|  |  |  |  |
| --- | --- | --- | --- |
| **Student’s Name:** | Mitchell McKeown | **Date:** |  |
| **Lecturer’s Name:** | Asmat Ullah | **Course/Class:** | HNC/SD/A |
| **Unit Title:** | SD: Developing Small Scale Standalone Applications | **Unit Number:** | H17W 34 |
| **Outcome/s** | Outcome 2, 3, 4 | **Assessment Task:** |  |

**After your assessment has been marked, you will be able to read your feedback and advice on how to prepare for re-assessment on the reverse side of this sheet.**

|  |
| --- |
| **The work submitted in this assessment is entirely my own work**  **Signature Declaration ......................**Mitchell McKeown**........................................** |

|  |
| --- |
| **DESCRIPTION OF ASSESSMENT AND INSTRUCTIONS**  **This assessment requires you to produce a Java desktop application.**  **ASSESSMENT CONDITIONS**  **This assessment is open book.**  This assessment can be undertaken either within or out with the classroom. Once the project is completed students should be prepared to answer questions and provide a code walk through of their application, verifying the solution they have developed is their own original work. |

**This Section is for Staff Use only:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Result** | **Initials** | **Opportunity\*** |
|  |  |  |  |

\*Insert opportunity 1 or 2. A 3rd opportunity may be available only if the student applies in writing to the Curriculum Leader explaining the exceptional circumstances – however, this may not always be possible.

**Before re-sitting this assessment you should follow the following Remediation Advice.**

**Feedback on the attached assessment to help you in your course**

|  |
| --- |
| ***Signature of Lecturer ........................................... Date............................................*** |

**Assessment task 3**

**Outcomes covered 2, 3 and 4**

**Assessment task instructions**

**Open-book assessment — Develop a desktop application**

In this task you are expected to develop, test and deploy a relatively simple desktop application.

You should produce a user interface design that illustrates how the application is expected to behave.

The application developed should:

* include at least two event handlers
* use at least two standard libraries
* use a range of operators
* use selection and/or repetition constructs as appropriate
* include appropriate internal documentation (comments and naming conventions)

You are also required to design a basic usability test for the developed application.

The test should be undertaken by a minimum of **three users** and should include a minimum of **five** **appropriate questions**.

You should analyse the results of the usability test and produce some recommendations on how the application could be improved.

You are expected to hand in:

* the user interface design.
* the code produced.
* evidence of the app working (eg video or screenshots)
* the usability test.
* recommendations based on results of the usability test.

EPOS –electronic point of sale.

You are a software developer with an entrepreneurial flair and decide that you would like to build a business in providing EPOS solutions.

You will build a small scale standalone working prototype EPOS desktop application that you will present to a panel of investors in order to gain funding to develop a full featured application.   
The application you develop will contain enough EPOS features to demonstrate your technical abilities in engineering EPOS software.

## EPOS – The background

Retail customer service agents use computerised electronic tills that are equipped with a Java Virtual Machine. A Java forms EPOS application can be installed on this machine.

There will be two types of user roles for your application.

1. Customer service agents – referred to as **Sales** **Agent**
2. Administration managers– referred to as **Manager**

The application will contain the following:

## Security Features.

1. A Login screen will be used to authenticate the user. This will be displayed on start-up of the application. Only valid authenticated users will be allowed entry to the system.

## All User Features.

1. Users will be able to view their account details.
2. Users will be able to view the following reports on their sales
3. Total customers served, Average sale cost, Minimum sales cost, Maximum sale cost, Total sales takings.

Manager Features.

1. Only Managers of the system should be able to create, update or delete users/products.
2. Managers will be able to restart the system. This will delete all sales data from the database.

## Sales Features.

1. All logged in sales agents or managers will be able to select products and checkout.
2. A **minimum of 9** products will be offered for sale/ordering.
3. A running total of the order will be displayed on a section of the screen.
4. A checkout button will be used to complete the order.
5. Orders can be cancelled by the agent.

Only the final sale value will be saved for the logged in sales agent.

The following algorithms can be used to implement the minimum and maximum cost methods.

|  |
| --- |
| **Design Algorithm CalcLowest** Set the form variable lowest to first value in the global arrayList  For curindex=1 up to count-1  If value at index curindex is less than lowest Set lowest to value at index curindex  End if End loop |
| **Design Algorithm CalcHighest** Set the form variable highest to first value in the global arrayList  For curindex=1 up to count-1  If value at index curindex is greater than highest Set highest to value at index curindex  End if  End loop |

**Stage 1: Design** the application screens using mock up software and present this to the EPOS product owner (your lecturer) for approval.

**Stage 2**: Write the user stories (requirements) for this app.

**Stage 3:** **Create** the application.

**Stage 4**: Create and run test cases that prove your application as working. These should include full screenshots of your working application.

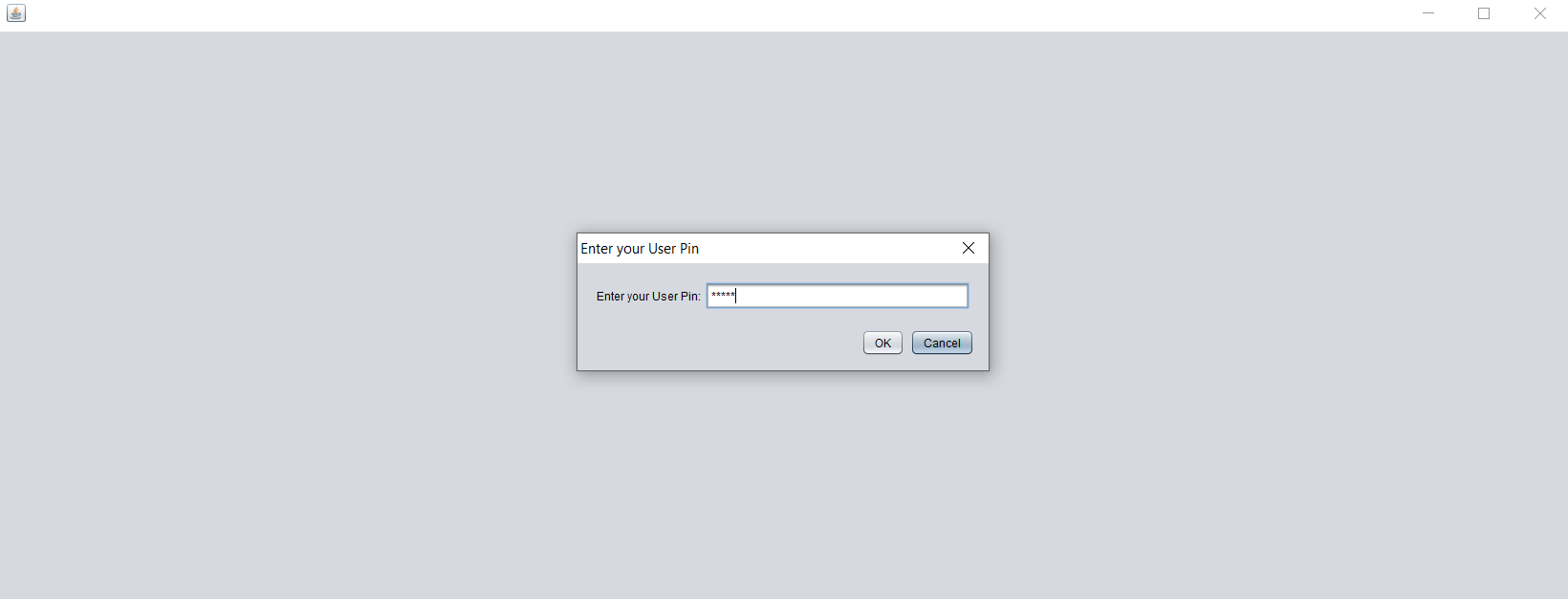
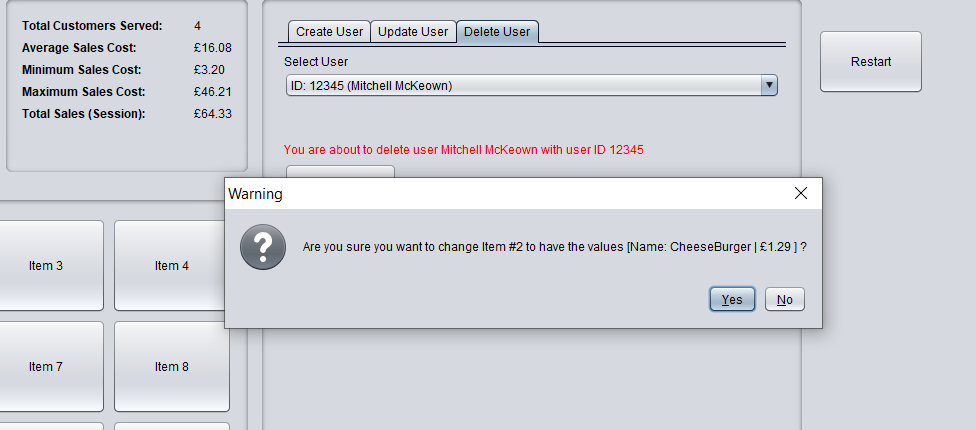
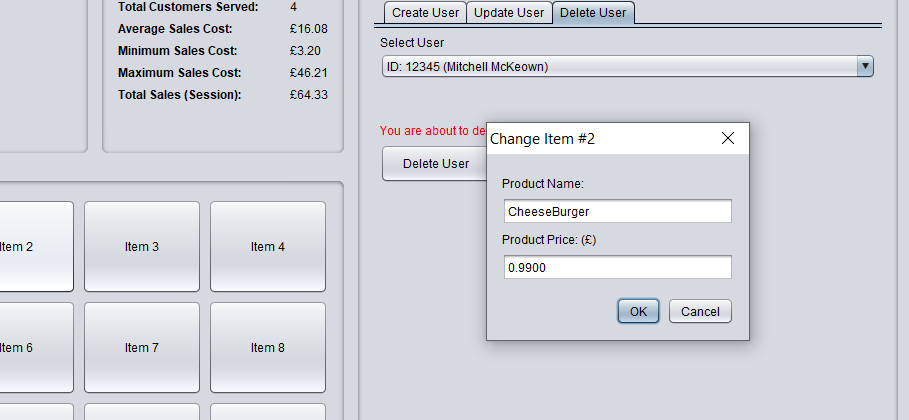
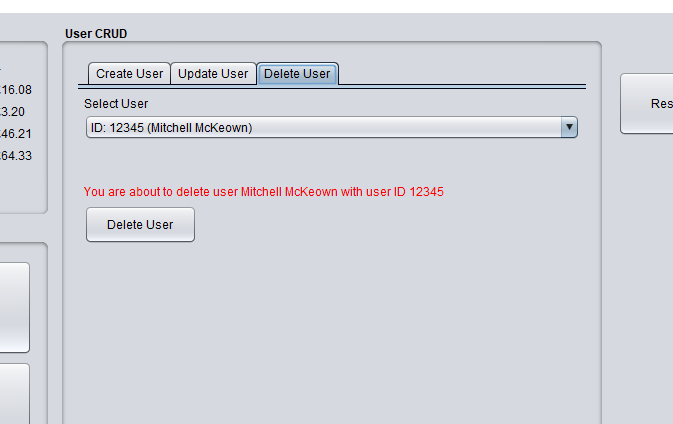
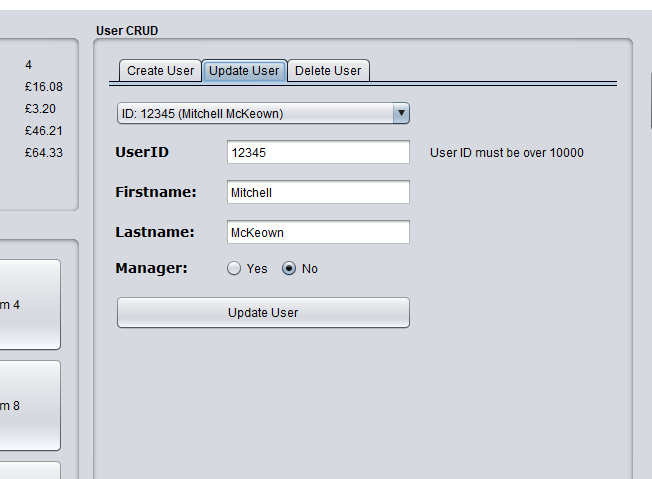
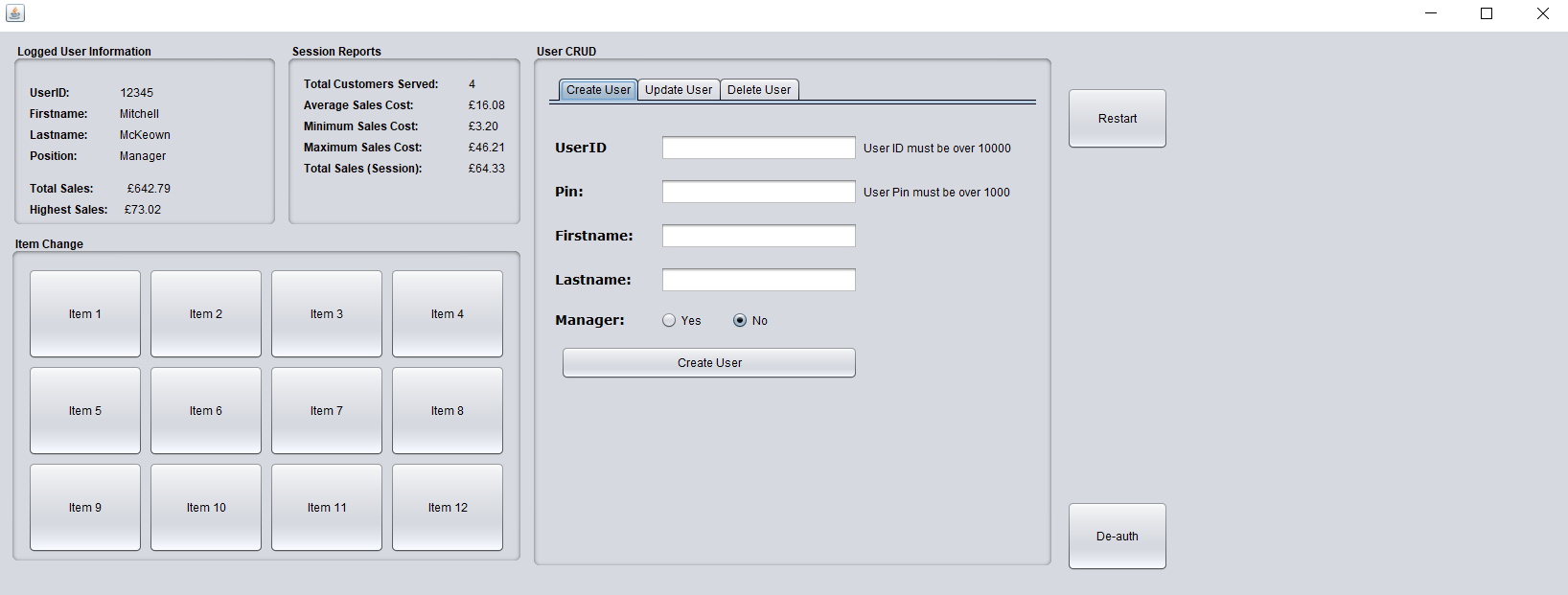
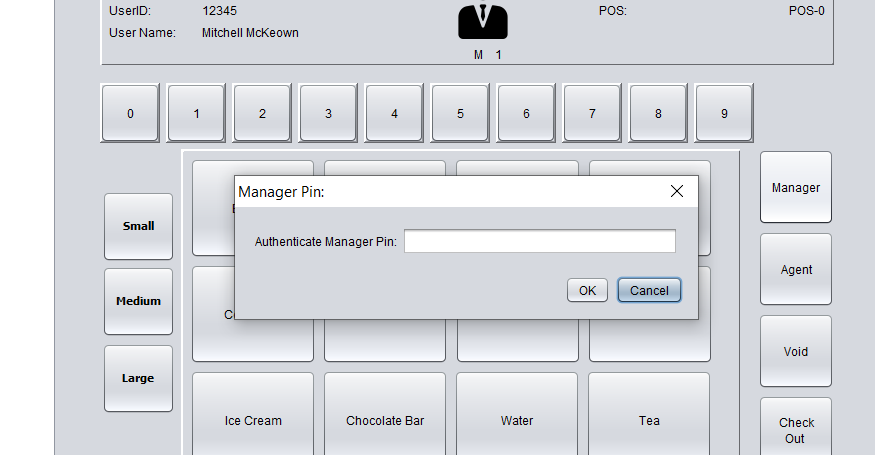
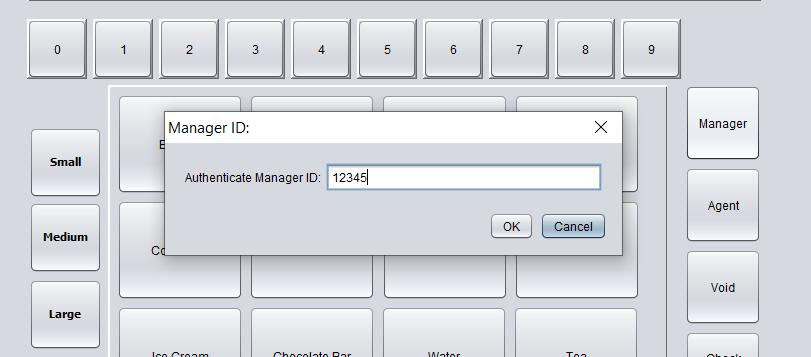
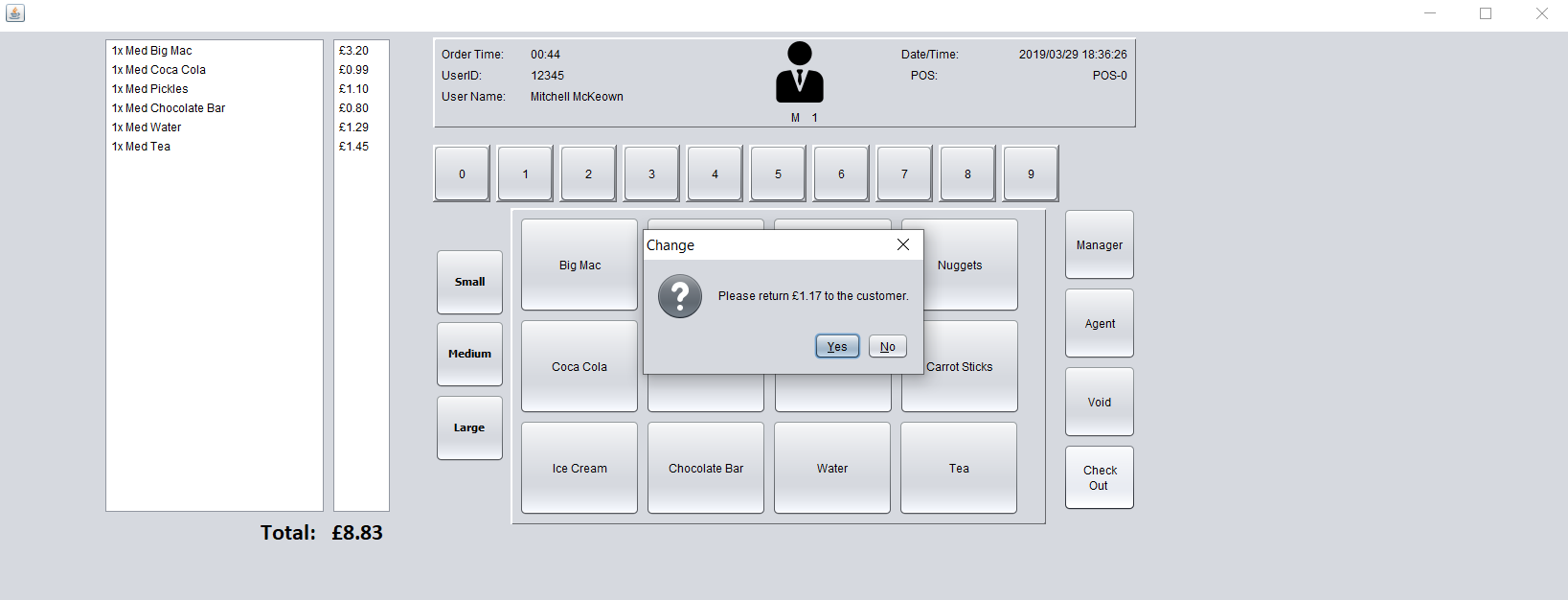
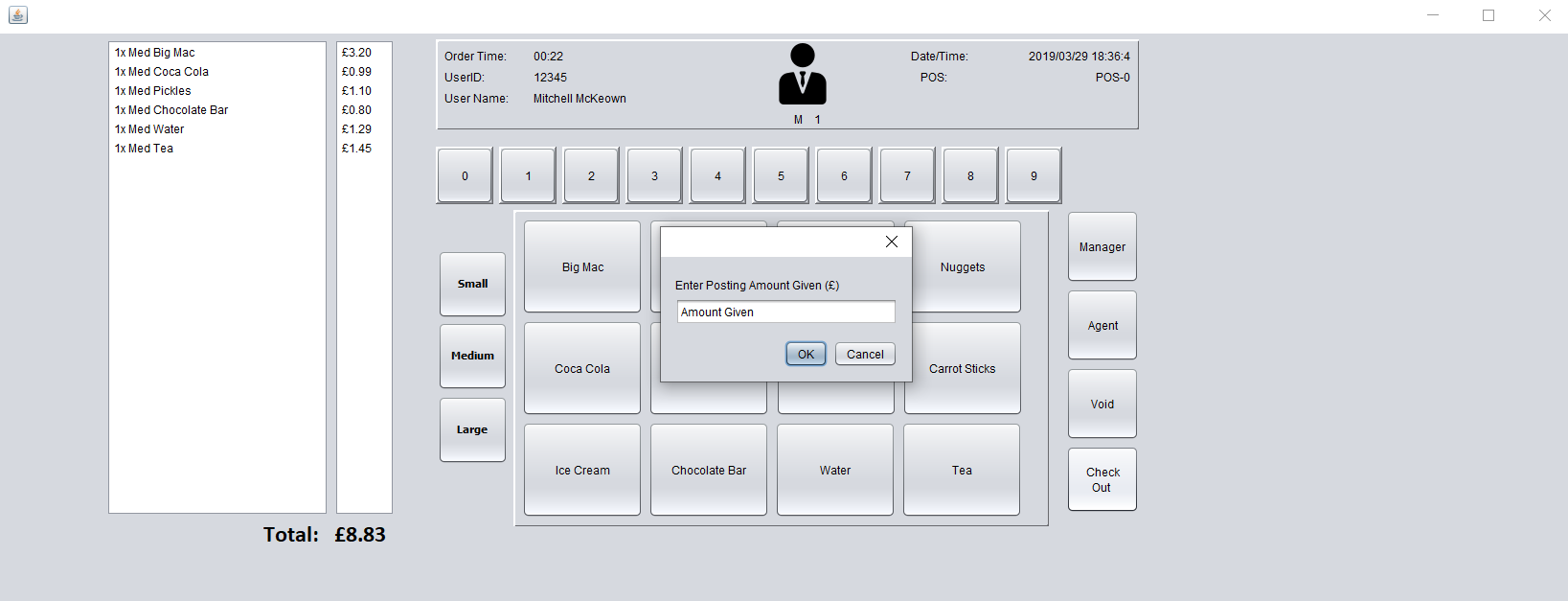
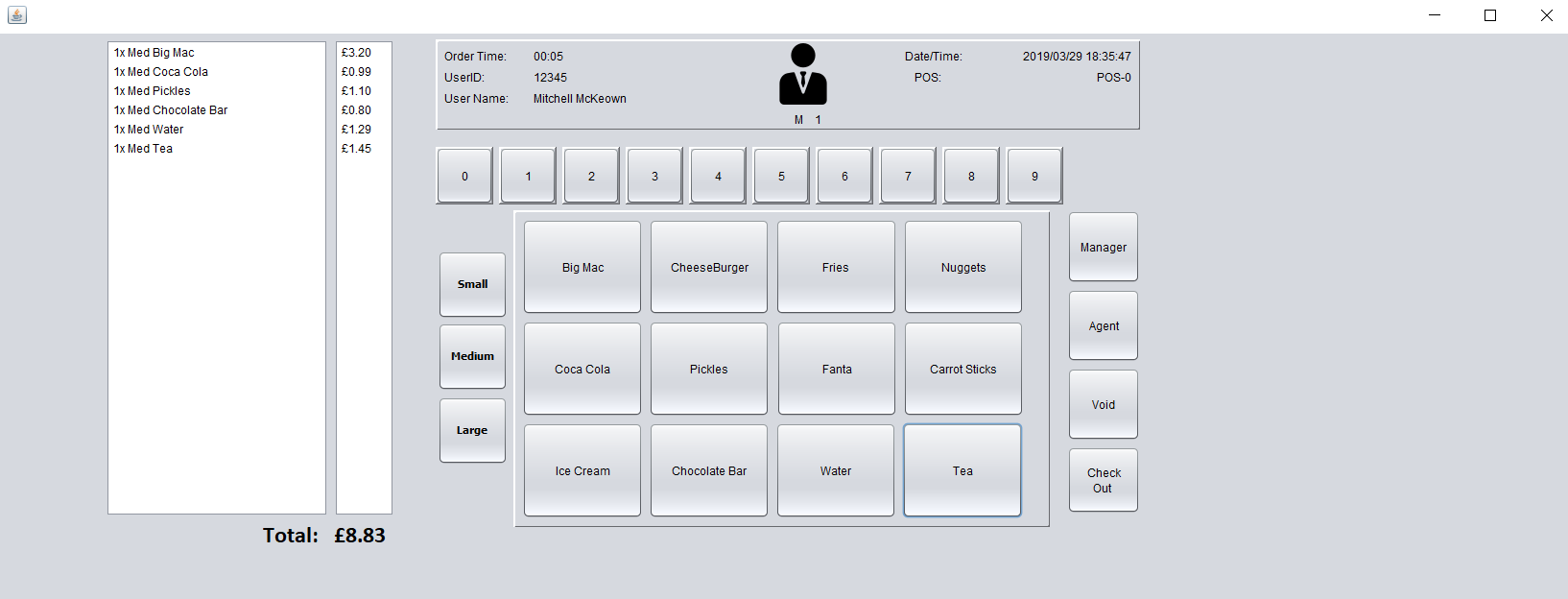
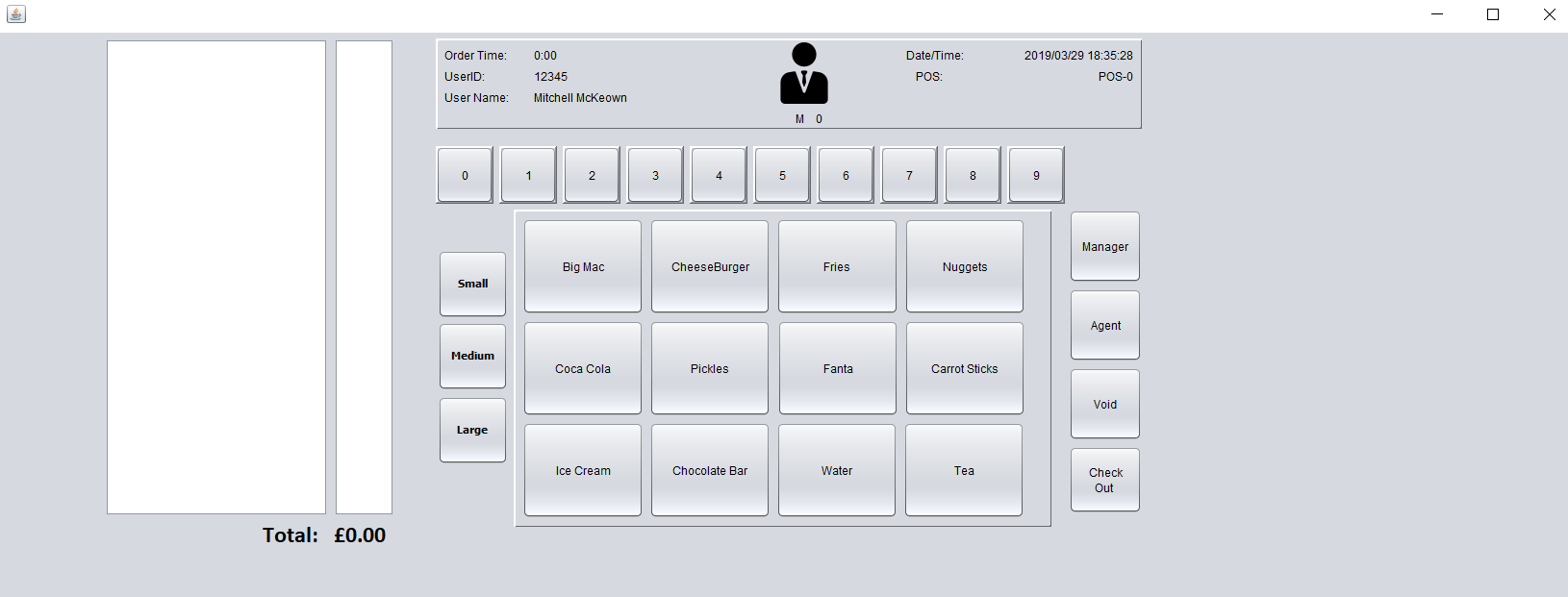
**Stage 5: Test and Deploy** –   
Create a distribution folder – ensure that your application runs **independently** of the NetBeans environment.   
  
**Stage 6:** Run a usability test with **at least 3 users**. Summarise and present your findings in a report.

**Appendix 1**

**Use the following table to create your test document – These test case should test the following features as a minimum. If you have implemented more than the following, then produces cases for those features too.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOFTWARE DEVELOPMENT** | | | | |
| **Test Data** | | | | |
| **No** | **User Action** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| 1 | User Successful Login | It is expected that upon entering a successful username and password, the user logs in and the manager items are inaccessible until authentication. | Upon entering successful credentials, the user was able to log in without access to manager details. | Pass. |
| 1.a | Appended User Data. | On Successful login, user data should be appended to all appropriate panels. | Data is successfully appended. | Pass. |
| 2 | Add Item | If a user presses one of the activation items, it should successfully pass the data to the viewable box (including the price) and update the total. | Upon pressing an item, it parses the data through successfully and updates the price. | Pass. |
| 3 | Remove Item | If a user has an item added, clicking on it and pressing “Void” should remove that item from the cart and successfully deduct the price. | If an item is pressed and “voided”, it successfully gets removed from the cart and the price is updated. If the item is the last item in the list, then voiding that cancels the order. | |
| 4 | Multiplying Items | If the user clicks one of the corresponding item amounts (1 – 9) then clicks on an item, it should multiply that item and value before appending it to the cart. | Items that are multiplied are successfully added to the cart with updated data. | |
| 5 | Sized Items | In this EPOS system, you have the option to have an item “Small”, “Medium” or “Large”. Each with their own price. Clicking the “Large” button, then an item, should append updated information to the table including multiplied items if applicable. | Clicking “3”, “Large”, “Item X” successfully appends 3 items, with “Large” characteristics and updates the price. | |
| 6 | Checking Out Cart. | Upon checking out the system should prompt for a posting amount (I.e. how much the customer has handed over) and do calculations based on the total price to return how much change should be given. | The system successfully hands out the correct change based on an order. | |
| 7 | Agent Menu | When pressing the Agent Menu button, the system should return you to the menu in which it posts data about the session and history of that user. | In the Agent Menu, it provides appropriate information regarding the users sales and information. The sales and data are re-calculated on each access of the menu. | |
| 8 | Manager Authentication | Since we may expect a manager to log in during a user session, the manager button is accessible by all. When clicking the manager button, you must authenticate with manager credentials. Any user with the manager flag “0” can not access this menu. | In the condition where the user flag is 0, the access to the manager menu is denied. If the user flag is 1, the access to the manager menu is granted. | |
| 9 | Parsing manager information | Upon entering the manager menu, information regarding the session and history of the logged user should be posted. | Information regarding the session and the history of the current logged user is posted to the manger menu. | |
| 10 | Manager CRUD Functions | In the management menu, there are multiple CRUD options for both user and items. These menus should be visible to the manager on successful authentication. | The menus post and show all CRUD options for the EPOS System. | |
| 10a. | Create User | The manager should be able to create users through this CRUD panel. Upon entering required information including formatting (i.e. numbers for ID/pass), the user should be successfully created and posted into the database. | Upon adding a new user “Test Data” with ID “11223” and pass “12345” and the manager flag set to “1 (Yes)” – the user was successfully added to the database with a hashed passcode. | |
| 10b | Update a User | All users in the database should be appended to the box at the top of the Update User session. Upon changing the first-name of “Test Data” to “Functionality” – it should automatically update the database with this new information. Data for that user should also be automatically appended to the fields. | The user selected is automatically updated and the new data is sent to the stored in the database. | |
| 10c | Delete User | Selecting a user to delete should parse the data to the warning sign above the “Delete Button”. If “Functionality Test” is selected and the delete button is pressed, their data should be wiped from the database. | The user data is successfully deleted from the database. | |
| 10d | Update Item | If an item box is selected to be updated, the current information stored within that item index should be shown and the manager should be able to promptly change it. Upon it being changed, the database is updated and the main EPOS system should also update to accommodate the change of price/item. | Upon changing the name “Big Mac” which is priced at “£3.20” to “Test Mac” priced at “£4.00” – the database and EPOS system was updated with the new data on that index. | |
| 10e | Restart System | The system should be expected to end the current Session and log out all users on the EPOS System. A new SessionID is given to the next logged in user so the information such as average sales, customers served etc are fresh and contain no data from the last session. | Upon a manager pressing “Restart”, the system logs the user out and ends the current session. | |
| 10f | De-Auth | Alternatively, a manager can exit the menu by de-authenticating the access which will ask for authentication on the event that the manager menu is accessed again. | If the de-auth button is pressed, the user is logged out and a further attempt to access the manager menu asks for another log in. | |

**Screenshots of Program Working**

****

|  |  |  |  |
| --- | --- | --- | --- |
| **SOFTWARE DEVELOPMENT** | | | |
| **ERROR LOG** | | | |
| **ERROR NO** | **ERROR TYPE** | **ERROR (SCREENSHOT)** | **SOLUTION** |
|  |  |  |  |
|  |  |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Candidate Assessment Record**  **Assessment 2** | | | | | |
| **SD: Developing Small Scale Standalone Applications** | | | | | |
| **Student Name** |  | | **Class** | |  |
|  | | | | | |
|  | | **Satisfactory/ Unsatisfactory** | | **Comments** | |
|  | |  | |  | |
|  | |  | |
|  | |  | |
|  | |  | |
| **Assessors Signature:** | | **Date:** | | | |