**Employee Management System**

* What is this System?

This employee management system created by r/programmerhumour is designed to simplify the process of record maintenance of employees in an organisation and modernize the old pen and paper style system of data management. This system can be used in managing the information of employees for HR functions and can be implemented in practically any business environment, as it’s unique design allows easy collaboration and adaptation to easily manage and maintain employee data and business records. In general, this software can be used as a form of comprehensive Human Resource Management System software.

* What does this System do?

This essential goal of this system is to manage your employees and perform tasks that are essential for a healthy work/business environment. Tasks such as tracking employee hiring dates, sick leave dates, disability status, viewing records of employee complaints to address and even employment status have never been easier with this software. The simplistic yet effective layout of the User Interface provided allows for seamless interaction and collaboration with the database (using SQL and database manipulation techniques in the background) while ensuring that you have the ease of access to your data all in one location.

* What pre-requisites are needed to use this software?

Users interacting with our software need only have a basic understanding of how to use a computer and be familiar with using computer programs or any software in general to be able to interact with this system. But because our software has been designed to be very user friendly and offer a simple interaction yet effective result, navigation and mastering the controls of this software will be a easily obtained skill after a few uses.

* Constraints of this software

The following criteria are required for the program to run correctly:

* Program is only compatible with the Windows environment.
* Systems running this software must have a minimum of 4GB RAM
* Systems must have .NET 4.5 package installed
* 5GB of free HDD space

\*Optional: For best results, systems running this software should have a 4th generation Pentium processor or better. (AMD equivalent: Athlon x2 or better)

**Screenshots of what this software’s UI looks like:**

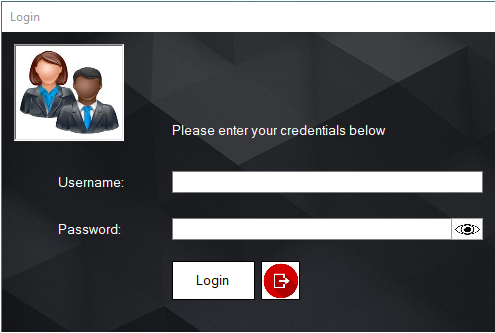
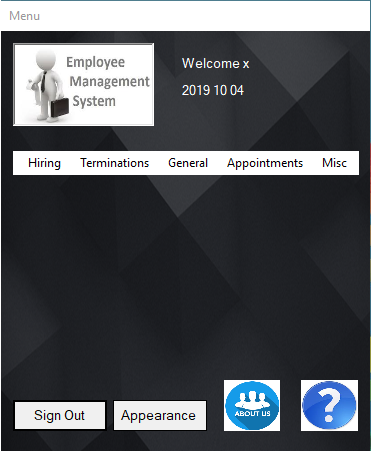
****

Figure - Login Menu

****

****

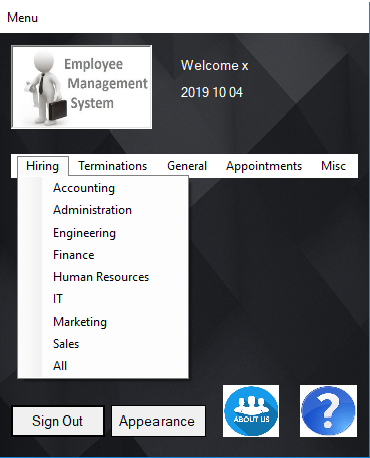
****

Figure - Home Screen Menu

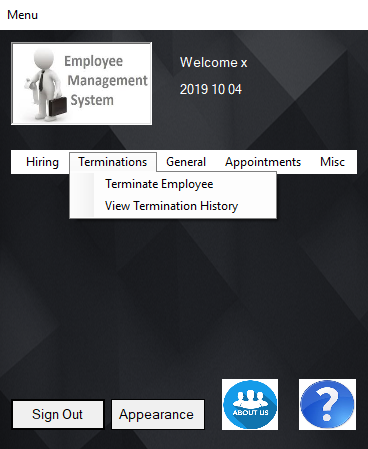
****

Figure - Home Screen Menu2

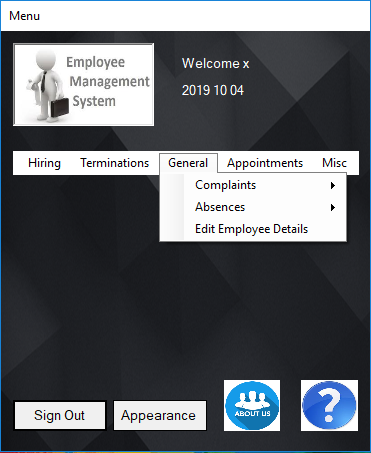
****

Figure - Home Screen Menu3

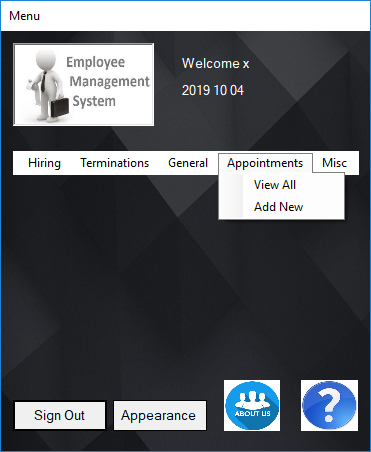
****

Figure - Home Screen Menu4

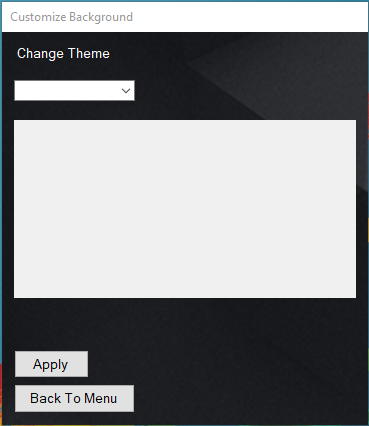
****

Figure - Change Appearance Screen

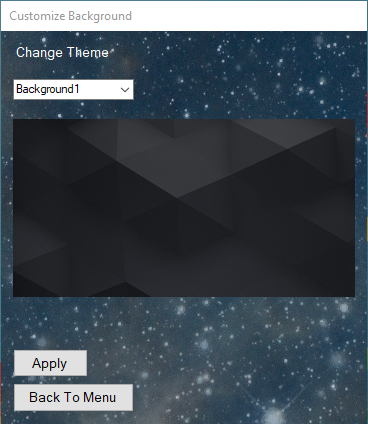
****

Figure - Change Appearance Screen2

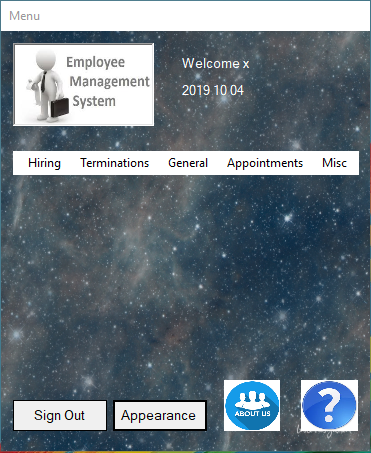
****

Figure - Changed Appearance Home Screen

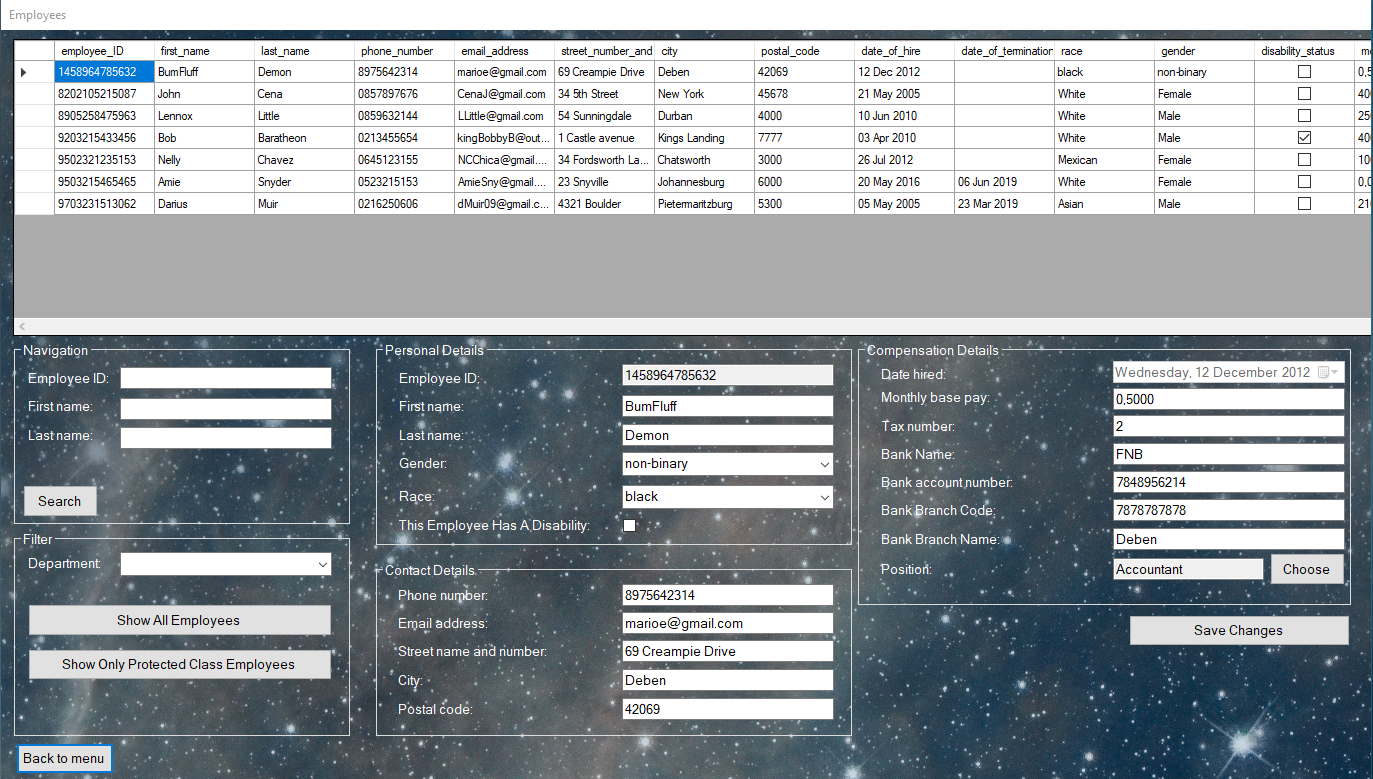
****

Figure - Employees Form

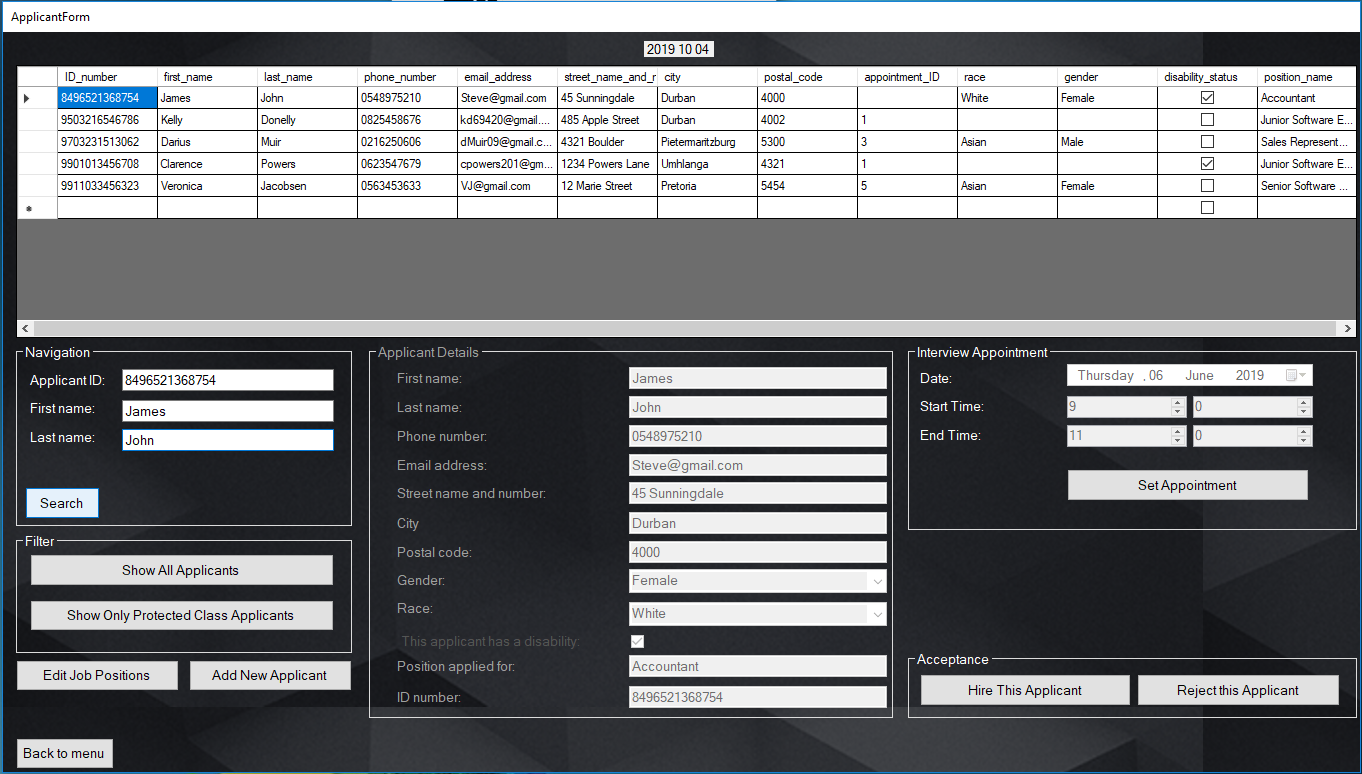
****

Figure 10- Applicant Form

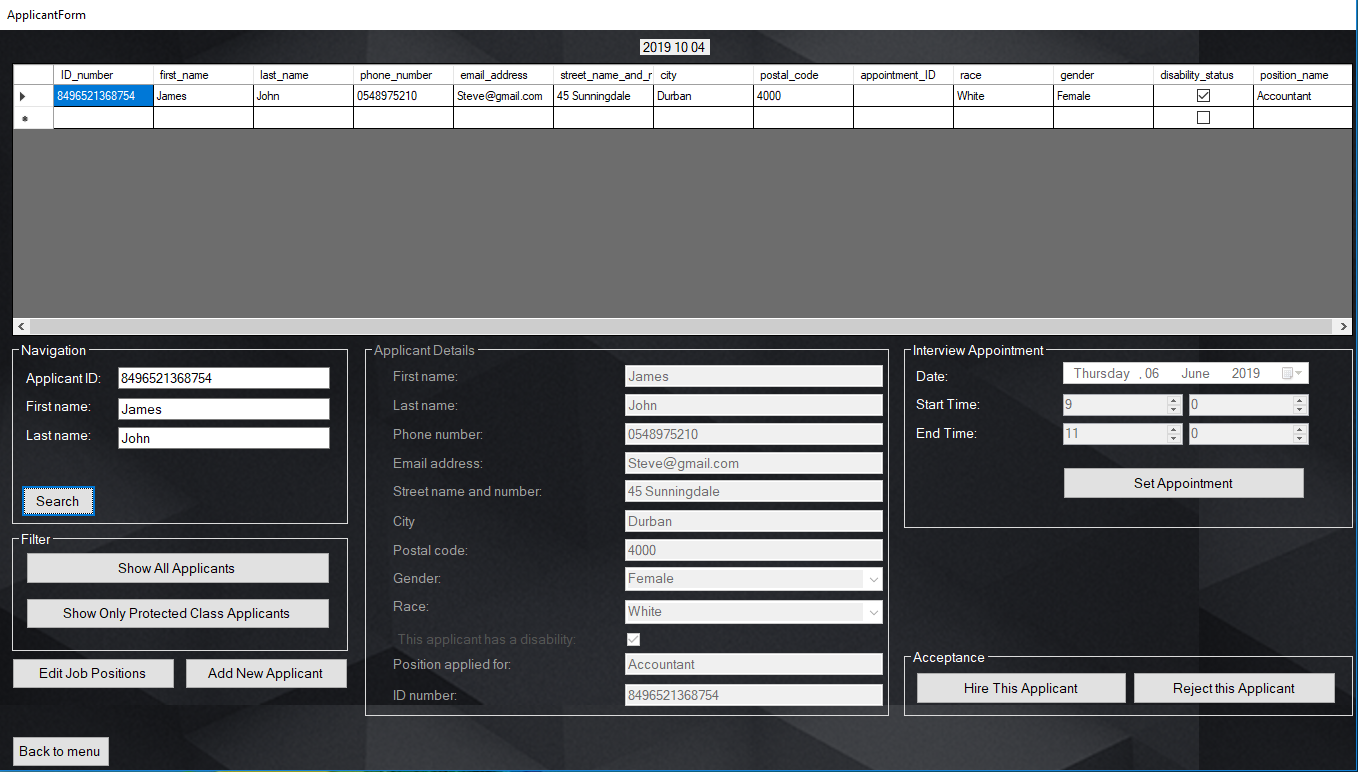
****

Figure - Applicant Form2

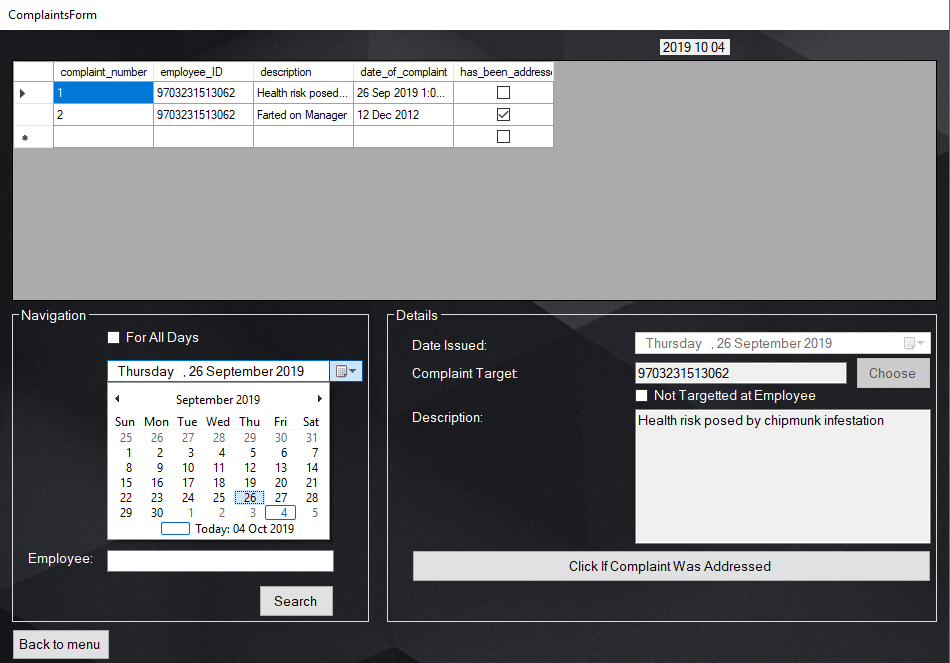
****

Figure -Complaints Form

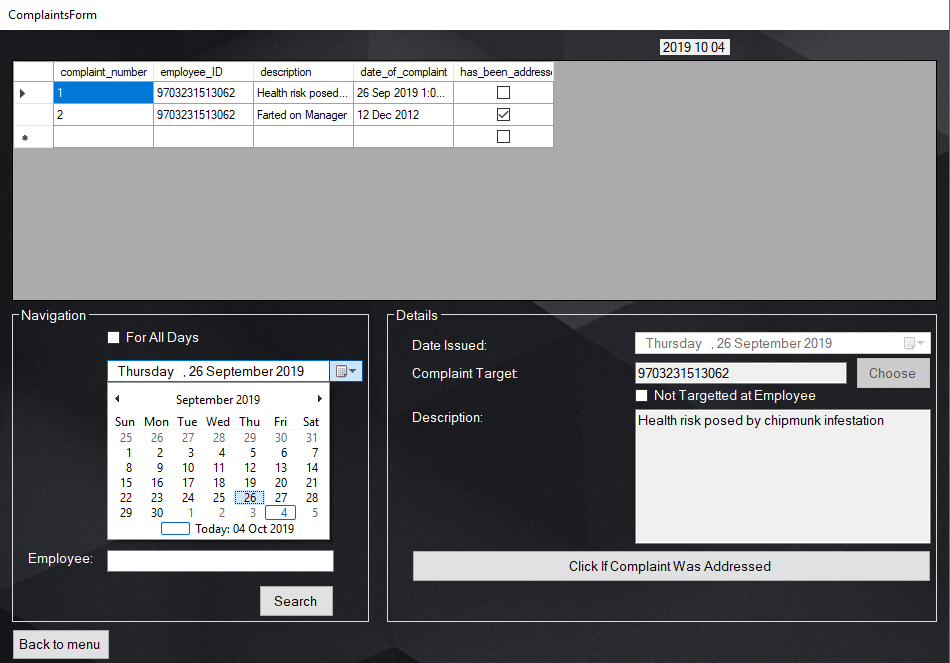
****

Figure - Complaints Form2

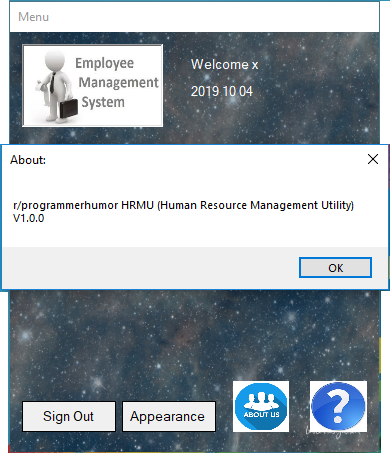
****

Figure - About Form

**SQL QUERIES USED IN THIS SOFTWARE**

//selects only active employees: for adding to appointments, editing details, filing complaints

Select \*-employee\_status from employee where employee\_status = active

//select all applicants by department

Select \* from applicant where position\_ID in(select position\_ID from position where department = @dept)

HIRING PROCESSES:

//adding an applicant

If exists(select \* from applicant where ID = @id)

Begin

Update applicant set

Phone\_num = @phoneNum

Email = @email

Street\_det = @street

City = @city

Post\_code = @post\_code

Disability\_status = @disability

Resume = @resume

Where ID = @id

End

Else

Begin

Insert into applicant values(@id, @Fname, @Lname, @........)

End

//add a resume to an applicant

Update applicant set resume = @resume

Where ID = @id

//attach an appointment to an applicant

Update applicant set appointment\_ID = @aptID

Where ID = @id

//delete an applicant

Delete from applicant where ID = @id

//hire an employee

If exists(select \* from employee where EmployeeID = @applicant\_id)

Begin

Update employee set

Salary = @salary

Position = @position

…….

Where ID = @id

End

Else

Begin

Insert into employee values(@id, @Fname, @Lname, @........);

Update employee set

Salary = @salary

……

End

Update position set (num\_available = num\_available – 1) where positionID = @position

TERMINATION PROCESSES:

//select only terminated employees for termination history

Select\* from employee where not employee\_status= active

//terminate an employee

//remove them from appointments

Delete from appt\_guest where Employee\_ID = @id

//change employed status

Update employee set

Employee\_status = @terminationType

Termination\_date = @terminationdate

Where employee\_ID = @id

//delete terminated employees

Delete from complaints where employee\_ID in (select employee\_ID from employee where not employee\_status = active);

Update payments set employee\_ID = NULL

Where employee\_ID in (select employee\_ID from employee where not employee\_status = active);//unless we remove the payments table

Delete from employee where not employee\_status = active;

APPOINTMENTS

//add new appointment: one use is when setting the interview appointment for an applicant.this will execute just before the “//attach an appointment to an applicant” code

>check that the input times do not coincide with existing appointments

Insert into appointment values(……)

>go to employee table,filtered to only show id,fname,lname , and only for active employees.

>on btn add\_to\_appt click:

Insert into appt\_guest values @EntryNum,@appt\_ID,@employee\_ID

>send emails

//change appt details

> go to appointments form

>when user clicks an appointment, changes details and clicks done:

Update appointment set start\_date = @startDate, end\_date = @endDate where appointment\_number = @aptNum

>if the user clicks remove guest button, go to a reduced employee form:

Select ID,fname,lname from employee where ID in (select employee\_ID from appt\_guest where appointment\_number = @aptNum)

>when the user clicks an employee and clicks the uninvite button:

Select email from employee where ID = @ID

>add these to an array UNINVITED

Delete from appt\_guest where appointment\_number = @aptNum AND employee\_ID = @ID

>when the user clicks the done button go to the appointment form

>if the user clicks add guest button, go to a reduced employee form:

Select ID,fname,lname from employee where not ID in (select employee\_ID from appt\_guest where appointment\_number = @aptNum)

>when the user clicks an employee and clicks the invite button:

Insert into appt\_guest values (@entryNum, @aptNum, @ID)

>when the user clicks the done button go to the appointment form

>send update notices

Select email from employee where employee\_ID in(select employee\_ID from appt\_guest where appointment\_number = @aptNum)

>for each row: sendEmail(hrEmail,row.emailValue,”Change of Appointment details”, “You are hereby informed of the following change: the appointment previously scheduled for dateOLD at starttimeOLD to endtimeOLD has been rescheduled to dateNEW at starttimeNEW to endtimeNEW”)

>for each email in UNINVITED: sendEmail(hrEmail,uninvited.emailValue,”Change of Appointment details”, “You are hereby informed of the following change: you are no longer scheduled for an appointment on dateOLD starttimeOLD to endtimeOLD”)