    p = open('footer','r')
    for i in p.readlines():
        print i
    p.close()


def eventCreate(type, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        termID = findcurterm()
        cur.execute('insert into eventBase (eventDateTime, eventType, eventStatus, eventChorus, event_termBase_termID)
values (NOW(), "%s", "1", "%s", "%s")'%(str(type), str(chorus), str(termID)))
        cur.close()
        conn.commit()
        conn.close()
        eventID = findEvent()
```

```
        return eventID
    except Exception, e:
        __log(e)
        return None


def findEvent():
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select eventID from eventBase where eventStatus = 1')
        eventID = cur.fetchone()[0]
        cur.close()
        conn.close()
        return eventID
    except Exception, e:
        __log(e)
        return None


def addAttend(RFID, eventID, eventChorus, late):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('insert into attendBase (attend_RFID, attend_eventBase_eventID, attend_late) values ("%s", "%s",
"%s")'%(str(RFID), str(eventID), str(late)))
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return


def addStudent(fname, lname, studentID, RFID, officer, acgroup, credit, voicePart, gradYear, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('insert into studentBase (fname, lname, studentID, RFID, officer, acgroup, credit, voicePart, gradYear,
attended, missed, excuse, chorus) values ("%s", "%s", "%s", "%s", "%s", "%s", "%s", "%s", "%s", 0, 0, 0,
"%s")'%(str(fname), str(lname), str(studentID), str(RFID), str(officer), str(acgroup), str(credit), str(voicePart), str(gradYear),
str(chorus)))
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return


def delStudent(RFID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('delete from studentBase where RFID = %s'%str(RFID))
        cur.execute('delete from standingBase where RFID = %s'%str(RFID))
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return
```

```
def findStudent(fname, lname):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select RFID from studentBase where fname = "%s" and lname = "%s" order by RFID desc' %(str(fname),
str(lname)))
        RFID = cur.fetchone()[0]
        cur.close()
        conn.close()
        return RFID
    except Exception, e:
        __log(e)
        return None


def updateStudent(fname, lname, RFID, officer, acgroup, credit, voicePart, gradYear):
    try:
        RFID1 = findStudent(fname,lname)
        conn = __get_conn()
        if RFID != RFID1:
            RFID2 = RFID
            cur1 = conn.cursor()
            cur1.execute('update attendBase set attend_RFID="%s" where attend_RFID="%s"'%(str(RFID2), str(RFID1)))
            cur1.execute('update standingBase set RFID="%s" where RFID="%s"'%(str(RFID2), str(RFID1)))
            cur1.execute('update excuseBase set excuseRFID="%s" where excuseRFID="%s"'%(str(RFID2), str(RFID1)))
            cur1.close()
        else:
            RFID2 = RFID1
        cur2 = conn.cursor()
        cur2.execute('update studentBase set RFID="%s", officer="%s", acgroup="%s", credit="%s", voicePart="%s",
gradYear="%s" where RFID="%s"'%(str(RFID2), str(officer), str(acgroup), str(credit), str(voicePart), str(gradYear),
str(RFID1)))
        cur2.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return


def newTerm(term, year):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('delete from standingBase')
        cur.execute('update studentBase set attended = 0, missed = 0, excuse = 0')
        cur.execute('update termBase set active = 0')
        cur.execute('insert into termBase (term, year, active) values ("%s", "%s", 1)'%(str(term), str(year)))
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return


def findcurterm():
    try:
        conn = __get_conn()
        cur = conn.cursor()
```

```
        cur.execute('select max(termID) from termBase')
        termID = cur.fetchone()[0]
        cur.close()
        conn.commit()
        conn.close()
        return termID
    except Exception, e:
        __log(e)
        return None

def update_studentBase(eventID, eventType, eventChorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        if eventType < 2:
            cur.execute('update studentBase set missed=missed+1 where chorus="%s" and RFID not in (select attend_RFID
from attendBase where attend_eventBase_eventID = "%s")'%(str(eventChorus), str(eventID)))
            cur.execute('update studentBase set attended=attended+1 where chorus="%s" and RFID in (select attend_RFID from
attendBase where attend_eventBase_eventID = "%s" and attend_late = "0")'%(str(eventChorus), str(eventID)))
            cur.execute('update studentBase set late=late+1 where chorus="%s" and RFID in (select attend_RFID from
attendBase where attend_eventBase_eventID = "%s" and attend_late = "1")'%(str(eventChorus), str(eventID)))
        elif eventType == 2:
            cur.execute('update studentBase set attended=attended+1 where chorus="%s" and RFID in (select attend_RFID from
attendBase where attend_eventBase_eventID = "%s")'%(str(eventChorus), str(eventID)))
            cur.execute('update studentBase set missed=missed-1 where chorus="%s" and RFID in (select attend_RFID from
attendBase where attend_eventBase_eventID = "%s")'%(str(eventChorus), str(eventID)))
        elif eventType == 3:
            cur.execute('update studentBase set attended=attended where chorus="%s" and RFID in (select attend_RFID from
attendBase where attend_eventBase_eventID = "%s")'%(str(eventChorus), str(eventID)))
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return



def closeEvent(eventID):
    try:
        today = datetime.date.today()
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select eventType from eventBase where eventID="%s"'%str(eventID))
        eventType = cur.fetchone()[0]
        cur.execute('select eventChorus from eventBase where eventID="%s"'%str(eventID))
        eventChorus = cur.fetchone()[0]
        if eventChorus == 2:
            update_studentBase(eventID, eventType, 0)
            update_studentBase(eventID, eventType, 1)
            cur.execute('insert into reportBase (report_eventBase_eventID, report_eventBase_eventType, report_chorus,
report_date) values  ("%s", "%s", "%s", "%s")'%(str(eventID), str(eventType), str(0), str(today)))
            cur.execute('insert into reportBase (report_eventBase_eventID, report_eventBase_eventType, report_chorus,
report_date) values  ("%s", "%s", "%s", "%s")'%(str(eventID), str(eventType), str(1), str(today)))
            cur.execute('select report_number from reportBase where report_eventBase_eventID="%s" and
report_chorus="0"'%str(eventID))
            numberGlee = cur.fetchone()[0]
            cur.execute('select report_number from reportBase where report_eventBase_eventID="%s" and
report_chorus="1"'%str(eventID))
            numberAlden = cur.fetchone()[0]
            mailReport(numberGlee, "gc-secretary@wpi.edu", 0)
            mailReport(numberAlden, "av-secretary@wpi.edu", 1)
```

```
            mailReport(numberGlee, "jfd@wpi.edu", 0)
            mailReport(numberAlden, "jfd@wpi.edu", 1)
        else:
            update_studentBase(eventID, eventType, eventChorus)
            cur.execute('insert into reportBase (report_eventBase_eventID, report_eventBase_eventType, report_chorus,
report_date) values  ("%s", "%s", "%s", "%s")'%(str(eventID), str(eventType), str(eventChorus), str(today)))
            cur.execute('select report_number from reportBase where report_eventBase_eventID="%s"'%str(eventID))
            number = cur.fetchone()[0]
            if eventChorus == 0:
                email = "gc-secretary@wpi.edu"
            elif eventChorus == 1:
                email = "av-secretary@wpi.edu"
            mailReport(number, email, eventChorus)
            mailReport(number, "jfd@wpi.edu", eventChorus)
        cur.execute('update eventBase set eventStatus=0')
        cur.execute('update standingBase set isNew = "0"')
        cur.close()
        conn.commit()
        conn.close()
        update_standingBase()
    except Exception, e:
        __log(e)
    return


def mailReport(number, email, chorus):
    try:
        term = getTerm()
        m = os.popen('/usr/sbin/sendmail -t','w')
        m.write('Content-type: text/html\n')
        m.write('From: attendance@wpi.edu\n')
        m.write('To: %s\n'%str(email))
        m.write('Subject: New Report Available\n\n')
        m.write('<html><body>A new report is available for viewing.<br />')
        m.write('You may view it <a
href="http://users.wpi.edu/~amsmith514/attendance/report/index.cgi?type=0&number=%s&chorus=%s&termID=%s">here
</a><br />\n'%(str(number),str(chorus),str(term)))
        m.write('If there are any issues with this report, please reply to this email.<br /></body></html>')
        m.close()
    except Exception, e:
        __log(e)
    return


def attendEvents(RFID, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_date from reportBase where report_eventBase_eventID in (select attend_eventBase_eventID
from attendBase where attend_RFID = "%s" and attend_late=0) and report_eventBase_eventID not in (select
excuse_eventBase_eventID from excuseBase where excuseRFID = "%s") and report_chorus="%s"'%(str(RFID), str(RFID),
str(chorus)))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def excuseEvents(RFID, chorus):
```

```
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_date from reportBase where report_eventBase_eventID in (select excuse_eventBase_eventID
from excuseBase where excuseRFID = "%s") and report_chorus="%s"'%(str(RFID),str(chorus)))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def lateEvents(RFID, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_date from reportBase where report_eventBase_eventID in (select attend_eventBase_eventID
from attendBase where attend_RFID = "%s" and attend_late=1) and report_chorus="%s"'%(str(RFID),str(chorus)))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def missEvents(RFID, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_date from reportBase where report_eventBase_eventID not in (select
attend_eventBase_eventID from attendBase where attend_RFID = "%s") and report_chorus="%s"'%(str(RFID),str(chorus)))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def listAttend(eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select lname, fname, RFID from studentBase where RFID in (select attend_RFID from attendBase where
attend_eventBase_eventID = "%s") order by lname'%str(eventID))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def listStudents(chorus):
    try:
        conn = __get_conn()
```

```
        cur = conn.cursor()
        cur.execute('select * from studentBase where chorus="%s" order by lname'%str(chorus))
        l = cur.fetchall()
        cur.close()
        conn.close()
        return l
    except Exception, e:
        __log(e)
        return None


def update_standingBase():
    try:
        conn = __get_conn()
        cur1 = conn.cursor()
        cur2 = conn.cursor()
        cur1.execute('delete from standingBase where RFID not in (select RFID from studentBase where missed > 1)')
        cur2.execute('select fname, lname, RFID from studentBase where missed > 1 and RFID not in (select RFID from
standingBase)')
        l = cur2.fetchall()
        for r in l:
            fname = r[0]
            lname = r[1]
            RFID = r[2]
            cur2.execute('insert into standingBase (fname, lname, RFID, isNew) values ("%s", "%s", "%s", 1)'%(str(fname),
str(lname), str(RFID)))
        cur1.close()
        cur2.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return


def listReports(chorus, termID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select * from reportBase where report_chorus=%s and report_eventBase_eventID in (select eventID from
eventBase where event_termBase_termID="%s")'%(str(chorus), str(termID)))
        list = cur.fetchall()
        cur.close()
        conn.close()
        return list
    except Exception, e:
        __log(e)
        return None


def addexcuse(eventID, RFID, excuse, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('insert into excuseBase (excuse_eventBase_eventID, excuseRFID, excuseNotes, excuseChorus) values
("%s", "%s", "%s", "%s")'%(str(eventID), str(RFID), str(excuse), str(chorus)))
        cur.execute('insert into attendBase (attend_RFID, attend_eventBase_eventID, attend_late) values ("%s", "%s",
"0")'%(str(RFID), str(eventID)))
        cur.execute('update studentBase set missed=missed-1, excuse=excuse+1 where RFID="%s"'%str(RFID))
        cur.close()
        conn.commit()
        conn.close()
```

```
        update_standingBase()
    except Exception, e:
        __log(e)
    return


def listExcuses(eventID, chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select excuseRFID, excuseNotes from excuseBase where excuse_eventBase_eventID = "%s" and
excuseChorus = "%s"'%(str(eventID), str(chorus)))
        list = cur.fetchall()
        cur.close()
        conn.close()
        return list
    except Exception, e:
        __log(e)
        return None


def getName(RFID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname from studentBase where RFID = "%s"'%str(RFID))
        name = cur.fetchone()
        cur.close()
        conn.close()
        return name
    except Exception, e:
        __log(e)
        return None


def getTerm():
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select termID from termBase where active = 1')
        term = cur.fetchone()
        cur.close()
        conn.close()
        return term
    except Exception, e:
        __log(e)
        return None


def getTermList():
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select * from termBase order by termID desc')
        list = cur.fetchall()
        cur.close()
        conn.close()
        return list
    except Exception, e:
        __log(e)
        return None
```

```
def listRehearse(chorus, termID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_number, report_date from reportBase where report_chorus=%s and
report_eventBase_eventID in (select eventID from eventBase where event_termBase_termID="%s")'%(str(chorus),
str(termID)))
        list = cur.fetchall()
        cur.close()
        conn.close()
        return list
    except Exception, e:
        __log(e)
        return None


def findRoster(chorus):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, voicePart, gradYear from studentBase where chorus="%s" and RFID not in (select
RFID from standingBase) order by voicePart, lname'%str(chorus))
        rosterList = cur.fetchall()
        cur.close()
        conn.close()
        return rosterList
    except Exception, e:
        __log(e)
        return None


def rehearseRoster(eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, voicePart from studentBase where RFID in (select attend_RFID from attendBase
where attend_eventBase_eventID = "%s") order by voicePart, lname'%str(eventID))
        rosterList = cur.fetchall()
        cur.close()
        conn.close()
        return rosterList
    except Exception, e:
        __log(e)
        return None


def findStanding(chorus, new):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, officer, credit, acgroup from studentBase where RFID in (select RFID from
standingBase where chorus="%s" and isNew="%s") order by lname'%(str(chorus), str(new)))
        list = cur.fetchall()
        cur.close()
        conn.close()
        return list
    except Exception, e:
        __log(e)
        return None
```

```
def getMissedList(chorus, eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, officer, credit, acgroup from studentBase where chorus="%s" and RFID not in (select
attend_RFID from attendBase where attend_eventBase_eventID="%s") order by lname'%(str(chorus), str(eventID)))
        missList = cur.fetchall()
        cur.close()
        conn.close()
        return missList
    except Exception, e:
        __log(e)
        return None


def getAttendList(chorus, eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, officer, credit, acgroup from studentBase where chorus="%s" and RFID in (select
attend_RFID from attendBase where attend_eventBase_eventID="%s") and RFID not in (select excuseRFID from
excuseBase where excuse_eventBase_eventID="%s") order by lname'%(str(chorus), str(eventID), str(eventID)))
        attendList = cur.fetchall()
        cur.close()
        conn.close()
        return attendList
    except Exception, e:
        __log(e)
        return None


def getExcuseList(chorus, eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, officer, credit, acgroup from studentBase where chorus="%s" and RFID in (select
excuseRFID from excuseBase where excuse_eventBase_eventID="%s") order by lname'%(str(chorus), str(eventID)))
        excuseList = cur.fetchall()
        cur.close()
        conn.commit()
        conn.close()
        return excuseList
    except Exception, e:
        __log(e)
        return None


def getLateList(chorus, eventID):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select fname, lname, officer, credit, acgroup from studentBase where chorus="%s" and RFID in (select
attend_RFID from attendBase where attend_eventBase_eventID="%s" and attend_late="1") and RFID not in (select
excuseRFID from excuseBase where excuse_eventBase_eventID="%s") order by lname'%(str(chorus), str(eventID),
str(eventID)))
        attendList = cur.fetchall()
        cur.close()
        conn.close()
        return attendList
    except Exception, e:
        __log(e)
        return None
```

```
def getReport(number):
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('select report_date, report_eventBase_eventID, report_eventBase_eventType from reportBase where
report_number="%s"'%str(number))
        report = cur.fetchone()
        cur.close()
        conn.close()
        return report
    except Exception, e:
        __log(e)
        return None


def reset():
    try:
        conn = __get_conn()
        cur = conn.cursor()
        cur.execute('delete from reportBase')
        cur.execute('delete from excuseBase')
        cur.execute('delete from attendBase')
        cur.execute('delete from eventBase')
        cur.execute('delete from termBase')
        cur.execute('alter table reportBase auto_increment = 1')
        cur.execute('alter table excuseBase auto_increment = 1')
        cur.execute('alter table attendBase auto_increment = 1')
        cur.execute('alter table eventBase auto_increment = 1')
        cur.execute('alter table termBase auto_increment = 1')
        cur.close()
        conn.commit()
        conn.close()
    except Exception, e:
        __log(e)
    return
```

style.css code:

```
.tabhead
{
border-right:1px solid black;
border-left:1px solid black;
border-top:2px solid black;
border-bottom:2px solid black;
}

.tabdat
{
border:1px solid black;
}

.tab
{
border: 3px solid black;
border-collapse: collapse;
width: 100%;
}
```

header code:

```
<!DOCTYPE HTML>
<html>
<head>
<link rel="stylesheet" type="text/css" href="style.css">
<title><!-- TITLE --></title>
```

footer code:

```
<br /><br /><div style="border-top: 1px solid black; width=100%; text-align: center; font-size: 12px;"><br />
<div style="float: right;"><select onchange="nav()" id="url">
<option value="0"> - Quick Navigation - </option>
<option value="index.cgi?">Home</option>
<option value="event/index.cgi?">Events </option>
<option value="student/index.cgi?">Students</option>
<option value="student/index.cgi?page=list&"> - List</option>
<option value="student/index.cgi?page=add&"> - Add</option>
<option value="report/index.cgi?">Reports</option>
<option value="report/index.cgi?type=0&"> - Rehearsal</option>
<option value="report/index.cgi?type=1&"> - Standing</option>
<option value="report/index.cgi?type=2&"> - Roster</option>
<option value="excuse/index.cgi?">Excuse</option>
<option value="excuse/index.cgi?action=list&"> - List</option>
<option value="excuse/index.cgi?action=add&"> - Add</option>
</select>
</div>
This attendance system was created as part of an Interactive Qualifying Project. <br />To report errors or suggestions, please
email <a href="mailto:attendance@wpi.edu">attendance@wpi.edu</a>.
<script type="text/javascript">

var home = "http://users.wpi.edu/~amsmith514/attendance/";
var pw = "pw=[redacted]";

function nav(){
  var url = document.getElementById('url').value;
  if(url!="0"){
    window.location = home+url+pw;
  }
}
</script>
</body></html>
```

Main index.cgi code:

```
#! /usr/bin/python

import cgi
import engine


def printerror():
    try:
        p = open('error.log','r')
        print '</head><body><table class="tab">'
        print '<tr><th class="tabhead">Error Log</th></tr>'
        for i in p.readlines():
            print '<tr class="tabdat"><td>'
            print i
            print '</td></tr>'
        print '</table>'
        p.close()
    except Exception, e:
```

```
        engine.__log(e)


def printadmin():
    p = open('admin.html','r')
    for i in p.readlines():
        print i
    p.close()


def printpw():
    print '</head><body><br /><form align="center" action="index.cgi" method="post">Password: <input type="password"
name="pw" /><input type="submit" value="Submit" /></form></body></html>'


def printmenu():
    p = open('menu.html','r')
    for i in p.readlines():
        print i
    p.close()


def main():
    engine.printheader('Main Menu')
    form = cgi.FieldStorage()
    pw = form.getfirst('pw',None)
    action = form.getfirst('action',None)
    if action == None:
        if pw != '[redacted]':
            printpw()
        else:
            printmenu()
            engine.printfoot()
    elif action == 'reset':
        if pw == '[redacted]':
            engine.reset()
            printmenu()
        else:
            printadmin()
    elif action == 'error':
        if pw == '[redacted]':
            printerror()
        else:
            printadmin()
    elif action == 'admin':
        printadmin()
    if action != None:
        engine.printfoot()


main()
```

Main menu.html code:

```
</head>
<body><br />
<table align="center">
<td><form action="event/index.cgi" method="post">
<input type="hidden" name="pw" value="[redacted]" />
<input type="submit" value="Events" /></form></td>
<td><form action="student/index.cgi" method="post">
```

```
<input type="hidden" name="pw" value="[redacted]" />
<input type="submit" value="Students" /></form></td>
<td><form action="report/index.cgi" method="post">
<input type="hidden" name="pw" value="[redacted]" />
<input type="submit" value="Reports" /></form></td>
<td><form action="excuse/index.cgi" method="post">
<input type="hidden" name="pw" value="[redacted]" />
<input type="submit" value="Excuses" /></form></td>
</tr></table>
```

Events index.cgi code:

```
#! /usr/bin/python

import engine
import cgi


def printnewterm():
    p = open('newterm.html','r')
    for i in p.readlines():
        print i
    p.close()


def printroster(rosterList):
    p = open('roster.html','r')
    for i in p.readlines():
        n = i.find('<!-- ROSTER -->')
        l = 0
        if n != -1:
            print i[0:n]
            for r in rosterList:
                l = l + 1
                print '<tr><td class="tabdat">%s %s</td><td class="tabdat">'%(str(r[0]), str(r[1]))
                if r[2] == 0:
                    print 'Soprano 1'
                elif r[2] == 1:
                    print 'Soprano 2'
                elif r[2] == 2:
                    print 'Alto 1'
                elif r[2] == 3:
                    print 'Alto 2'
                elif r[2] == 4:
                    print 'Tenor 1'
                elif r[2] == 5:
                    print 'Tenor 2'
                elif r[2] == 6:
                    print 'Bass 1'
                elif r[2] == 7:
                    print 'Bass 2'
                elif r[2] == 8:
                    print 'Other'
                print '</td></tr>'
            print '<tr><td class="tabdat">Total:</td><td class="tabdat">%s Students</td></tr>'%str(l)
            print i[n+15:]
        else:
            print i
    p.close()
```

```
def printattend(RFID):
    p = open('attend.html','r')
    for i in p.readlines():
        n = i.find('<!-- STUDENTS -->')
        if n != -1:
            print i[0:n]
            n = engine.getName(RFID)
            if n != None:
                print '<tr><td class="tabdat">%s %s</td><td class="tabdat">%s</td></tr>'%(str(n[0]),str(n[1]),RFID)
            else:
                print '<tr><td class="tabdat">Error</td><td class="tabdat">%s</td></tr>'%str(RFID)
            print i[n+18:]
        print i
    p.close()


def printpw():
    print '</head><body><br /><form align="center" action="index.cgi" method="post">Password: <input type="password"
name="pw" /><input type="submit" value="Submit" /></form>'


def printopenevent(eventID, eventChorus):
    p = open('openevent.html','r')
    for i in p.readlines():
        n = i.find('<!-- EVENTID -->')
        l = i.find('<!-- EVENTCHORUS -->')
        if n != -1:
            print i[0:n]
            print '"%s"'%str(eventID)
            print i[n+16:]
        elif l != -1:
            print i[0:l]
            print '"%s"'%str(eventChorus)
            print i[l+20:]
        else:
            print i
    p.close()


def printevent():
    p = open('newevent.html','r')
    for i in p.readlines():
        print i
    p.close()


def main():
    engine.printheader('Event Mode')
    form = cgi.FieldStorage()
    action = form.getfirst('action',None)
    eventID = form.getfirst('eventID',None)
    eventType = form.getfirst('eventType',None)
    eventChorus = form.getfirst('eventChorus',None)
    RFID = form.getfirst('RFID',None)
    term = form.getfirst('term',None)
    year = form.getfirst('year',None)
    late = form.getfirst('late',None)
    pw = form.getfirst('pw',None)
    if action == None:
        if pw != '[redacted]':
            next = 'Pass'
```

```
            else:
                next = 'Event'
        elif action == 'addEvent' and eventType != None and eventChorus != None:
            eventID = engine.eventCreate(eventType, eventChorus)
            next = 'Open'
        elif action == 'close':
            if eventID == None:
                eventID = engine.findEvent()
            engine.closeEvent(eventID)
            next = 'Roster'
        elif action == 'newTerm':
            if term == None or year == None:
                next = 'Term'
            else:
                engine.newTerm(term, year)
                next = 'Event'
        elif action == 'attend':
            if eventID == None:
                eventID = engine.findEvent()
            if RFID != None:
                if late == '1':
                    engine.addAttend(RFID, eventID, eventChorus, 1)
                else:
                    engine.addAttend(RFID, eventID, eventChorus, 0)
            next = 'Attend'
        if next == 'Open':
            printopenevent(eventID, eventChorus)
            engine.printfoot()
        elif next == 'Term':
            printnewterm()
            engine.printfoot()
        elif next == 'Attend':
            printattend(RFID)
        elif next == 'Event':
            printevent()
            engine.printfoot()
        elif next == 'Roster':
            rosterList = engine.rehearseRoster(eventID)
            printroster(rosterList)
            engine.printfoot()
        elif next == 'Pass':
            printpw()


    main()
```

newevent.html code:

```
    </head>
    <body><br />
    <table align="center">
     <form action="index.cgi" method="post">
     <tr><th>Event Type: </th>
      <td><select name="eventType">
       <option value="0">Rehearsal</option>
       <option value="1">Concert</option>
       <option value="2">Makeup Rehearsal</option>
       <option value="3">Optional Rehearsal</option>
      </select></td>
     <th>Chorus:</th>
```

```
 <td><select name="eventChorus">
  <option value="0">Glee Club</option>
  <option value="1">Alden Voices</option>
  <option value="2">Festival Chorus</option>
 </select></td></tr></table>
 <table align="center"><tr><td><input type="hidden" name="action" value="addEvent" />
 <input type="submit" value="Create Event"></td>
 </form>
<td><form action="index.cgi" method="post"><input type="hidden" name="action" value="newTerm" /><input
type="submit" value="New Semester" /></form></td><td><form action="../index.cgi" method="post"><input
type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Main Menu" /></form></td></tr>
</table>
```

openevent.html code:

```
<script type="text/javascript">

window.onload = function() {
 document.getElementById("RFID").focus();
}

function makeLate(){
 document.getElementById('late').value = "1";
}

function closeEvent(){
 var r = confirm("Are you sure you want to close this event?");
 return r;
}

function addAttend(){
 var RFID = document.getElementById('RFID').value;
 var eventID = document.getElementById('eventID').value;
 var eventChorus = document.getElementById('eventChorus').value;
 var late = document.getElementById('late').value;
 var nt = new RegExp('[^0-9]');
 if(!(nt.test(RFID)) && RFID.length > 3){

document.getElementById('list').src='index.cgi?action=attend&eventID='+eventID+'&RFID='+RFID+'&eventChorus='+eve
ntChorus+'&late='+late;
 }
 document.getElementById('RFID').value='';
 document.getElementById('RFID').focus();
}

 </script>
</head>
<body><br /><table align="center">
<tr><td>
<form action="index.cgi" method="post" onsubmit="return closeEvent()">
<input type="hidden" name="eventID" id="eventID" value=<!-- EVENTID --> />
<input type="hidden" name="eventChorus" id="eventChorus" value=<!-- EVENTCHORUS --> />
<input type="hidden" name="action" value="close" />
<input type="hidden" name="pw" value="[redacted]" />
<input type="submit" value="Close Event" />
</form></td><td>
RFID: </td>
 <td><input type="text" id="RFID" name="RFID" value="" maxlength="5" onkeydown="if (event.keyCode == 13)
{document.getElementById('btn').click()}" /></td>
 <td><button type="button" id="btn" onclick="addAttend()">Submit</button></td><td><input type="button"
value="Late" onclick="makeLate()" /><input type="hidden" id="late" value="0" /></td></tr></table>
```

```
<p align="center">If your name does not appear below, please contact your secretary.</p>
 <div align="center"><iframe width="300" id="list" seamless src="index.cgi?action=attend"></iframe></div>
```

attend.html code:

```
</head>
<body>
 <table class="tab" align="center">
  <tr><th class="tabhead" style="width: 80%;">Name</th><th class="tabhead" style="width: 20%;">RFID</th></tr>
  <!-- STUDENTS -->
 </table>
</body></html>
```

roster.html code:

```
</head>
<body>
<table class="tab" align="center" style="width:50%">
<tr><th class="tabhead" width="50%">Name</th><th class="tabhead" width="50%">Voice Part</th></tr>
<!-- ROSTER -->
<tr><td></td><td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Event Menu" /></form></td></tr></table>
```

Student index.cgi code:

```
#! /usr/bin/python

import engine
import cgi


def printmenu():
    p = open('studentmenu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printdetail(RFID, chorus):
    attend = engine.attendEvents(RFID, chorus)
    late = engine.lateEvents(RFID, chorus)
    miss = engine.missEvents(RFID, chorus)
    excuse = engine.excuseEvents(RFID, chorus)
    name = engine.getName(RFID)
    fname = name[0]
    lname = name[1]
    p = open('detail.html','r')
    for i in p.readlines():
        l = i.find('<!-- LATE -->')
        n = i.find('<!-- NAME -->')
        a = i.find('<!-- ATTEND -->')
        m = i.find('<!-- MISSED -->')
        e = i.find('<!-- EXCUSE -->')
        c = i.find('<!-- CHORUS -->')
        if c != -1:
            print i[0:c]+'%s'%str(chorus)+i[c+15:]
        elif a != -1:
            print i[0:a]
            for r in attend:
                print '<tr><td class="tabdat">%s</td></tr>'%str(r[0])
```

```
            print i[a+15:]
        elif m != -1:
            print i[0:m]
            for r in miss:
                print '<tr><td class="tabdat">%s</td></tr>'%str(r[0])
            print i[m+15:]
        elif e != -1:
            print i[0:e]
            for r in excuse:
                print '<tr><td class="tabdat">%s</td></tr>'%str(r[0])
            print i[e+15:]
        elif l != -1:
            print i[0:l]
            for r in late:
                print '<tr><td class="tabdat">%s</td></tr>'%str(r[0])
            print i[l+13:]
        elif n != -1:
            print i[0:n]
            print '%s %s'%(str(fname), str(lname))
            print i[n+13:]
        else:
            print i
    p.close



def printlistchorus():
    p = open('listchorus.html','r')
    for i in p.readlines():
        print i
    p.close()


def printadd():
    p = open('studentadd.html','r')
    for i in p.readlines():
        print i
    p.close()


def printpw():
    print '</head><body><br /><form align="center" action="index.cgi" method="post">Password: <input type="password"
name="pw" /><input type="submit" value="Submit" /></form></body></html>'


def printstudentlist(chorus):
    p = open('studentlist.html','r')
    for i in p.readlines():
        n = i.find('<!-- STUDENTS -->')
        if n != -1:
            print i[0:n]
            s = engine.listStudents(chorus)
            r = 0;
            if s != None:
                for o in s:
                    r = r + 1
                    print '<tr><form action="index.cgi" method="post" onsubmit="return save(this)">'
                    print '<td class="tabdat"><input type="hidden" name="fname" value="%s" />%s %s<input type="hidden"
name="lname" value="%s" /></td>'%(str(o[0]), str(o[0]), str(o[1]), str(o[1]))
                    print '<td class="tabdat"><input type="hidden" name="studentID" value="%s" />%s</td>'%(str(o[2]),
str(o[2]))
```

```
            print '<td class="tabdat"><input type="text" size="3" maxlength="6" name="RFID" value="%s"
/></td>'%str(o[3])
            print '<td class="tabdat"><select name="officer">'
            if o[4] == 0:
               print '<option value="1">Yes</option><option selected value="0">No</option>'
            elif o[4] == 1:
               print '<option selected value="1">Yes</option><option value="0">No</option>'
            print '</select></td><td class="tabdat"><select name="acgroup">'
            if o[5] == 0:
               if chorus == '0':
                  print '<option selected value="0">None</option><option value="1">Simple Harmonic
Motion</option><option value="3">Audiophiles</option><option value="4">Simple Harmonic Motion and
Audiophiles</option>'
               elif chorus == '1':
                  print '<option selected value="0">None</option><option value="2">Technichords</option><option
value="3">Audiophiles</option><option value="5">Technichords and Audiophiles</option>'
            elif o[5] == 1:
               print '<option value="0">None</option><option selected value="1">Simple Harmonic
Motion</option><option value="3">Audiophiles</option><option value="4">Simple Harmonic Motion and
Audiophiles</option>'
            elif o[5] == 2:
               print '<option value="0">None</option><option selected value="2">Technichords</option><option
value="3">Audiophiles</option><option value="5">Technichords and Audiophiles</option>'
            elif o[5] == 3:
               if chorus == '0':
                  print '<option value="0">None</option><option value="1">Simple Harmonic Motion</option><option
selected value="3">Audiophiles</option><option value="4">Simple Harmonic Motion and Audiophiles</option>'
               elif chorus == '1':
                  print '<option value="0">None</option><option value="2">Technichords</option><option selected
value="3">Audiophiles</option><option value="5">Technichords and Audiophiles</option>'
            elif o[5] == 4:
               print '<option value="0">None</option><option value="1">Simple Harmonic Motion</option><option
value="3">Audiophiles</option><option selected value="4">Simple Harmonic Motion and Audiophiles</option>'
            elif o[5] == 5:
               print '<option value="0">None</option><option value="2">Technichords</option><option
value="3">Audiophiles</option><option selected value="5">Technichords and Audiophiles</option>'
            print '</select></td><td class="tabdat"><select name="credit">'
            if o[6] == 0:
               print '<option value="1">Yes</option><option selected value="0">No</option>'
            elif o[6] == 1:
               print '<option selected value="1">Yes</option><option value="0">No</option>'
            print '</select></td><td class="tabdat"><select name="voicePart">'
            if o[7] == 0:
               print '<option selected value="0">Soprano 1</option><option value="1">Soprano 2</option><option
value="2">Alto 1</option><option value="3">Alto 2</option><option value="8">Other</option>'
            elif o[7] == 1:
               print '<option value="0">Soprano 1</option><option selected value="1">Soprano 2</option><option
value="2">Alto 1</option><option value="3">Alto 2</option><option value="8">Other</option>'
            elif o[7] == 2:
               print '<option value="0">Soprano 1</option><option value="1">Soprano 2</option><option selected
value="2">Alto 1</option><option value="3">Alto 2</option><option value="8">Other</option>'
            elif o[7] == 3:
               print '<option value="0">Soprano 1</option><option value="1">Soprano 2</option><option
value="2">Alto 1</option><option selected value="3">Alto 2</option><option value="8">Other</option>'
            elif o[7] == 4:
               print '<option selected value="4">Tenor 1</option><option value="5">Tenor 2</option><option
value="6">Bass 1</option><option value="7">Bass 2</option><option value="8">Other</option>'
            elif o[7] == 5:
               print '<option value="4">Tenor 1</option><option selected value="5">Tenor 2</option><option
value="6">Bass 1</option><option value="7">Bass 2</option><option value="8">Other</option>'
            elif o[7] == 6:
```

```
            print '<option value="4">Tenor 1</option><option value="5">Tenor 2</option><option selected
value="6">Bass 1</option><option value="7">Bass 2</option><option value="8">Other</option>'
          elif o[7] == 7:
            print '<option value="4">Tenor 1</option><option value="5">Tenor 2</option><option value="6">Bass
1</option><option selected value="7">Bass 2</option><option value="8">Other</option>'
          elif o[7] == 8:
            if chorus == '0':
              print '<option value="4">Tenor 1</option><option value="5">Tenor 2</option><option value="6">Bass
1</option><option value="7">Bass 2</option><option selected value="8">Other</option>'
            elif chorus == '1':
              print '<option value="0">Soprano 1</option><option value="1">Soprano 2</option><option
value="2">Alto 1</option><option value="3">Alto 2</option><option selected value="8">Other</option>'
          print '</select></td>'
          min = o[8] - 1;
          max = o[8] + 1;
          mid = o[8];
          print '<td class="tabdat"><select name="gradYear"><option value="%s">%s</option><option selected
value="%s">%s</option><option value="%s">%s</option></select></td>'%(str(min), str(min), str(mid), str(mid),
str(max), str(max))
          print'<input type="hidden" name="chorus" value="%s" />'%str(o[12])
          if o[12] == 0:
            print '<td class="tabdat">Glee Club</td>'
          elif o[12] == 1:
            print '<td class="tabdat">Alden Voices</td>'
          print '<td class="tabdat">%s</td>'%str(o[9])
          print '<td class="tabdat">%s</td>'%str(o[10])
          print '<td class="tabdat">%s</td>'%str(o[11])
          print '<td class="tabdat">%s</td>'%str(o[13])
          print '<td class="tabdat"><input type="hidden" name="page" value="list" /><input type="hidden"
name="action" value="update" /><input type="submit" value="Save"></form></td>'
          print '<td class="tabdat"><form action="index.cgi" method="post"><input type="hidden" name="RFID"
value="%s" /><input type="hidden" name="page" value="detail" /><input type="hidden" name="chorus" value="%s"
/><input type="submit" value="Detail" /></form></td>'%(str(o[3]), str(o[12]))
          print '<td class="tabdat"><form action="index.cgi" method="post" onsubmit="return del(this)"><input
type="hidden" name="fname" value="%s" /><input type="hidden" name="lname" value="%s" /><input type="hidden"
name="page" value="list" /><input type="hidden" name="RFID" value="%s" /><input type="hidden" name="action"
value="del" /><input type="hidden" name="chorus" value="%s" /><input type="submit" value="Delete"
/></form></td>'%(str(o[0]), str(o[1]), str(o[3]), str(o[12]))
          print '</tr>'
      print '<tr><td class="tabdat">Total: %s
Student(s)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></
td><td></td><td></td><td></td></tr>'%str(r)
      print i[n+17:]
    print i
  p.close()


def main():
  engine.printheader('Student Mode')
  form = cgi.FieldStorage()
  action = form.getfirst('action',None)
  fname = form.getfirst('fname',None)
  lname = form.getfirst('lname',None)
  studentID = form.getfirst('studentID',None)
  RFID = form.getfirst('RFID',None)
  officer = form.getfirst('officer',None)
  acgroup = form.getfirst('acgroup',None)
  credit = form.getfirst('credit',None)
  voicePart = form.getfirst('voicePart',None)
  gradYear = form.getfirst('gradYear',None)
  chorus = form.getfirst('chorus',None)
  page = form.getfirst('page',None)
```

```
    pw = form.getfirst('pw',None)
    if page == None:
        if pw != '[redacted]':
            printpw()
        else:
            printmenu()
            engine.printfoot()
    if action == 'add':
        engine.addStudent(fname, lname, studentID, RFID, officer, acgroup, credit, voicePart, gradYear, chorus)
    elif action == 'del':
        engine.delStudent(RFID)
    elif action == 'update':
        engine.updateStudent(fname, lname, RFID, officer, acgroup, credit, voicePart, gradYear)
    if page == 'list':
        if chorus != None:
            printstudentlist(chorus)
        else:
            printlistchorus()
    elif page == 'add':
        printadd()
    elif page == 'detail':
        printdetail(RFID, chorus)
    if page != None:
        engine.printfoot()


main()
```

studentmenu.html code:

```
</head>
<body><br />
<table align="center">
<tr>
<td>
<form action="index.cgi" method="post">
 <input type="hidden" name="page" value="list" />
 <input type="submit" value="List Students" />
 </form></td>
<td>
<form action="index.cgi" method="post">
 <input type="hidden" name="page" value="add" />
 <input type="submit" value="Add Student" />
 </form></td>
<td>
<form action="../index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit"
value="Main Menu" /></form>
</td></tr></table>
```

studentadd.html code:

```
<script type="text/javascript">


function checkAdd(){
  var error = '';
  var fname = document.getElementsByName("fname")[0].value;
  var lname = document.getElementsByName("lname")[0].value;
  var studentID = document.getElementsByName("studentID")[0].value;
  var RFID = document.getElementsByName("RFID")[0].value;
  var chorus = document.getElementsByName("chorus")[0].value;
```

```
 var voicePart = document.getElementsByName("voicePart")[0].value;
 var acgroup = document.getElementsByName("acgroup")[0].value;
 var nt = new RegExp('[^A-z]');
 var numt = new RegExp('[^0-9]');
 if(nt.test(fname) || fname == ''){
  error = error + 'First Name Invalid.\n';
 }
 if(nt.test(lname) || lname == ''){
  error = error + 'Last Name Invalid.\n';
 }
 if(numt.test(studentID) || !(studentID.length > 8)){
  error = error + 'Student ID Invalid.\n';
 }
 if(numt.test(RFID) || !(RFID.length > 3)){
  error = error + 'RFID Invalid.\n';
 }
 if(chorus == 'null'){
  error = error + 'Chorus Invalid.\n';
 }else if(chorus == '0'){
  if(!(voicePart == '4' || voicePart == '5' || voicePart == '6' || voicePart == '7' || voicePart == '8')){
   error = error + 'Voice Part Invalid.\n';
  }if(!(acgroup == '0' || acgroup == '1' || acgroup == '3' || acgroup == '4')){
   error = error + 'Group Invalid.\n';
  }
 }else if(chorus == '1'){
  if(!(voicePart == '0' || voicePart == '1' || voicePart == '2' || voicePart == '3' || voicePart == '8')){
   error = error + 'Voice Part Invalid.\n';
  }if(!(acgroup == '0' || acgroup == '2' || acgroup == '3' || acgroup == '5')){
   error = error + 'Group Invalid.\n';
  }
 }
 if (error != ''){
  alert(error);
  return false;
 }else{
  return true;
 }
}


</script>
</head>
 <body>
 <table align="center">
 <form action="index.cgi" method="post" onsubmit="return checkAdd()">
  <tr><td>First Name:</td><td><input maxlength="25" type="text" name="fname" /></td></tr>
  <tr><td>Last Name:</td><td><input maxlength="25" type="text" name="lname" /></td></tr>
  <tr><td>Student ID:</td><td><input maxlength="9" type="text" name="studentID" /></td></tr>
  <tr><td>RFID:</td><td><input maxlength="5" type="text" name="RFID" onkeydown="if (event.keyCode == 13)
{return false}"/></td></tr>
  <tr><td>Officer:</td>
  <td>
   <select name="officer">
    <option value="1">Yes</option>
    <option value="0">No</option>
   </select>
  </td></tr>
  <tr><td>Credit:</td>
  <td>
   <select name="credit">
    <option value="1">Yes</option>
    <option value="0">No</option>
```

```
   </select>
  </td></tr>
 <tr><td>Chorus</td>
  <td><select name="chorus">
   <option value="null"> - Select - </option>
   <option value="0">Glee Club</option>
   <option value="1">Alden Voices</option>
  </select></td></tr>
   <tr><td>Voice Part:</td>
   <td>
    <select name="voicePart">
     <option value="null"> - Select - </option>
     <option value="0">Soprano 1</option>
     <option value="1">Soprano 2</option>
     <option value="2">Alto 1</option>
     <option value="3">Alto 2</option>
     <option value="4">Tenor 1</option>
     <option value="5">Tenor 2</option>
     <option value="6">Bass 1</option>
     <option value="7">Bass 2</option>
     <option value="8">Other</option>
    </select>
   </td></tr>
   <tr><td>Group:</td>
   <td>
    <select name="acgroup">
<option value="0">None</option>
<option value="1">Simple Harmonic Motion</option>
<option value="2">Technichords</option>
<option value="3">Audiophiles</option>
<option value="4">Simple Harmonic Motion and Audiophiles</option>
<option value="5">Technichords and Audiophiles</option>
    </select>
   </td></tr>
   <tr><td>Graduation Year:</td><td><select name="gradYear">
<option value="2012">2012</option>
<option value="2013">2013</option>
<option value="2014">2014</option>
<option value="2015">2015</option>
<option value="2016">2016</option>
<option value="2017">2017</option>
</td></tr>
   <tr><td>Add Another Student:</td><td><input type="checkbox" name="page" value="add" />
   <tr><td style="text-align: right;"><input type="hidden" name="action" value="add" /><input type="hidden" name="pw"
value="[redacted]" /><input type="submit" value="Add Student" /></td></form><td><form action="index.cgi"
method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Back"
/></form></td></tr>
 </table>
```

listchorus.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="page" value="list" /><input type="hidden"
name="chorus" value="0" /><input type="submit" value="Glee Club" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="page" value="list" /><input type="hidden"
name="chorus" value="1" /><input type="submit" value="Alden Voices" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Back" /></form></td></tr></table>
```

studentlist.html code:

```
<script type="text/javascript">
function del(form){
  var fname = form.elements["fname"].value;
  var lname = form.elements["lname"].value;
  var msg = "Are you sure you want to delete " + fname + " " + lname + "?";
  r = confirm(msg);
  return r;
}

function save(form){
  var RFID = form.elements["RFID"].value;
  var nt = new RegExp('[^0-9]');
  if (nt.test(RFID) || RFID.length < 4){
    alert('Invalid RFID.')
    return false;
  }else{
    return true;
  }
}
</script>
 </head>
 <body>

  <table class="tab">
   <tr>
    <th class="tabhead">
     Name
    </th>
    <th class="tabhead">
     Student ID
    </th>
    <th class="tabhead">
     RFID
    </th>
    <th class="tabhead">
     Officer
    </th>
    <th class="tabhead">
     Group
    </th>
    <th class="tabhead">
     Credit
    </th>
    <th class="tabhead">
     Voice Part
    </th>
    <th class="tabhead">
     Graduation Year
    </th>
    <th class="tabhead">Chorus</th>
    <th class="tabhead">
     Attend
    </th>    <th class="tabhead">
     Miss
    </th>    <th class="tabhead">
     Excuse
    </th>
    </th>    <th class="tabhead">
     Late
```

```
    </th>
    <td style="border-bottom: 2px solid black;"></td><td style="border-bottom: 2px solid black;"></td><td style="border-bottom: 2px solid black;"></td>
   </tr>
   <!-- STUDENTS -->
  </table>
<br />
  <table><tr><td><form action="index.cgi" method="post"><input type="hidden" name="page" value="list" /><input type="submit" value="Back" /></form></td><td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Student Menu" /></form></td></tr></table>
```

detail.html code:

```
</head>
<body>
 <table align="center">
  <tr><th>Name:</th><td><!-- NAME --></td></tr></table>
  <table style="border: 3px solid black; border-collapse: collapse; width: 100px;" align="center">
<tr><th class="tabhead">Attended</th></tr>
  <!-- ATTEND -->
<tr><th class="tabhead">Absent</th></tr>
  <!-- MISSED -->
<tr><th class="tabhead">Late</th></tr>
  <!-- LATE -->
<tr><th class="tabhead">Excused</th></tr>
  <!-- EXCUSE -->
</table><br />
<table align="center"><tr><td><form action="index.cgi" method="post">
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="hidden" name="page" value="list" />
<input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Students Menu" /></form></td></tr></table>
```

Excuse index.cgi code:

```
#! /usr/bin/python

import engine
import cgi


def printmenu():
    p = open('menu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printchoral():
    p = open('choral.html','r')
    for i in p.readlines():
        print i
    p.close()


def printpw():
    print '</head><body><br /><form align="center" action="index.cgi" method="post">Password: <input type="password" name="pw" /><input type="submit" value="Submit" /></form></body></html>'
```

```
def printtermlist(list, chorus):
    p = open('termlist.html','r')
    for i in p.readlines():
        l = i.find('<!-- LIST -->')
        c = i.find('<!-- CHORUS -->')
        if l != -1:
            print i[0:l]
            for r in list:
                print '<option value="%s">%s %s</option>'%(str(r[0]), str(r[2]), str(r[1]))
            print i[l+13:]
        elif c != -1:
            print i[0:c]+chorus+i[c+15:]
        else:
            print i
    p.close()


def printlist(excuses, chorus, termID):
    p = open('list.html','r')
    for i in p.readlines():
        n = i.find('<!-- EXCUSES -->')
        t = i.find('<!-- TERMID -->')
        c = i.find('<!-- CHORUS -->')
        if n != -1:
            print i[0:n]
            for r in excuses:
                name = engine.getName(r[0])
                print '<tr><td class="tabdat">%s %s</td>'%(str(name[0]), str(name[1]))
                print '<td class="tabdat">%s</td></tr>'%str(r[1])
            print i[n+16:]
        elif t != -1:
            print i[0:t]+termID+i[t+15:]
        elif c != -1:
            print i[0:c]+chorus+i[c+15:]
        else:
            print i
    p.close()


def printlistmenu(list, chorus, termID):
    p = open('listmenu.html','r')
    for i in p.readlines():
        g = i.find('<!-- LIST -->')
        c = i.find('<!-- CHORUS -->')
        t = i.find('<!-- TERMID -->')
        if g != -1:
            print i[0:g]
            for n in list:
                print '<option value="%s">%s</option>'%(str(n[1]), str(n[4]))
            print i[g+13:]
        elif c != -1:
            print i[0:c]+chorus+i[c+15:]
        elif t != -1:
            print i[0:t]+termID+i[t+15:]
        else:
            print i
    p.close()


def printadd(reports, students, chorus):
    p = open('add.html','r')
```

```
    for i in p.readlines():
        n = i.find('<!-- EVENTS -->')
        s = i.find('<!-- STUDENTS -->')
        c = i.find('<!-- CHORUS -->')
        if s != -1:
            print i[0:s]
            for r in students:
                print '<option value="%s">%s, %s</option>'%(str(r[3]), str(r[1]), str(r[0]))
            print i[s+17:]
        elif n != -1:
            print i[0:n]
            for r in reports:
                print '<option value="%s">%s</option>'%(str(r[1]), str(r[4]))
            print i[n+15:]
        elif c != -1:
            print i[0:c]+str(chorus)+i[c+15:]
        else:
            print i
    p.close()


def printaddmenu():
    p = open('addmenu.html','r')
    for i in p.readlines():
        print i
    p.close()


def main():
    engine.printheader('Excuse Mode')
    form = cgi.FieldStorage()
    action = form.getfirst('action',None)
    eventID = form.getfirst('eventID',None)
    RFID = form.getfirst('RFID',None)
    excuse = form.getfirst('excuse',None)
    chorus = form.getfirst('chorus',None)
    termID = form.getfirst('termID',None)
    more = form.getfirst('more',None)
    pw = form.getfirst('pw',None)
    if action == None:
        if pw != '[redacted]':
            printpw()
        else:
            printmenu()
            engine.printfoot()
    elif action == 'add':
        if eventID == None:
            if chorus != None:
                termID = engine.findcurterm()
                reports = engine.listReports(chorus, termID)
                students = engine.listStudents(chorus)
                printadd(reports, students, chorus)
            elif chorus == None:
                printaddmenu()
        elif eventID != None and RFID != None:
            engine.addexcuse(eventID, RFID, excuse, chorus)
            if more != None:
                termID = engine.findcurterm()
                reports = engine.listReports(chorus, termID)
                students = engine.listStudents(chorus)
                printadd(reports, students, chorus)
            else:
```

```
            printmenu()
    elif action == 'list':
        if eventID == None:
            if chorus != None:
                if termID != None:
                    list = engine.listReports(chorus, termID)
                    printlistmenu(list, chorus, termID)
                else:
                    list = engine.getTermList()
                    printtermlist(list, chorus)
            else:
                printchoral()
        else:
            excuses = engine.listExcuses(eventID, chorus)
            printlist(excuses, chorus, termID)
    if action != None:
        engine.printfoot()


main()
```

Excuse menu.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post">
<input type="hidden" name="action" value="list" />
<input type="submit" value="List Excuses" />
</form></td>
<td><form action="index.cgi" method="post">
<input type="hidden" name="action" value="add" />
<input type="submit" value="Add Excuse" />
</form></td>
<td><form action="../index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Main Menu" /></form></td></tr></table>
```

Excuse listemenu.html code:

```
</head>
<body><br /><table align="center"><form action="index.cgi" method="post">
<tr><td><select name="eventID"><!-- LIST --></select</td>
<td><input type="hidden" name="action" value="list" />
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="hidden" name="termID" value="<!-- TERMID -->" />
<input type="submit" value="Select Rehearsal" /></td></form>
<td><form action="index.cgi" method="post"><input type="hidden" name="action" value="list" /><input type="hidden"
name="chorus" value="<!-- CHORUS -->" /><input type="submit" value="Back" /></form></td>
<td><td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Excuse Menu" /></form></td></tr></table>
```

Excuse list.html code:

```
</head>
 <body>
  <table class="tab">
   <tr>
    <th class="tabhead" style="width:30%;">
     Name
    </th>
```

```
   <th class="tabhead" style="width:70%;">
    Excuse
   </th>
   </tr>
  <!-- EXCUSES -->
 </table>
<br />
<table><tr><td><form action="index.cgi" method="post">
<input type="hidden" name="action" value="list" />
<input type="hidden" name="termID" value="<!-- TERMID -->" />
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Excuse Menu" /></form></td></tr></table>
```

Excuse termlist.html code:

```
</head>
<body><br /><table align="center">
<td><form action="index.cgi" method="post">
<select name="termID"><!-- LIST --></select>
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="hidden" name="action" value="list" /></td>
<td><input type="submit" value="Show Excuses" /></form></td><td><form action="index.cgi" method="post"><input
type="hidden" name="action" value="list" />
<input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Excuse Menu" /></form></td></tr></table>
```

Excuse choral.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="action" value="list" /><input
type="hidden" name="chorus" value="0" /><input type="submit" value="Glee Club" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="action" value="list" /><input type="hidden"
name="chorus" value="1" /><input type="submit" value="Alden Voices" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Excuse Menu" /></form></td></tr></table>
```

Excuse addmenu.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post">
<input type="hidden" name="chorus" value="0" />
<input type="hidden" name="action" value="add" />
<input type="submit" value="Glee Club" />
</form></td>
<td><form action="index.cgi" method="post">
<input type="hidden" name="chorus" value="1" />
<input type="hidden" name="action" value="add" />
<input type="submit" value="Alden Voices" />
</form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Back" /></form></td></tr></table>
```

Excuse add.html code:

```
</head>
<body>
<table align="center">
<form action="index.cgi" method="post">
<tr><td>Name: </td><td><select name="RFID"><option>--</option><!-- STUDENTS --></select></td></tr>
<tr><td>Rehearsal: </td><td><select name="eventID"><option>--</option><!-- EVENTS --></select></td></tr>
<tr><td>Excuse:</td><td><input type="text" name="excuse" /></td></tr>
<input type="hidden" name="action" value="add" />
<input type="hidden" name="chorus" value="<!-- CHORUS -->" /><br />
<tr><td>Add Another Excuse: </td><td><input type="checkbox" name="more" value="1" /></td></tr></table>
<table align="center">
<tr><td style="text-align: right;"><input type="submit" value="Add Excuse" /></td><td>
</form>
<form action="index.cgi" method="post">
<input type="hidden" name="action" value="add" />
<input type="submit" value="Back" /></form></td><td><form action="index.cgi" method="post"><input type="hidden"
name="pw" value="[redacted]" /><input type="submit" value="Excuse Menu" /></form></td></tr></table>
```

Report index.cgi code:

```
#! /usr/bin/python

import engine
import cgi


def printstandingmenu():
    p = open('standingmenu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printrostermenu():
    p = open('rostermenu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printpw():
    print '</head><body><br /><form align="center" action="index.cgi" method="post">Password: <input type="password"
name="pw" /><input type="submit" value="Submit" /></form></body></html>'


def printtermlist(list, chorus):
    p = open('termlist.html','r')
    for i in p.readlines():
        l = i.find('<!-- LIST -->')
        c = i.find('<!-- CHORUS -->')
        if l != -1:
            print i[0:l]
            for r in list:
                print '<option value="%s">%s %s</option>'%(str(r[0]), str(r[2]), str(r[1]))
            print i[l+13:]
        elif c != -1:
            print i[0:c]+chorus+i[c+15:]
        else:
```

```
            print i
    p.close()


def printStanding(list, new):
    for r in list:
        print '<tr><td class="tabdat">%s, %s'%(str(r[1]), str(r[0]))
        if new == '1':
            print '*'
        print '</td><td class="tabdat">'
        if r[4] == 0:
            print 'None</td>'
        elif r[4] == 1:
            print 'Simple Harmonic Motion</td>'
        elif r[4] == 2:
            print 'Technichords</td>'
        elif r[4] == 3:
            print 'Audiophiles</td>'
        elif r[4] == 4:
            print 'Simple Harmonic Motion, Audiophiles</td>'
        elif r[4] == 5:
            print 'Technichords, Audiophiles</td>'
        print '<td class="tabdat">'
        if r[3] == 0:
            print 'No</td>'
        elif r[3] == 1:
            print 'Yes</td>'
        print '<td class="tabdat">'
        if r[2] == 0:
            print 'No</td></tr>'
        elif r[2] == 1:
            print 'Yes</td></tr>'


def standingReport(chorus):
    newList = engine.findStanding(chorus, '1')
    oldList = engine.findStanding(chorus, '0')
    s = open('standing.html','r')
    for i in s.readlines():
        n = i.find('<!-- STANDING -->')
        if n != -1:
            print i[0:n]
            printStanding(newList, '1')
            printStanding(oldList, '0')
            print i[n+17:]
        else:
            print i
    s.close()


def studentList(list):
    for r in list:
        print '<tr><td class="tabdat">%s, %s</td><td class="tabdat">'%(str(r[1]), str(r[0]))
        if r[4] == 0:
            print 'None</td>'
        elif r[4] == 1:
            print 'Simple Harmonic Motion</td>'
        elif r[4] == 2:
            print 'Technichords</td>'
        elif r[4] == 3:
            print 'Audiophiles</td>'
        elif r[4] == 4:
```

```
            print 'Simple Harmonic Motion, Audiophiles</td>'
        elif r[4] == 5:
            print 'Technichords, Audiophiles</td>'
        print '<td style="border: 1px solid black">'
        if r[3] == 0:
            print 'No</td>'
        elif r[3] == 1:
            print 'Yes</td>'
        print '<td class="tabdat">'
        if r[2] == 0:
            print 'No</td></tr>'
        elif r[2] == 1:
            print 'Yes</td></tr>'


def rehearsalReport(number, chorus, termID):
    report =  engine.getReport(number)
    today = report[0]
    eventID = report[1]
    eventType = report[2]
    attendList = engine.getAttendList(chorus, eventID)
    missList = engine.getMissedList(chorus, eventID)
    excuseList = engine.getExcuseList(chorus, eventID)
    lateList = engine.getLateList(chorus, eventID)
    if eventType == 0:
        eventType = 'Rehearsal'
    elif eventType == 1:
        eventType = 'Concert'
    elif eventType == 2:
        eventType = 'Make-up Rehearsal'
    elif eventType == 3:
        eventType = 'Optional Rehearsal'
    if chorus == '0':
        verbose = 'Glee Club'
    elif chorus == '1':
        verbose = 'Alden Voices'
    temp = open('rehearsal.html','r')
    for i in temp.readlines():
        t = i.find('<!-- EVENTTYPE -->')
        g = i.find('<!-- GROUP -->')
        d = i.find('<!-- DATE -->')
        l = i.find('<!-- LATE -->')
        a = i.find('<!-- ATTEND -->')
        m = i.find('<!-- MISSED -->')
        e = i.find('<!-- EXCUSE -->')
        c = i.find('<!-- CHORUS -->')
        z = i.find('<!-- TERMID -->')
        if t != -1:
            print i[0:t]+eventType+i[t+18:]
        elif z != -1:
            print i[0:z]+termID+i[z+15:]
        elif c != -1:
            print i[0:c]+'%s'%str(chorus)+i[c+15:]
        elif g != -1:
            print i[0:g]+verbose+i[g+14:]
        elif d != -1:
            print i[0:d]
            print today
            print i[d+13:]
        elif a != -1:
            print i[0:a]
            studentList(attendList)
```

```
            print i[a+15:]
        elif m != -1:
            print i[0:m]
            studentList(missList)
            print i[m+15:]
        elif e != -1:
            print i[0:e]
            studentList(excuseList)
            print i[e+15:]
        elif l != -1:
            print i[0:l]
            studentList(lateList)
            print i[l+13:]
        else:
            print i
        temp.close()


def printroster(rosterList):
    p = open('roster.html','r')
    for i in p.readlines():
        n = i.find('<!-- ROSTER -->')
        l = 0
        if n != -1:
            print i[0:n]
            for r in rosterList:
                l = l + 1
                print '<tr><td class="tabdat">%s %s</td><td class="tabdat">'%(str(r[0]), str(r[1]))
                if r[2] == 0:
                    print 'Soprano 1'
                elif r[2] == 1:
                    print 'Soprano 2'
                elif r[2] == 2:
                    print 'Alto 1'
                elif r[2] == 3:
                    print 'Alto 2'
                elif r[2] == 4:
                    print 'Tenor 1'
                elif r[2] == 5:
                    print 'Tenor 2'
                elif r[2] == 6:
                    print 'Bass 1'
                elif r[2] == 7:
                    print 'Bass 2'
                elif r[2] == 8:
                    print 'Other'
                print '</td><td class="tabdat">%s</td></tr>'%str(r[3])
            print '<tr><td class="tabdat">Total:</td><td class="tabdat">%s Students</td><td></td></tr>'%str(l)
            print i[n+15:]
        else:
            print i
    p.close()


def printmenu():
    p = open('menu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printrehearsemenu():
```

```
    p = open('rehearsalmenu.html','r')
    for i in p.readlines():
        print i
    p.close()


def printrehearse(list, chorus, termID):
    p = open('rehearse.html','r')
    for i in p.readlines():
        g = i.find('<!-- LIST -->')
        t = i.find('<!-- TERMID -->')
        c = i.find('<!-- CHORUS -->')
        if c != -1:
            print i[0:c]+chorus+i[c+15:]
        elif t != -1:
            print i[0:t]+termID+i[t+15:]
        elif g != -1:
            print i[0:g]
            for r in list:
                print '<option value="%s">%s</a><br />'%(str(r[0]), str(r[1]))
            print i[g+13:]
        else:
            print i
    p.close()


def main():
    engine.printheader('Report Mode')
    form = cgi.FieldStorage()
    type = form.getfirst('type',None)
    number = form.getfirst('number',None)
    chorus = form.getfirst('chorus',None)
    termID = form.getfirst('termID',None)
    pw = form.getfirst('pw',None)
    if type == None:
        if pw != '[redacted]':
            printpw()
        else:
            printmenu()
            engine.printfoot()
    elif type == '0' and number != None:
        rehearsalReport(number, chorus, termID)
    elif type == '0' and number == None:
        if chorus != None:
            if termID != None:
                list = engine.listRehearse(chorus, termID)
                printrehearse(list, chorus, termID)
            else:
                list = engine.getTermList()
                printtermlist(list, chorus)
        else:
            printrehearsemenu()
    elif type == '1' and chorus != None:
        standingReport(chorus)
    elif type == '1' and chorus == None:
        printstandingmenu()
    elif type == '2' and chorus != None:
        rosterList = engine.findRoster(chorus)
        printroster(rosterList)
    elif type == '2' and chorus == None:
        printrostermenu()
    if type != None:
```

```
        engine.printfoot()

main()
```

Report menu.html code:

```
</head>
<body><br /><table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="type" value="0" /><input type="submit"
value="Rehearsal Reports" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="1" /><input type="submit"
value="Standing Report" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="2" /><input type="submit"
value="Roster Report" /></form></td>
<td><form action="../index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Main Menu" /></form></td></tr></table>
```

Report rehearsalmenu.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="type" value="0" /><input type="hidden"
name="chorus" value="0" /><input type="submit" value="Glee Club" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="0" /><input type="hidden"
name="chorus" value="1" /><input type="submit" value="Alden Voices" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report rehearsal.html code:

```
</head>
<body>
 <table class="tab">
  <tr><th>Event Type:</th><td> <!-- EVENTTYPE --></td>
  <th>Group:</th><td> <!-- GROUP --></td>
  <th>Date:</th><td> <!-- DATE --></td></tr></table><br />
  <table class="tab">
<tr><th class="tabhead">Attendance</th>
<th class="tabhead">Groups</th>
<th class="tabhead">Credit</th>
<th class="tabhead">Officer</th></tr>
  <!-- ATTEND -->
<tr><th class="tabhead">Absent</th>
<th class="tabhead">Groups</th>
<th class="tabhead">Credit</th>
<th class="tabhead">Officer</th></tr>
  <!-- MISSED -->
<tr><th class="tabhead">Late</th>
<th class="tabhead">Groups</th>
<th class="tabhead">Credit</th>
<th class="tabhead">Officer</th></tr>
  <!-- LATE -->
<tr><th class="tabhead">Excused</th>
<th class="tabhead">Groups</th>
<th class="tabhead">Credit</th>
<th class="tabhead">Officer</th></tr>
  <!-- EXCUSE -->
</table><br />
<table><tr><td><form action="index.cgi" method="post">
```

```
<input type="hidden" name="type" value="0" />
<input type="hidden" name="termID" value="<!-- TERMID -->" />
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report rehearse.html code:

```
</head>
<body><br /><table align="center"><form action="index.cgi" method="post">
<tr><td><select name="number"><!-- LIST --></select</td>
<td><input type="hidden" name="type" value="0" />
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="hidden" name="termID" value="<!-- TERMID -->" />
<input type="submit" value="Select Rehearsal" /></td></form>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="0" /><input type="hidden"
name="chorus" value="<!-- CHORUS -->" /><input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report rostermenu.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="type" value="2" /><input type="hidden"
name="chorus" value="0" /><input type="submit" value="Glee Club" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="2" /><input type="hidden"
name="chorus" value="1" /><input type="submit" value="Alden Voices" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report roster.html code:

```
</head>
<body>
<table class="tab" align="center" style="width:50%">
<tr><th class="tabhead" width="33%">Name</th><th class="tabhead" width="33%">Voice Part</th><th class="tabhead"
width="33%">Graduation Year</th></tr>
<!-- ROSTER -->
</table>
<table align="center"><td width="50%"><button type="button"
onclick="window.location='index.cgi?type=2'">Back</button></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></table>
```

Report standingmenu.html code:

```
</head>
<body><br />
<table align="center">
<tr><td><form action="index.cgi" method="post"><input type="hidden" name="type" value="1" /><input type="hidden"
name="chorus" value="0" /><input type="submit" value="Glee Club" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="type" value="1" /><input type="hidden"
name="chorus" value="1" /><input type="submit" value="Alden Voices" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input
type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report standing.html code:

```
</head>
<body>
 <table class="tab" style="border-bottom: 0px;">
<tr><th>Currently Not in Good Standing</th></tr></table>
<table class="tab">
<tr><th class="tabhead">Name</th>
<th class="tabhead">Groups</th>
<th class="tabhead">Credit</th>
<th class="tabhead">Officer</th></tr>
  <!-- STANDING -->
<tr><td style="border-top:1px solid black;"></td><td style="border-top:1px solid black;"></td><td style="border-top:1px solid black;"></td><td>* Denotes new</td>
  </table><br />
<table><tr><td><button type="button" onclick="window.location='index.cgi?type=1'">Back</button></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Reports Menu" /></form></td></tr></table>
```

Report termlist.html code:

```
</head>
<body><br /><table align="center">
<td><form action="index.cgi" method="post">
<select name="termID"><!-- LIST --></select>
<input type="hidden" name="chorus" value="<!-- CHORUS -->" />
<input type="hidden" name="type" value="0" /></td>
<td><input type="submit" value="Show Reports" /></form></td><td><form action="index.cgi" method="post"><input type="hidden" name="type" value="0" />
<input type="submit" value="Back" /></form></td>
<td><form action="index.cgi" method="post"><input type="hidden" name="pw" value="[redacted]" /><input type="submit" value="Reports Menu" /></form></td></tr></table>
```

```
mysql> describe excuseBase;
+---------------------------+-----------+------+-----+---------+----------------+
| Field                     | Type      | Null | Key | Default | Extra          |
+---------------------------+-----------+------+-----+---------+----------------+
| excuseID                  | int(11)   | NO   | PRI | NULL    | auto_increment |
| excuse_eventBase_eventID  | int(11)   | YES  |     | NULL    |                |
| excuseChorus              | int(11)   | YES  |     | NULL    |                |
| excuseRFID                | int(11)   | YES  |     | NULL    |                |
| excuseNotes               | char(255) | YES  |     | NULL    |                |
+---------------------------+-----------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> describe attendBase;
+--------------------------+---------+------+-----+---------+----------------+
| Field                    | Type    | Null | Key | Default | Extra          |
+--------------------------+---------+------+-----+---------+----------------+
| attendID                 | int(11) | NO   | PRI | NULL    | auto_increment |
| attend_RFID              | int(11) | YES  |     | NULL    |                |
| attend_eventBase_eventID | int(11) | YES  |     | NULL    |                |
| attend_late              | int(11) | YES  |     | NULL    |                |
+--------------------------+---------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> describe eventBase;
+-----------------------+----------+------+-----+---------+----------------+
| Field                 | Type     | Null | Key | Default | Extra          |
+-----------------------+----------+------+-----+---------+----------------+
| eventID               | int(11)  | NO   | PRI | NULL    | auto_increment |
| eventDatetime         | datetime | YES  |     | NULL    |                |
| eventType             | int(11)  | YES  |     | NULL    |                |
| eventStatus           | int(11)  | YES  |     | NULL    |                |
| eventChorus           | int(11)  | YES  |     | NULL    |                |
| event_termBase_termID | int(11)  | YES  |     | NULL    |                |
+-----------------------+----------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

```
mysql> describe reportBase;
+---------------------------+---------+------+-----+---------+----------------+
| Field                     | Type    | Null | Key | Default | Extra          |
+---------------------------+---------+------+-----+---------+----------------+
| report_number             | int(11) | NO   | PRI | NULL    | auto_increment |
| report_eventBase_eventID  | int(11) | YES  |     | NULL    |                |
| report_eventBase_eventType| int(11) | YES  |     | NULL    |                |
| report_chorus             | int(11) | YES  |     | NULL    |                |
| report_date               | date    | YES  |     | NULL    |                |
+---------------------------+---------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

```
mysql> describe standingBase;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| fname | char(25) | YES  |     | NULL    |       |
| lname | char(25) | YES  |     | NULL    |       |
| RFID  | int(11)  | YES  |     | NULL    |       |
| isNew | int(11)  | YES  |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

```
mysql> describe termBase;
+--------+---------+------+-----+---------+----------------+
| Field  | Type    | Null | Key | Default | Extra          |
+--------+---------+------+-----+---------+----------------+
| termID | int(11) | NO   | PRI | NULL    | auto_increment |
| year   | int(11) | YES  |     | NULL    |                |
| term   | char(2) | YES  |     | NULL    |                |
| active | int(11) | YES  |     | NULL    |                |
+--------+---------+------+-----+---------+----------------+
4 rows in set (0.00 sec)
```

```
mysql> describe studentBase;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| fname     | varchar(25) | YES  |     | NULL    |       |
| lname     | varchar(25) | YES  |     | NULL    |       |
| studentID | int(11)     | YES  |     | NULL    |       |
| RFID      | int(11)     | YES  | UNI | NULL    |       |
| officer   | int(11)     | YES  |     | NULL    |       |
| acgroup   | int(11)     | YES  |     | NULL    |       |
| credit    | int(11)     | YES  |     | NULL    |       |
| voicePart | int(11)     | YES  |     | NULL    |       |
| gradYear  | int(11)     | YES  |     | NULL    |       |
| attended  | int(11)     | YES  |     | NULL    |       |
| missed    | int(11)     | YES  |     | NULL    |       |
| excuse    | int(11)     | YES  |     | NULL    |       |
| chorus    | int(11)     | YES  |     | NULL    |       |
| late      | int(11)     | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
14 rows in set (0.00 sec)
```

References

Timetrex, (2012). Retreived November 4th, 2012 , from https://www.timetrex.com/time_and_attendance.php

Attendance App, (2012). Retreived November 6th, 2012 , from http://www.dave256apps.com/attendance/

The Attendance App, (2012). Retreived November 4th, 2012 , from https://itunes.apple.com/us/app/the-attendance-app/id483346207

Check In Easy, (2012). Retreived November 5th, 2012 , from https://itunes.apple.com/us/app/check-in-easy-guest-list-event/id491661649?mt=8

iAttendance, (2012). Retreived November 5th, 2012 , from https://itunes.apple.com/us/app/iattendance-great-attendance/id472600513?mt=8