Claude Butnaru – CSMS | Dev Log

First of all, this application is designed from scratch by me, Claude Butnaru with no templates, drag and drop or anything like that. I made this app ( the UI design, back-end coding, and database) with pure code, no NetBeans, no nothing. I wrote the code in Microsoft VS-Code, one person in class said something weird that VS-Code had drag and drop – Crazyness! there is no drag and drop in VS-Code he probably was thinking about NetBeans by Apache. I don’t know maybe people should keep their egoes in check. Here is a shot of my code in VS-Code – no drag and drop here.

Text

Description automatically generated

I started keeping this log about a week after I got started on this project, so I’ll try to remember what I did each day

1. 2/4/22 – 2/7/22. I started working on the main class and started to get a feel for Java as I’m not used to it. I found that Java has different interfaces to choose from called LookAndFeel and you can overwrite them with the UIManager. Different elements such as the window title bar or color. The way this is done is by hiding the native frame of the Microsoft windows and bring in the Oracle UI. The oracle UI can be overwritten. I spend several days on this to make it look to my likings. I chose a retro look something similar how windows looked back in the ‘90’s.
2. 2/8/22. Still worked on the front-end UI trying to figure everything out and how to place elements on a 2d vector x, y. Everything has to be calculated since the results will be displayed programatically from a database. I learned a lot about java swing and many of the elements that comes with. Some of them include JFrame, JButton, JPanel etc
3. 2/9/22. I started working on the main login Ui. I decided to make a Ui where a manager or an employee can log in. Also, I added a calculator, which in the future I would like to connect straight with the database, so the inventory could be done easier. I came across some layout problems and got sort of stuck because I want the app to be sort of a liquid layout no matter what size it is. And then I came across the GridBag displaying everything on a grid. This is perfect for my app since it keeps everything symmetrical no matter the size of the window and also centered.
4. 2/10.22. I’ve decided to rewrite everything inside the grid, so I did, took me several hours.
5. 2/11/22. I started writing the main menu bar and all the drop-down menus. I decided to place a button bar under the main menu bar. And I wrote it as a JPanel. It didn’t work out at all because of my grid. When I would resize the window, this bar wouldn’t resize correctly. I stopped working I was stuck again so I paused.
6. 2/12/22. After about an hour working today, I came across the JToolBar this was the perfect element for my button bar in code is buttonToolBar in the main class. I also added some buttons to it.
7. 2/13/22. I needed to learn about events in Java to use them for just about everything, from the buttons to refreshing the page when I display from the database. So, I spent some time learning about the type of events they have in Java. Then I added some event handlers for the buttons. I also stylized the buttons inside these events, stuff like the color on hover on click etc. I made the buttons transparent and put some images that I found on the internet as backgrounds. This can be found in a separate class called ToolBarButton. This class creates a button with a constructor with parameters that take the icon of the button and the text of the button.
8. 2/14/22. I also needed to learn how to connect a portable database in Java. I looked around and I’ve decided to use a small portable database called jdbc sqlite. It seems to be very popular with small java apps and the documentation is good. I built a database class called DB and made a basic connection; it seems to be working.
9. 2/15/22. I created 2 classes: ManagerUI and EmployeeUI. The way the login page works is there is an integer in the main class called displayNumber. When diplayNumber is 0, it displays the 3 buttons only( manager, employee, cash register) when it’s set to 1, I refresh the UI and display the login and password fields. That’s as far as I got with this. Eventually, I’ll be using this integer more.
10. 2/16/22. Today I worked more on stylizing this app. I wasn’t happy with the colors, so I made everything dark.
11. 2/17/22. I worked on EmployeeUi. Made only one tab. Added the employeeUi code in the switchUI method in the Main class. When the integer display number is set to 2 the employeeUI is diplayed when the button elpoyee login is clicked. Added setDisplayNumber in LoginUI action event.
12. 2/18/22. I got stuck last night in the ManagerUI class. setLoginButtonsVisible() method the variable displayNumber from main class would reset to 0 after pushing the login button. After many breaks and taking time off again and again, I realized that because this variable was in the Main class, and it wasn’t static I had to create a new Main object to set it inside the action handler. So, every time I clicked a button, the event handler ran, and it would create a new Main object and it would reset this variable, so I had a choice of just to make it static or make it local, so I chose to make it local since the main class is too cluttered anyway and I wrote a getter and a setter for it . It’s about 1:20 am and finally figured out. If it’s set to 0 it shows the login and password fields, if it’s set to 1 it refreshes the UI and opens the ManagerUI, if it’s set to 2 it refreshes and opens the EmployeeUI, and if it’s set to 3 it opens the calculator. Here is a shot of the action handlerthat was a mistake displayNumber3 should be in cash register.
13. I started working on the database class (DB) by making a connection and making a method – createTableEmployees. I could let the manager to add more tables, but I think I’m just going to hard code this table in since for this application there won’t be a need to add to many categories. However, for products there will have to be categories and products in each category.

Products

Employee

Categories

Products

Duties

Clock and payroll

Here is the basic createTableEmployees() method I created.

|  |
| --- |
| public static void createTableEmployees() {          // SQL statement for creating a new table          String sql = "CREATE TABLE IF NOT EXISTS employees (\n"                  + " id integer PRIMARY KEY,\n"                  + " name text NOT NULL,\n"                  + " capacity real\n"                  + ");";            try (Connection conn = DriverManager.getConnection(url);                  Statement stmt = conn.createStatement()) {              // create a new table              stmt.execute(sql);          } catch (SQLException e) {              System.out.println(e.getMessage());          }      } |
|  |

1. 2/19/22 I created EmployeeLabel class. This class inherits from JLabel (native to java swing lib) this way I don’t have to repeat the name of the label every time I need to modify its properties. Also, it can create a new label in other classes with this class just by creating a new object. The labels have mouse listeners for every action. I’m not sure that this is necessary, but I use them to change the labels background colors on mouseovers.
2. 3/2/22 and 3/3/22 Ive been working on displaying the employee pane I made a small method to display the lables in alternate colors.
3. *// Sets the labels colors*
4. private void setLabelColors() {
5. if (ManagerUI.countLabels % 2 == 0) {
6. setBackground(normalColor1);
7. } else {
8. setBackground(normalColor2);
9. }
10. }

Also, I created another class that creates the textfields. I figured I display the employees on textfields so they can be modified right there, and I’ll put a button on the bottom to save the changes into the database. I had to put this variable (image above) countLabels in the ManagerUI class because if I put it in the EmployeeLabel class it gets increased every time the class is created, and nothing happens. So, I increase it in the constructor in EmployeeLabel. So, as you can see int the image above if countLabels is even the label is created in one color and odd in the other. Here is a shot:

A screenshot of a computer

Description automatically generated with medium confidence

Since this is set straight into the EmployeeLabel class all labels created will be in alternated colors. Even the ones that will be displayed on the left. I forgot to say on the left there will be labels with names from the database. As you click on a name on the left the info will be displayed on the right in these fields. I’m going to stop here for tonight.

16. 3/4/22 I modified the method that displays the employees into a a method that returns an arrayList. Initially I was going to use a simple string array but an arrayList is more versitile in case I have to do more complicated stuff. The method is inside the DB class. Shot below:

Text

Description automatically generated

I added an an ID label and textfield on top of the employee UI. The ID is an autoincremented integer in the database.

        EmployeeLabel IDLbl = new EmployeeLabel("  Employee ID");

        EmployeeTextField IDTxt = new EmployeeTextField("Here is the ID");

employeePanel.add(IDLbl);

        employeePanel.add(IDTxt);

I got it to work to display the first person from the database on the employee interface. I get it from the list and display all the info into the textfields. shot below

A screenshot of a computer

Description automatically generated with medium confidence

Here is a sample from the method inside ManagerUI that displays the info:

Text

Description automatically generated

But this was just to see if it works eventually I’ll have to change it to work dinamically with all the employees;