Hourly Employee Roster

Katrina Gudnin

BYU MBA 614- Spreadsheet Automation

4/13/2011

Executive summary

When I was working at Utah Valley University, I saw a lot of time-consuming processes in my every-day work that I wish were simplified/ automated at the time. And this was true for everyone on my team. One of my fellow co-workers had numerous responsibilities and was constantly trying to juggling several jobs at once. In addition, being interrupted by numerous phone calls, she would sometimes start a project and forget to finish everything that she intended to do. This project is meant to make her life easier.

Several user forms were created to enable a user to easily manage the records on the employee roster: add, edit, delete and, mainly, move records between different worksheets. My goal was to try to minimize a chance of errors and to simplify the process.

Implementation

Worksheets

The workbook consists of several worksheets, each for a certain step in the hiring process:

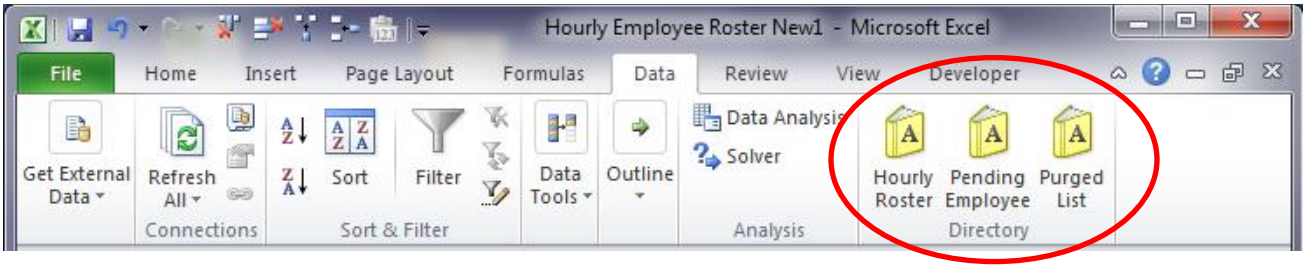
- Initial application ("Pending Student" sheet that keeps track of all the applicants)
- No Hire List ("Purged List" keeps track of records of the applicants who for one reason or another were not approved)
- Employment (when confirmation from HR office is received, an applicant is moved from the "Pending" list to Full-Time employment:
    - Day Shift Employees ("Day Hourly Roster")
    - Night Shift Employees ("Night Hourly Roster")

When a new applicant is received, the information is entered in the Pending sheet, where the person stays while the HR office is processing the information and conducting the background checks. As soon as the information is verified, the HR office notified the hiring department of the decision, and then the applicant is either moved to the Purged List or Hourly Roster.

Ribbon

The user has several custom buttons available in the Data Tab to choose from, depending on employee's current standing in the hiring/employment process (Figure 1).

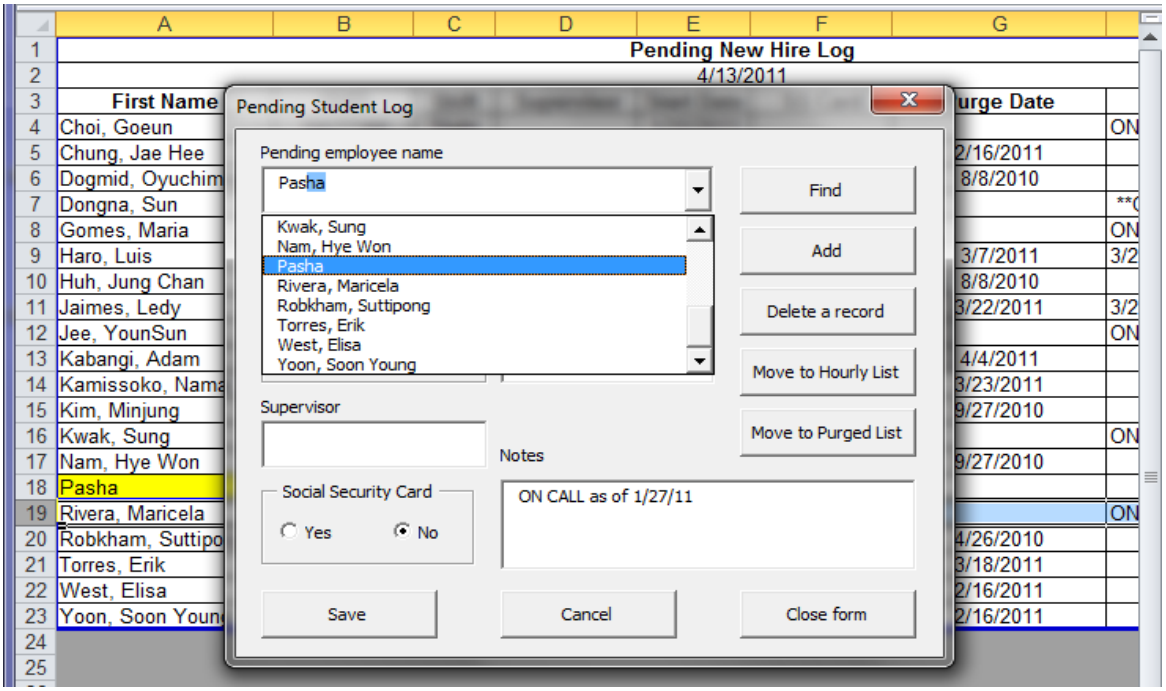
Figure 1



Forms

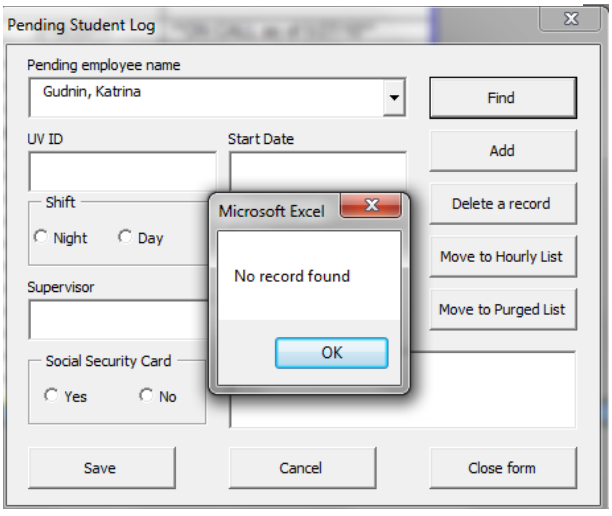
Each custom button from the ribbon calls a different user form: Purged records, Pending applicants, and Hourly employees. When each form is opened, the corresponding worksheet becomes active and the list of employees from that worksheet is automatically populated into the drop box (Figure 2). The discussion in this section is based on the Pending user form example since the procedures discussed in are the same for all user forms.

Figure 2

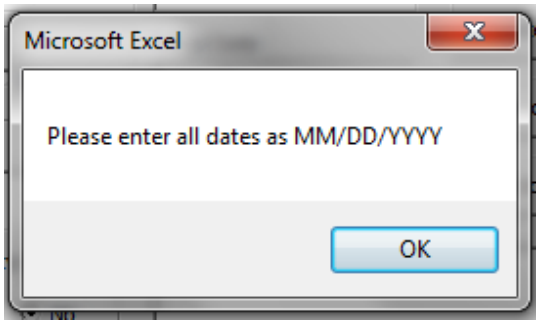


Each form allows the user to search the records, edit, and delete them. Find button searches the database, populates the fields when a match is found, and notifies the user when no employee, matching the criteria, exists (Figure 3). Delete button returns the same message if no record found and prompts the user to select a name if Pending Employee Name box is blank. When the record is deleted or a new record is added to the list, the Pending Employee Name combo box is updated to reflect the changes and the employees on the corresponding worksheet are sorted in the alphabetical order. Save button at the bottom of the form validates the UVID , which has to be a unique

Figure 3



eight-digit number, and the format of the date. If text or unrecognized number format is entered into a Start Date, Purge Date, or other fields requiring a date, the following message appears:

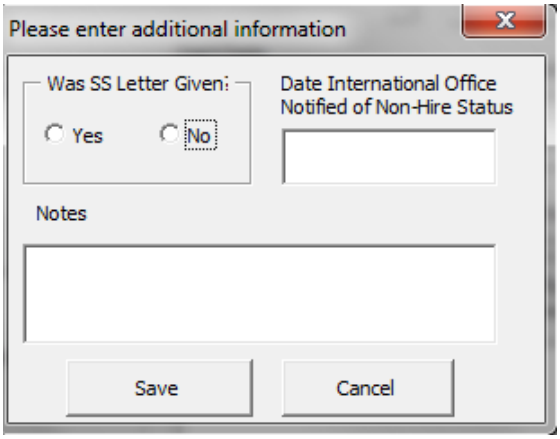


Moving records between the worksheets

In addition to searching and manipulating the records on each list, the Pending form allows the user to move employees to either "Purged List" or "Hourly Roster". When an applicant is moved to Purged records, a user is prompted to enter the additional information received from the HR Office (Figure 4).

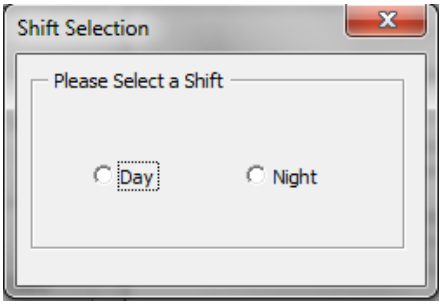
When the information is entered and the user clicks Save button, the record is added to the list of employees on the Purged worksheet and deleted from the Pending worksheet. The record is automatically removed from the Pending Employee dropdown list and added to the Purged Employee dropdown list. After the record is transferred, the form automatically closes and the user returns to the Pending form. The Cancel button makes no changes to the record, takes the user to the Pending form and keeps it populated with the employee's original information.

Figure 4



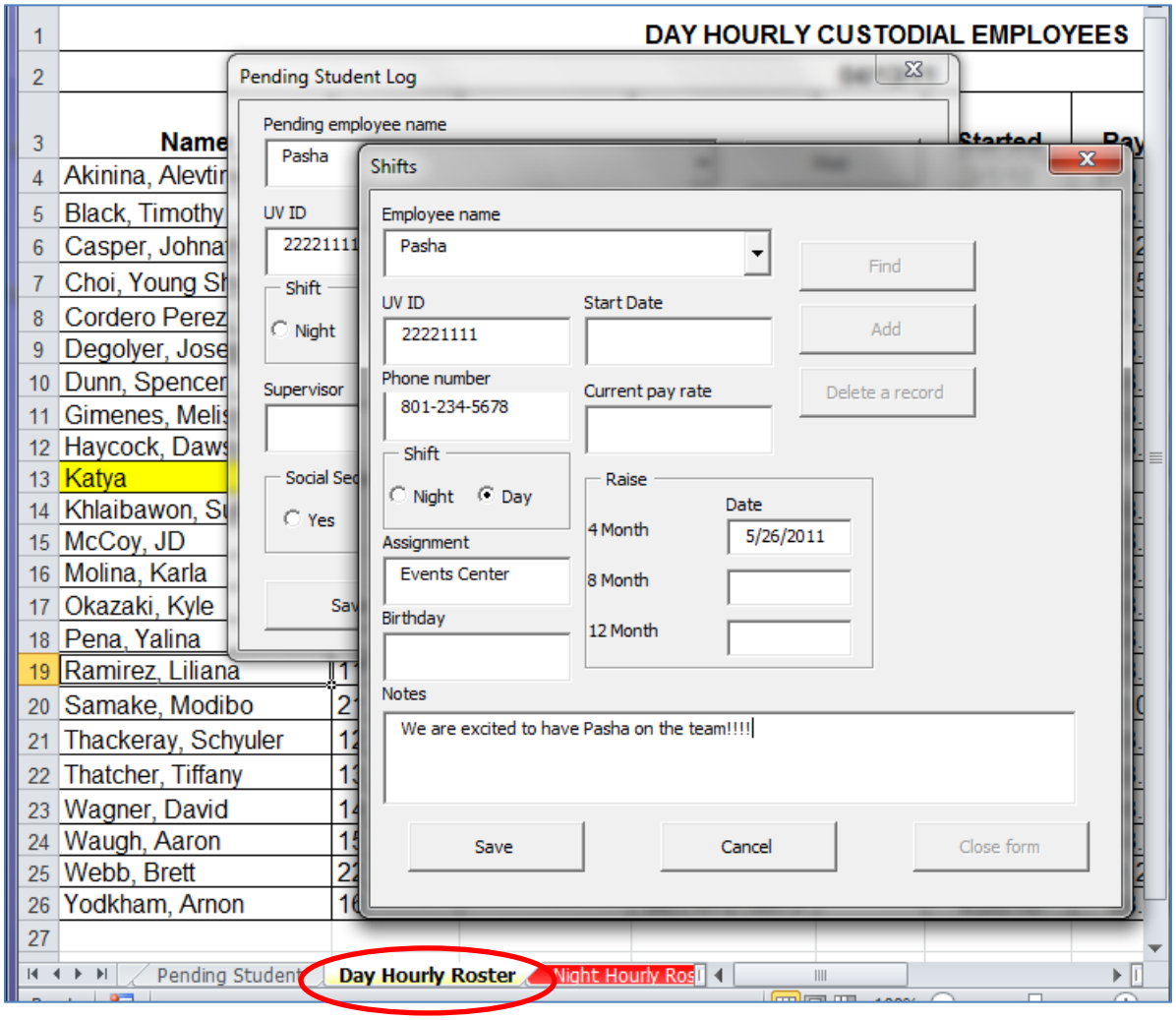
When an employee is moved from the Pending list to the Hourly Roster, the user is prompted to select the shift before advancing to the next form (Figure 5).

Figure 5



As soon as the shift is selected, the Shift Selection form closes, Shift form opens, and the "Day Hourly Roster" or "Night Hourly Roster" worksheet, depending on the selection, becomes active. The existing information from the Pending form is transferred over to the shift form, where the user can edit the record or enter additional information (Figure 6).

Figure 6



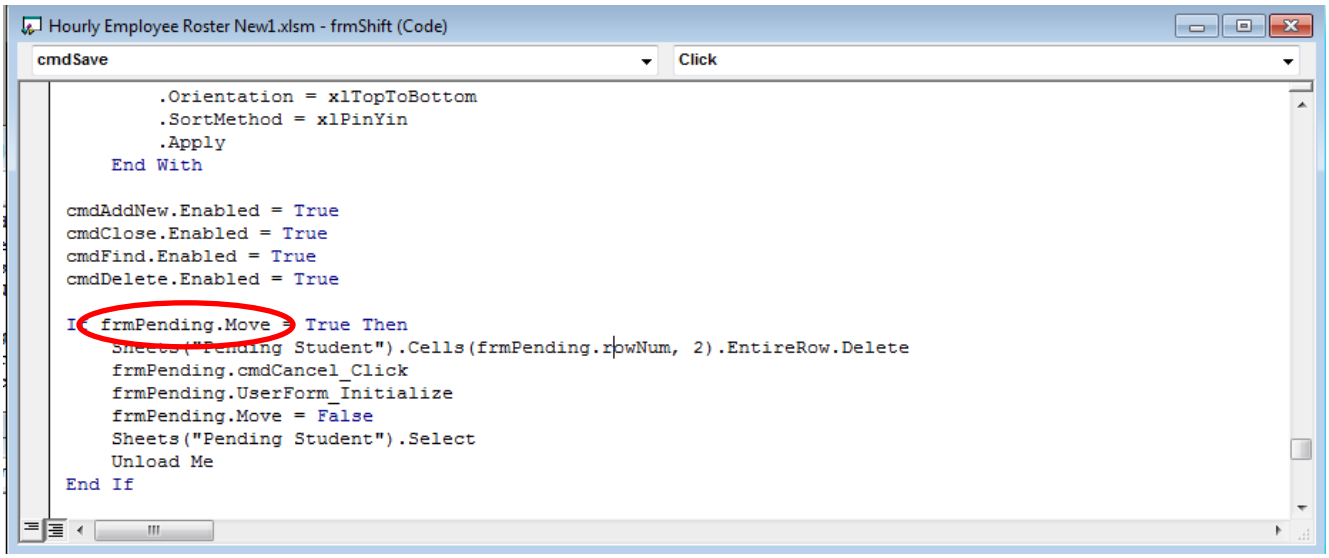
The Add, Find, Delete a Record and Close the Form buttons are greyed out allowing the user to either cancel or proceed with moving an employee. Again, the Save button adds a new record to the worksheet, deletes it from the Pending list, and makes the changes to the Employee Name dropdown lists. The Cancel button makes no changes to the records, closes the Shift form, and takes the user back to the Pending form.

Learning points

This project was my first experience coding in VBA. I had never used macros before and, thus, this was a great opportunity to learn how I can apply the material learned in class in the real world situation. I learned how to work with user forms (especially combo boxes), how to compare different type of data, and format cells. I was faced with a few challenges as well.

One of the difficulties that I encountered was when I was moving employees from the Pending List to the Hourly Roster. I wanted the Cancel and Save buttons to behave differently depending on whether the user opened the form from the ribbon to manipulate the existing records or accessed the form when moving the records. When opened from the ribbon, the Shift form should stay active. When opened through the Pending form, these buttons should clear the form and/or save a record, close the Shift Form and return to the Pending form. My solution was to create a global variable (Boolean) that would change the value accordingly. Thus, when I was moving employees, the form would close itself and return to the previous one, and when modifying the existing records, it would stay open. Figure 7 shows the additional code that was added to the save sub to achieve this behavior.

Figure 7



Another issue that I was faced with was modifying the Hourly Roster form. There are two shifts available: day and night, my co-worker needs to keep separate worksheets for each one. The information on both sheets is the same and I wanted to use the same subs for both. So I created another user form that would prompt the user to choose the shift before loading the form. A variable was used to reference the chosen sheet and thus I could use the same procedures to manipulate both worksheets.

In the end, even though the project was challenging and time-consuming, it was valuable, interesting, fun and definitely worth that time!