Grismely Hiraldo, Andy Gonzalez, Arsenio Pena, Danny Jaramillo
Csc 322
Cook n’ Look Computer Spec Report

Cook n’ Look Computer

We developed an online computer store that minimizes the hassle of shopping for parts or an entire system by offering a minimalist layout and convenience that other computer stores may not have.

Users of the system: visitors/browsers, registered customers, store clerks, store managers, computer parts companies

Main features of the system:

1. **The online store has a homepage showing 3 suggested systems chosen by a store manager and 3 most popular computers per number of sales.**

Although we were unable to properly display the top 3 recommended items from the store manager, as well as the top 3 most sold items in the GUI, the backend portion of this home page was completed. By looking through the list of computer parts, we can simply display the top 3 most bought items and display them on the home page. The 3 recommended items of the store manager can be hard coded inside the backend.

2. **The system provides choices of OS (e.g., win, mac, linux), main purpose (business, scientific computing, or gaming), and architecture (e.g., intel, arm) for the customer to choose from, a new page associated with the choices for different parts, such as cpu, gpu, ram, hard disk, battery, screen, software with possible constraints (a more powerful gpu needs better battery, gaming purpose needs better gpu/screen resolution etc.), and the prices, voting, discussions about each item (if available)**

Our store was solely based on Python and Tkinter, everything was done from scratch. Most web based frameworks contain ways to dynamically add elements into the GUI. One of our biggest challenges was creating a working filtering system that also worked dynamically. By saving responses in a list, we can check if the filtering inputs match with the default arguments. In the browsing page the user is prompted to check off their desired check buttons to find their desired product and their price range, here the filter system checks the users request and searches the product list for the correct items. The elements that did not get implemented for this part of the project was the “Operating System” inside the filtering,the voting to be determined by the manager, and the constraints when selecting several parts inside the cart.

3. **A visitor can browse the listings of the computer and parts and discussion forums**

Functionalities of the visitor have been properly implemented. Visitors or “Guests” only have the ability to view computer parts, and utilize the filtering system in order to browse. Only when the guest signs in, or creates an account, that the registered user now has the ability to make purchases, participate in forums, and complain about completed orders.

4. **A visitor can apply to be a registered customer with a unique working email address, the system maintains a “avoid” list of email addresses, any application whose email address is in this list will be denied and send a denial message to the address, any future applications will be denied without reply.**

Visitors are prompted to login at the top right corner if they don't have an account with us they are prompted to click register that will direct them to a new page that takes their first name, last name, email, password and the same re-entered password. Warning and denial messages will appear if the user is attempting to create an account with an email that's already in use or if the two passwords do not match. Constraints for account creation have been implemented successfully in the backend. A new user must provide a valid email address that is not already in the system, matching passwords, and provide a balance for the account. If any of the above requirements are not met, the guest will be unsuccessful with the account creation.

5. **A registered customer must provide a working credit card or deposit money to the account for possible purchase.**

A customer is allowed to add a balance to an account upon registration and additionally when they click the “account” section it will redirect you to the user’s account page which stores their first name, email, balance, as well as the amount of warnings that they have.

6. **A registered customer can browse the system, make purchase options, search info, browse his/her own private purchasing history/expenses, comments and (start) votes on the items s/he purchased already. A customer can buy one whole computer or just a part, and can discuss with other customers or the store clerks via the forum.**

A customer can browse and navigate through the system via a scrollbar, they are given the option of adding to cart through a button.When the button is pressed the cart logo on the top right corner automatically updates to (x), signifying however many items have been appended into the cart. Backend code was provided that gives the user the ability to comment about a type of product per section but was not applied to the GUI. We were unable to provide an orders page. Any information available on the text file was linked to the GUI, the users name and their message. In the backend, once a comment is placed it is first looked through for “bad words” the bad words were predetermined in a list, where the most common curse words we believed would possibly be entered are present and checked for. Then the comment is placed in a text file and organized as “comment” | “user’s name”. After being placed inside the text file it is displayed on to the screen, where the user can tap through and see each comment.

7. **At the submission of a purchase decision, the amount will be checked against the customer's account or credit, if not enough money, the submission is returned with a warning message. If ok, the amount will be charged. The system then puts the purchase to the delivery subsystem.**

On the GUI, successful purchases have not been implemented. On the backend, when a user performs the checkout, the total for the items are first totaled, then checked if the users balance - the total is zero or a positive number. If either conditions are met, then the purchase was successful. If not, the user is then prompted with a message, "Not enough balance".

8. **There are at least 2 delivery companies that will bid on each item available in the delivery subsystem, a store clerk chooses one based on the bidding. If the winning company's bidding price is not the lowest, the clerk should provide justifications about her/his choice, otherwise the system will generate a warning on the clerk for possible cheating and shown to the manager.**

Unfortunately, implementing the delivery companies and tracking information has not been finished. The discussed implementation for the bidding system was after a shipping company logged in their main screen would hold orders to bid on. After entering their bid amount, the store clerk would see the order with the registered user's information and the shipping cost specified by the shipping company. The clerk would then have a button for each company and if the amount was not the lowest a popup text box would appear on the screen where the clerk would enter their justification. The enter button would save their response and close the textbox. The information entered in the textbox would then be presented to the manager where if it was empty or the justification was simply not acceptable to them, they could then act accordingly.

9. **The delivery company should provide tracking information for the customer to know its whereabouts.**

Delivery company and tracking information were not implemented, our approach would've been to have a delivery column in the orders section where you view the orders that you placed before. In this section the delivery column would provide a tracking number that would be randomly generated by a backend function in correspondence to whatever delivery company was selected. For example, our three different companies: FedEx, UPS, and USPS would have distinct beginnings to the tracking numbers; they would be differentiated by the starting letter "F" , "U" or "US."

10. **A registered customer has an account in the system with information such as available money/credit, home address, purchased computer/items history, and complaints s/he received and filed, votes s/he casted.**

This information is stored in the "account" page when clicked it will prompt the user to a page storing their first name, last name, email, available money/credit stored as balance, as well as the amount of warnings/complaints that they have received. We were not able to implement an order history, votes casted or home address.

11. **A registered customer can complain about the purchased items, clerks and delivery companies s/he dealt with, which are available to the store managers and the complained clerks/companies/delivery, who should counter with their side of info to clear the warning, the manager can decide to let the warning stay or removed and informing all parties with his/her justifications. A clerk, computer and delivery company that received 3 standing warnings is suspended by the system automatically, the parts of the suspending computer companies and the bidding right of the delivery companies are suspended from the store as well. A customer whose complaint is reversed will receive one warning. A thrice warned customer is removed from the system and put in the "avoid" list. The store manager has the power to remove any customer and clerk with justifications, even with less than 3 warnings. A suspended customer will be informed by email and given the last chance to clean up his/her account.**

Customers that have had complaints filed against them will have the amount displayed in the accounts section, labeled as "Warnings." Backend code has a conditional statement that assures warnings are below 3 as 3 leads to suspension. Managers are given the ability to add warnings, as well as suspend existing user accounts. Functionalities that include customers being able to complain about completed purchases, or the people they have dealt with (Clerks/Delivery Companies) have not been completed. On the GUI, we were unable to display the suspended accounts, as well as the accounts with a specific number of warnings, but the functionality on the backend is complete.

12. **In the discussions forum, customers can complain about other customers if they find violating words/sentences/attitude which will be again determined by the manager to stay or reverse as Item 11, or the customer used words that are in the "taboo" list maintained by the clerk/manager, one such violation will automatically generate one warning and the violating words are redacted by *.**

Backend code was developed that "filters" or censors inappropriate words or comments by users, once a user makes a post when they click "submit" the post is first passed through the censoring function where a word bank of "bad words" is kept if any of the words the words in the wordbank match the words in the user's comment they are replaced with "*" and THEN comment is added to the forums text file and uploaded to the GUI.

13. **Each team must have one creative feature counted as 10% of the final system to showcase your idea on what ideal feature this store should have. A *super creative* feature will receive a complimentary bonus (but <=10%).**

For the creative idea, despite lacking the implementation we were drawn between a scavenger hunt, a "rig of the day", and a budget system. The scavenger hunt would be similar to the physical version where puzzles would be laid out throughout the website, and upon completion a registered user would receive a coupon code when checking out. "Rig of the day" would be similar to the way store managers would put up recommended computers but instead an algorithm would decide a recommended rig with each part needed to make it. The final one is a budget system, where a user would input their budget and an algorithm would create various computers based on their needs (whether for gaming, business, school, etc.).

Acknowledgments:

Andy Gonzalez oversaw the GUI and contributed to the filtering system. **Grismely Hiraldo** contributed to the forum and login system; **Danny Jaramillo** contributed to filtering and forum development and **Arsenio Pena** was responsible for the filtering and checkout system.

Repo:
https://github.com/djara1214/Csc322-Project.git