CCNY Tech Group

Due 4/27/21

Second phase report

Group S: Joshua Chang, Sakil Khan, Huihong Yu, Nana Bonsu, Dipankar Chakravorty

**Table of Contents**

[1. Introduction 2](#_Toc70376513)

[2. All Use Cases 3](#_Toc70376514)

[2.1 Login & User account 4](#_Toc70376515)

[2.2 Payment and Delivery 4](#_Toc70376516)

[2.3 Discussion Forum/Complaint System 5](#_Toc70376517)

[2.4 Change Recommended Products 6](#_Toc70376518)

[3. E-R Diagram 6](#_Toc70376519)

[4. Detailed Design 7](#_Toc70376520)

[4.1 Create Account 7](#_Toc70376521)

[4.2 Product Recommendations 7](#_Toc70376522)

[4.3 Customer Review 7](#_Toc70376523)

[4.4 User Account 8](#_Toc70376524)

[4.5 Transaction 8](#_Toc70376525)

[5. System Screens 9](#_Toc70376526)

[6. Minutes 13](#_Toc70376527)

[7. Address of git repo 14](#_Toc70376528)

# Introduction

Figure 1 shows the overall class collaboration diagram of the entire computer store system. It is comprised of all users, I/O which are UI screens, and datastores which are xml files. Only connections between all of the classes are shown; no interactions between them are shown.

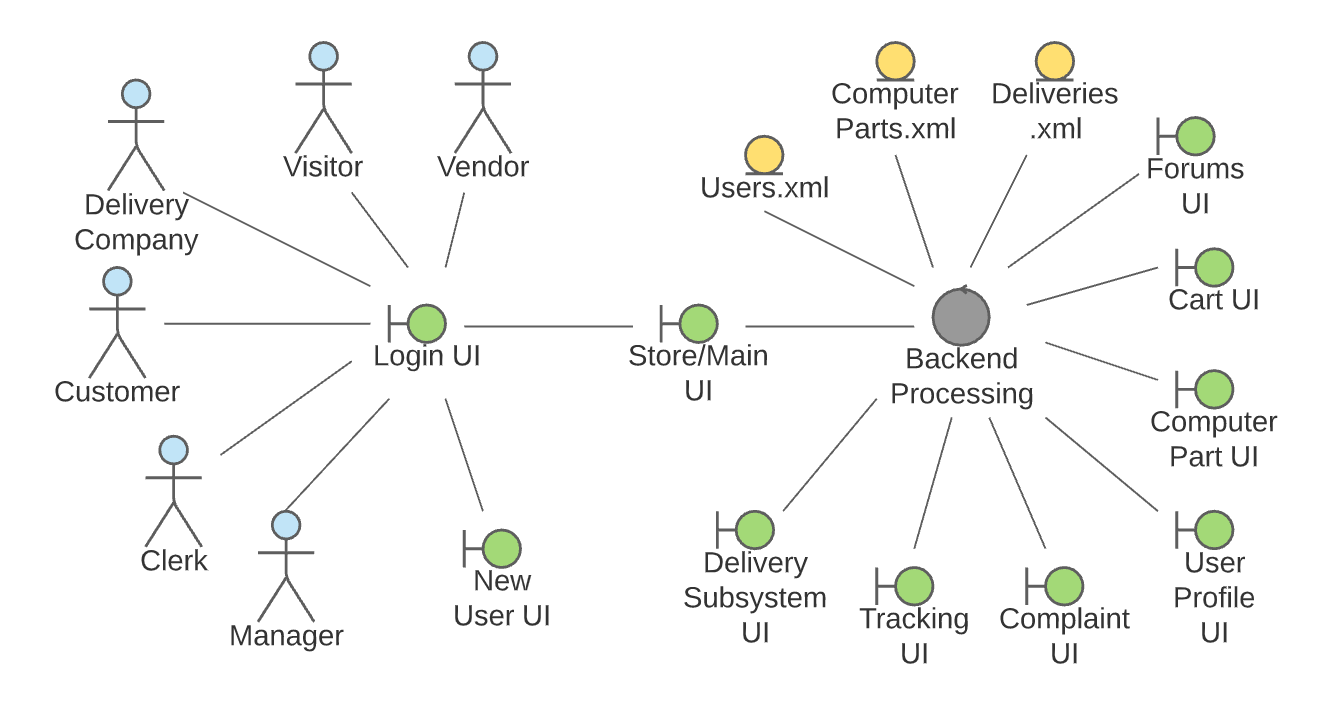


Figure 1: Overall class collaboration diagram

# All Use Cases

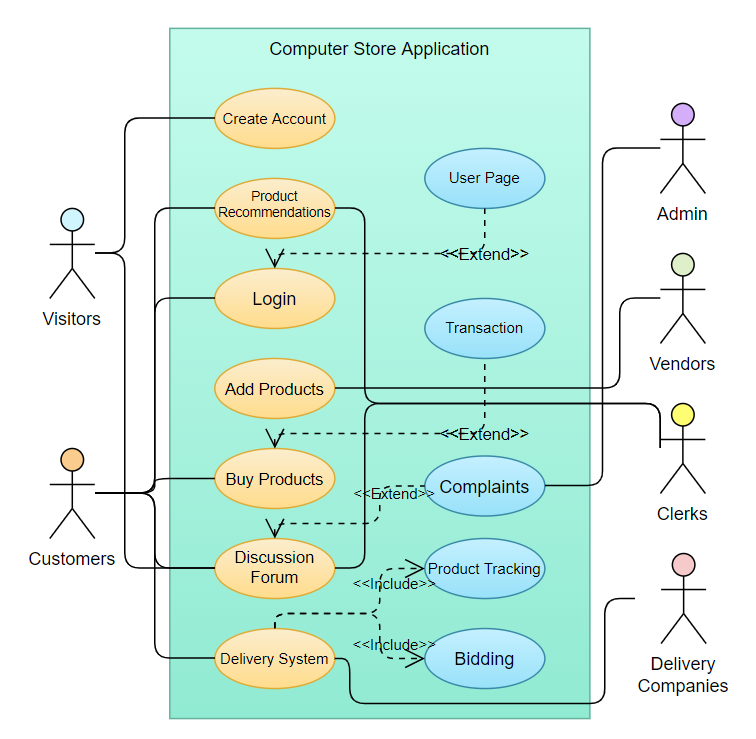


Figure 2: Overall use case diagram

Figure 2 is a use case diagram that lists the key features of the application, like creating accounts or viewing recommended items. It also shows all the users of the system and which feature they will use and interact with.

## Login & User account

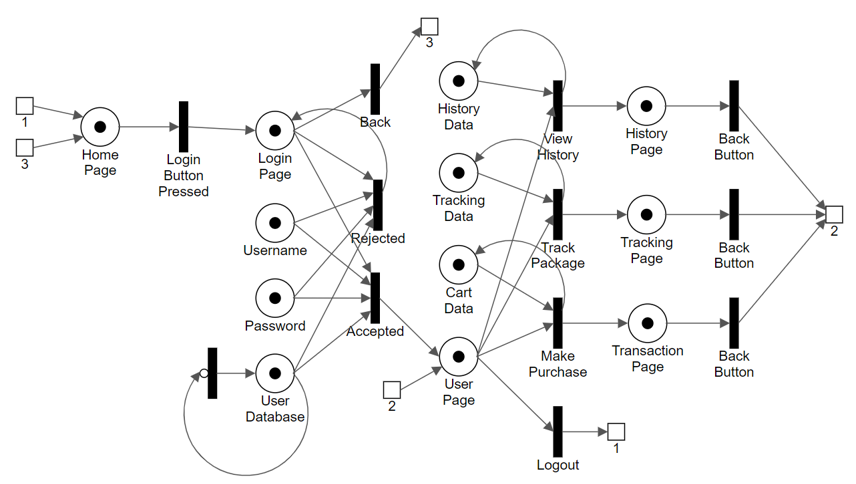


Figure 3: Petri-net for user logging in and accessing user information

The login and user account features are shown in figure 3. We start by clicking the login button in the homepage and login with the right credentials matched with the user database. Once in the user page, customers can see their purchase history, tracking info, cart items and can return to the homepage by logging out.

## Payment and Delivery

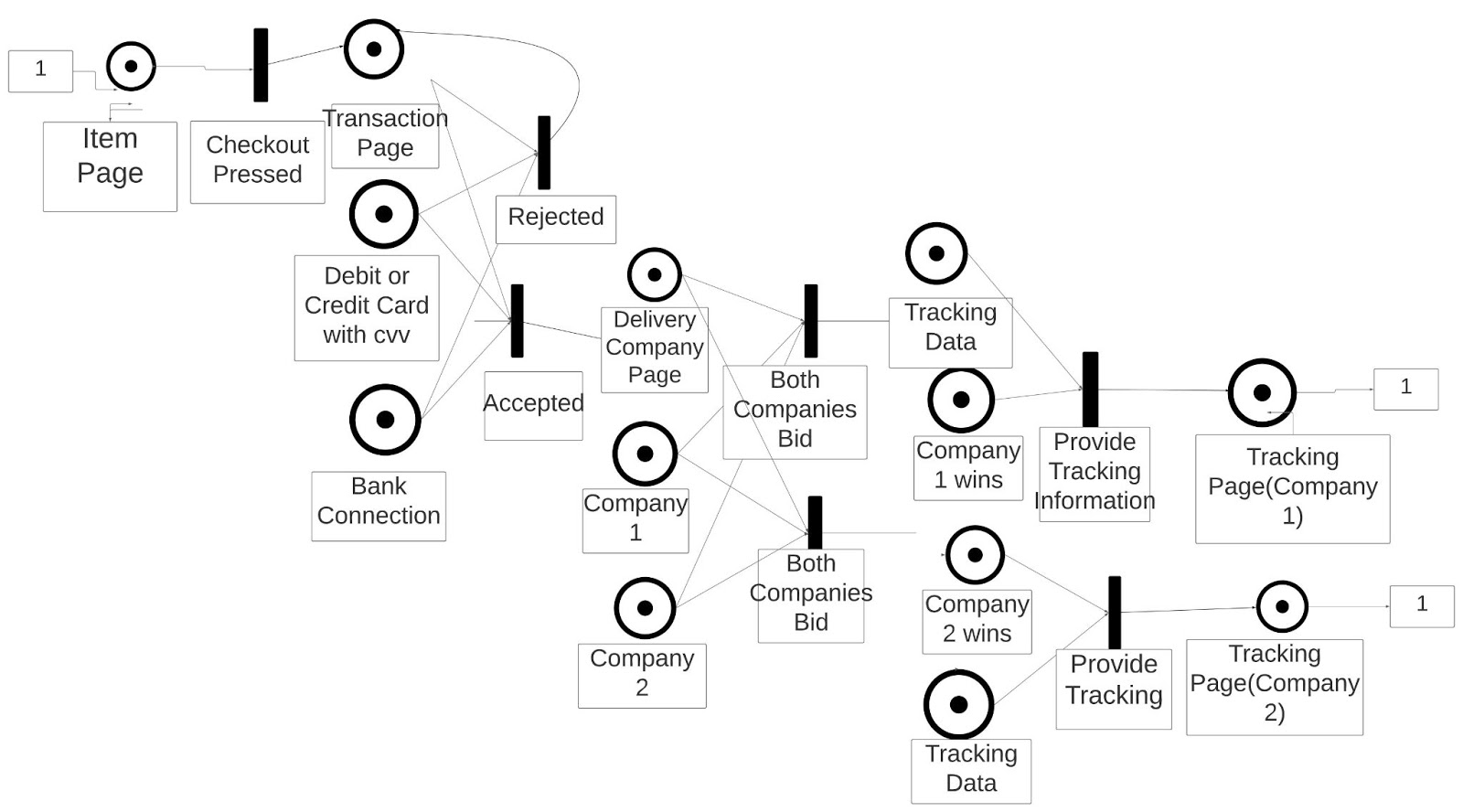


Figure 4: Petri-net for payment and delivery use cases

The payment and delivery feature is represented in the petri net above. The clicking the checkout/payment button, where they will be prompted to enter their payment information. If it is rejected they have to enter again, if its accepted, then the delivery company has to bid on the items. For each item that a company wins, the provide tracking information to the user using tracking data. Then the user is taken back to the home page, after this is done.

## Discussion Forum/Complaint System

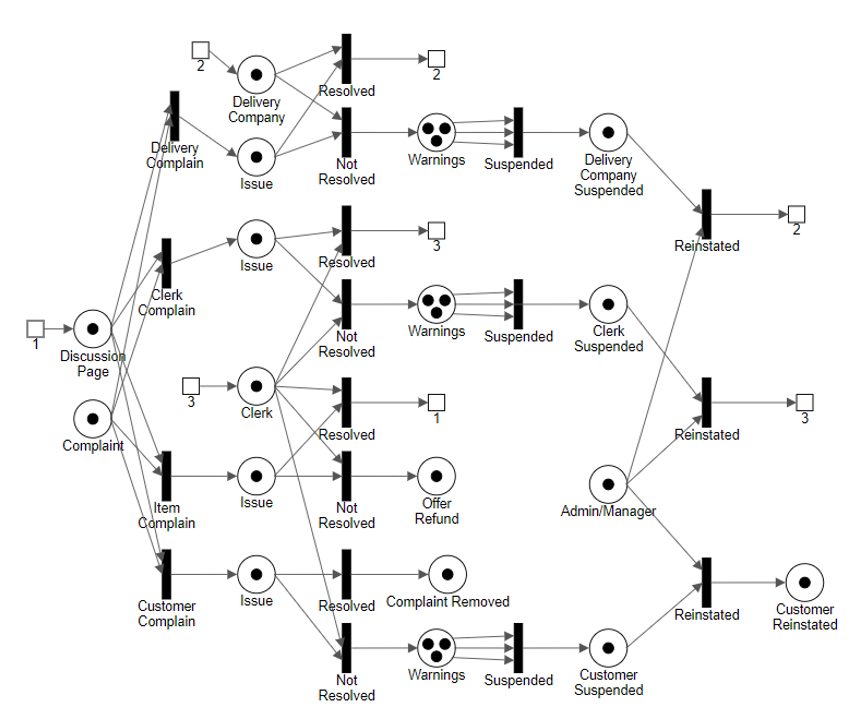


Figure 5: Petri-net for discussion forum/complain system

The discussion forum and complaint system can be represented by the petri-net in figure 5. When a complaint is filed the accused can defend themselves to resolve the issue. If the issue is not resolved it turns into a warning. After 3 warnings the accused is suspended automatically and the admin is notified who can investigate and reinstate the suspended user if deemed appropriate.

## Change Recommended Products

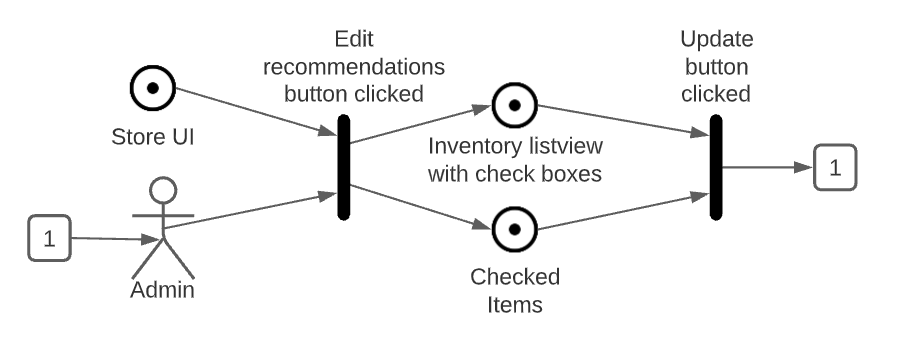


Figure 6: Petri-net for changing recommended products

For the recommended products on the homepage to change, an admin is required. Only the admin will have the privilege to edit the recommended products. There will be a separate UI—a listview widget containing all of the computer parts in the store’s inventory. From there, the admin can select three items and update the system so that those items just selected will appear as the new recommended products.

# E-R Diagram

Diagram

Description automatically generated

Figure 7: E-R diagram of entire computer store system

# Detailed Design

## Create Account

class NewUserForm():

\_\_init\_\_(login\_reference):

# MOST UI CODE IS AUTO-GENERATED

# Member variables

self.login = login\_reference

self.incoming\_user = None

self.username = lineEdit\_username.text()

self/password = lineEdit\_password.text()

self.email = lineEdit\_email.text()

# Connect signals with functions

button\_create.clicked.connect(create\_account(username, password))

def create\_account():

users = load\_users(Users.xml)

customers = users.customers

# check if user already exists by using email and username

for customer in customers:

if (self.username == customer.username or

self.email == customer.email)

print("user already exists")

self.incoming\_user = Customer(username, password, email,

other\_attributes)

# add new customer to customers list

customers.add(incoming\_user)

# serialize customers list into xml

serialize\_customers()

# hide new user form

self.ui.hide()

# show login form

self.login\_reference.show()

## Product Recommendations

def update\_recommendations():

# get checked items from a listview containing all items currently

# in stock

checked\_items = inventory.checkedItems()

i = 0

for checked\_item in checked\_items:

recommendations[i] = checked\_item

## Customer Review

Def customer\_review()

{if(ProductId))

Const review=(

name=req.item.name

id=req.review.id

Rating=number(rating)

comment=req.item.comment)

}

product.review.push(review)

product.save()

print(“”)

Else pring(“”)

Transaction Use Case

Will view the cart, when checkout is pressed

Then will ask for credit card

Function to check payment…

## User Account

Retrieve the xml data on user

Change text to display user name, address, balance, and bank info

By default call cart()

On event clicked cart()

On event clicked history()

On event clicked track()

On event clicked logout()

cart()

Display list of items from user cart data

For each item give options to remove item from cart

If (checkout is clicked)

Move to transaction page

history()

Display list of items from user history data

For each item give option to leave review

track()

Display list of items from user track data

Get track info from delivery data

Display arrival time

logout()

Return to homepage

## Transaction

//this is an if statement that checks if the checkout button is pressed

If(checkoutButton.isPressed() == true) {

CheckoutPayment()

}

//Method to check if payment info works correctly

Def checkoutPayemt(userDebitCardNum):

If(userDebitCardNum.isValid and userDebitCard.balance > totalpriceOfItems){

Print(“Thanks for the payment, we will update you with delivery details}

putinDeliverySystem(listofItems)

deliveryCompanyBid()

} else {

print(Please check your debit card or you don’t have enough money)

//function to initiate the delivery bid

def deliveryCompanyBid() {

deliveryItemsforCompany1 //list to store items for

itemsforCompany2 //list to store chose items

for items in deliverySystem:

deliverycompany1bid

deliverycompany2bid,

store clerk chooses one of the above

if(storeClerk.choice != min(deliverucompanybid1, deliverycompanybid2)

clerk has to explain

if(clerk.explain == NULL){

system.generateCheatingWarning(storeclerk)

}

Then, both delivery companies provide data for the items they will deliver.

# System Screens

This section includes screenshots of various UI pages for various features. They are initial drafts and will be modified in the future.

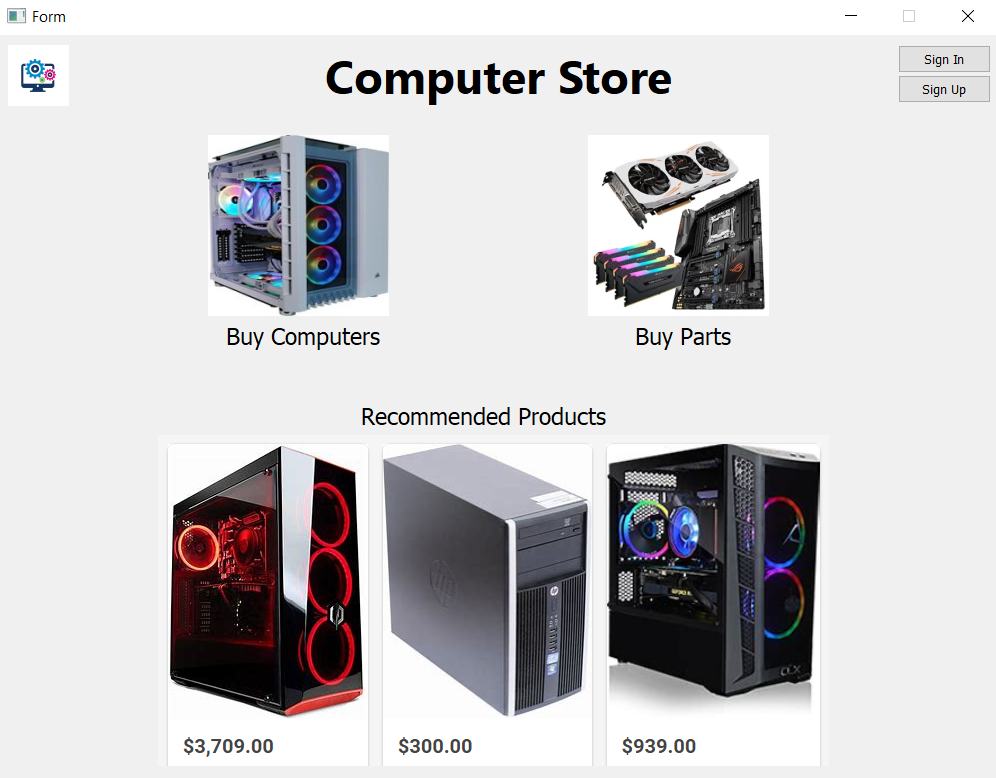


Figure 8: Initial draft of homepage

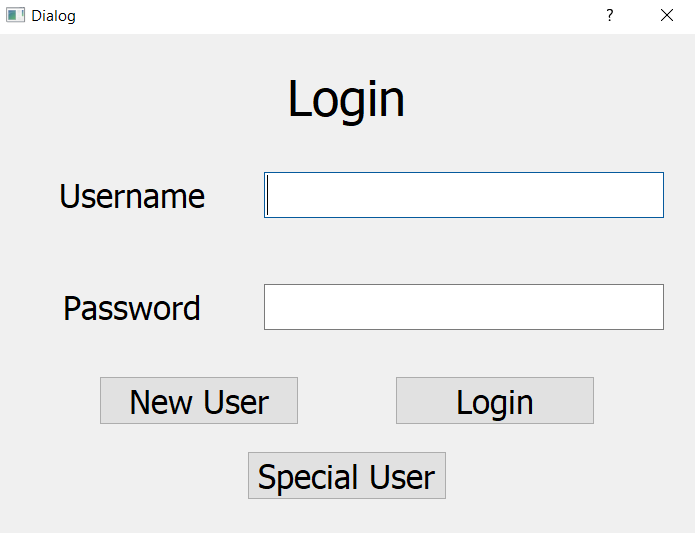


Figure 9: Initial draft of login screen

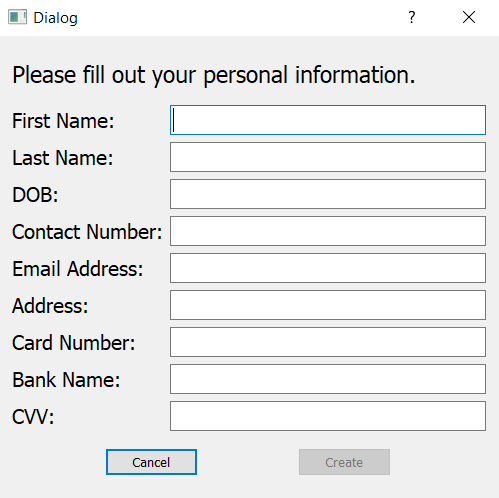


Figure 10: Initial draft of new user form

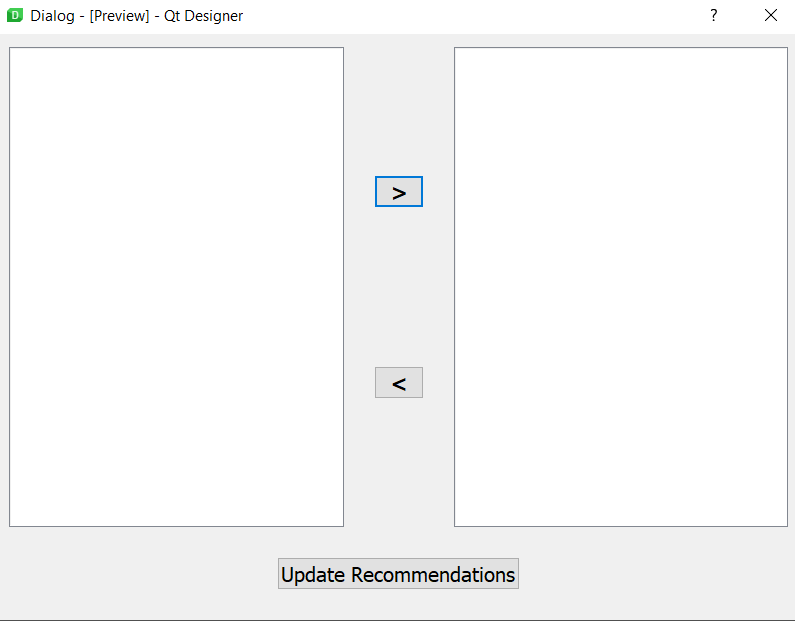


Figure 11: Initial draft of update-recommendations form

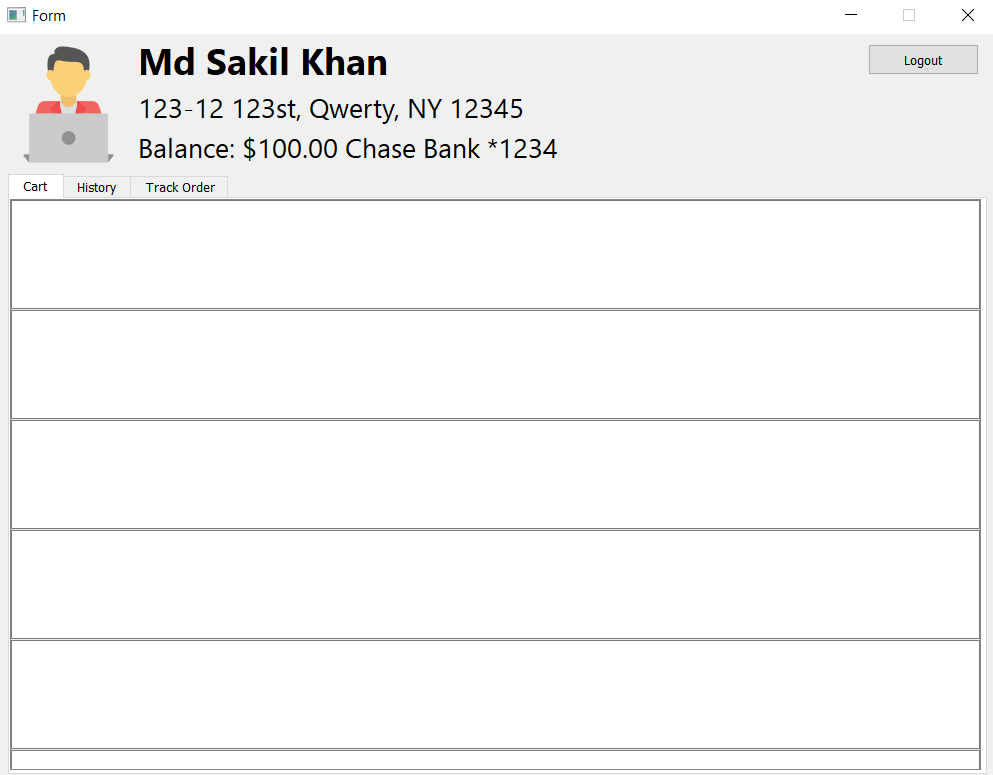


Figure 12: Initial draft of user profile page

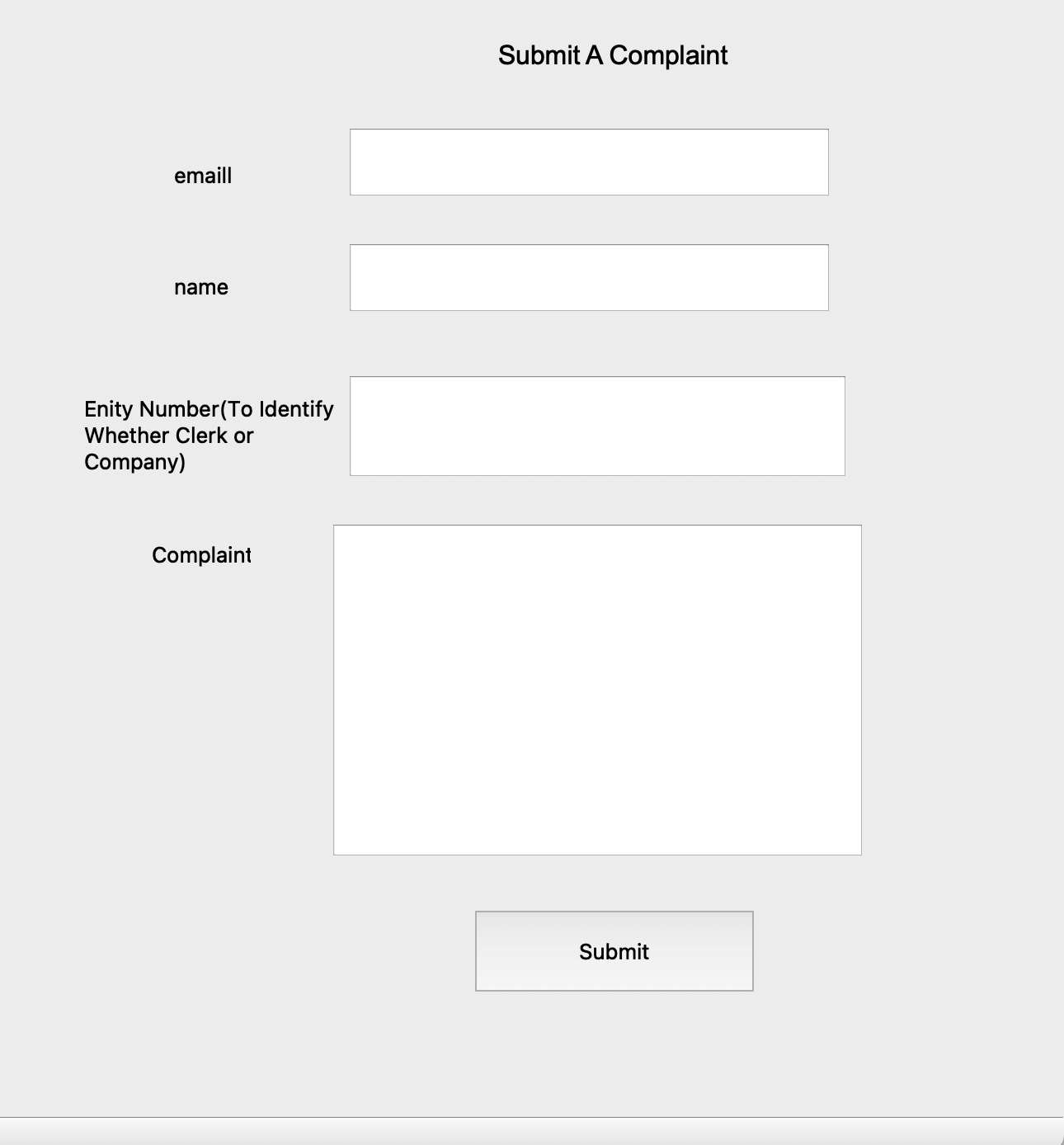


Figure 13: Initial draft of complaint form

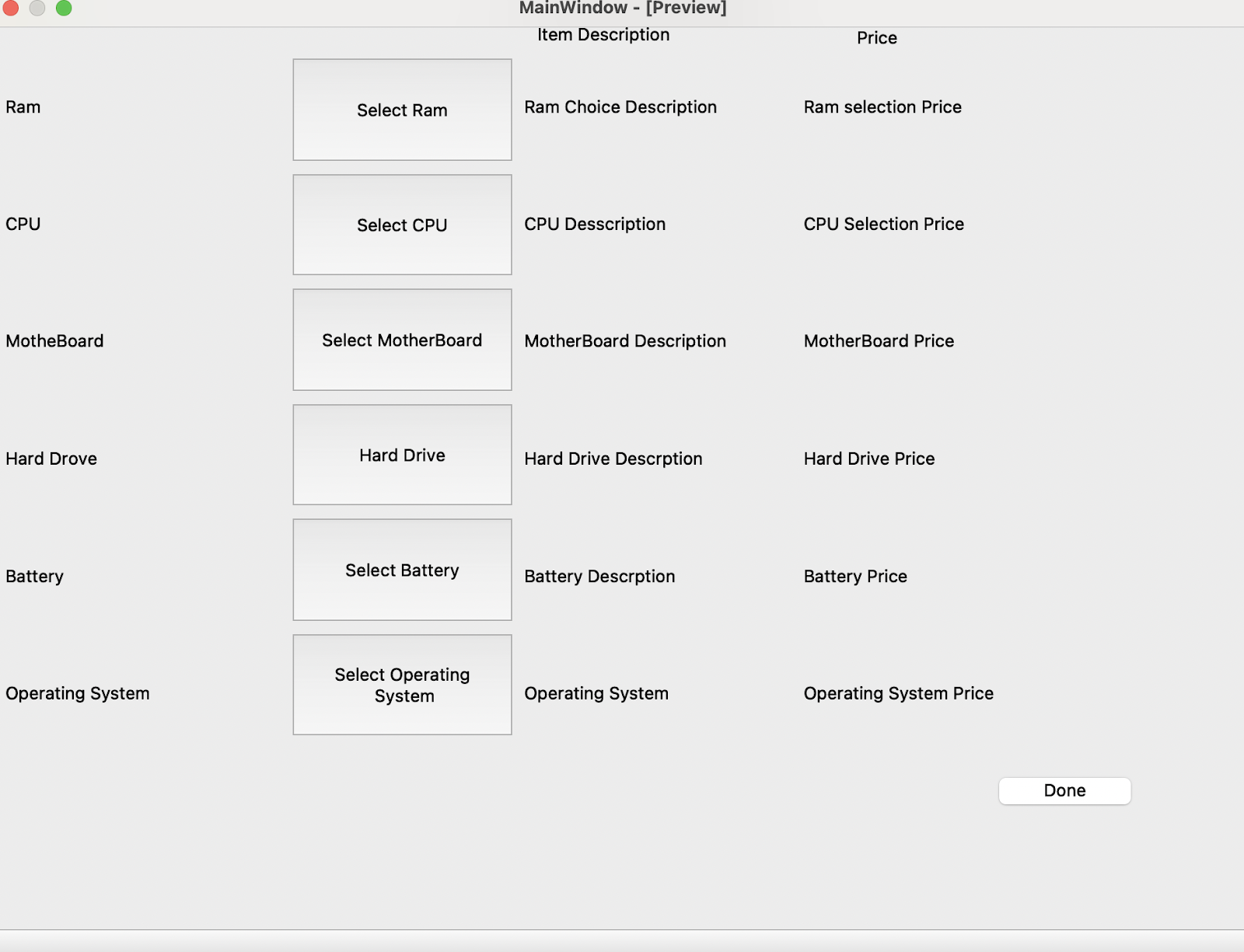


Figure 14: Initial draft of computer parts list

# Minutes

|  |  |
| --- | --- |
| date | log |
| 4/23/2021 | Read the specifications together. Discussed what needs to get completed and how to divide up components of report among team members |
| 4/24/2021 | Joshua & Huihong - Created initial design for login and user registration Joshua - completed Overall Collaboration Class Diagram  Huihong - completed e/r diagram  Sakil - completed overall use case diagram, and login petri-net |
| 4/25/2021 | Sakil - added user account screen |
| 4/26/2021 | Sakil - completed discussion forum petri-net and main screen  Nana - completed transaction petri net |
| 4/27/2021 | Sakil - added user page pseudocode  Nana- Added transaction pseudcode |

# Address of git repo

<https://github.com/jchang005/CSC32200-Project>