*Process MeNtOR 3.o*

*Uni-SEP*

CoinMaster

**Design Document**

**Deliverable 3**

|  |  |
| --- | --- |
| Version: | 0. |
| Print Date: | Mar.31 |
| Release Date: |  |
| Release State: | Initial |
| Approval State: | Draft |
| Approved by: |  |
| Prepared by: | CS2212 Group 11 |
| Reviewed by: |  |
| Path Name: |  |
| File Name: | Group11-SDD-CS2212B.doc |
| Document No: | 3 |

# Contents

1 Test Cases 3

2 Group Meeting Logs 5

# Test Case

|  |  |
| --- | --- |
| **Test ID** | 01 |
| **Category** | Login Evaluation & Credential DB Connection |
| **Requirements Coverage** | UC1 – The user logs into the system |
| **Initial Condition** | The software has been initiated and the login interface has shown. |
| **Procedure** | 1. The user enters a valid user name  2. The user enters the corresponding valid password  3. The user clicks *Submit!* |
| **Expected Outcome** | The software will connect the DB and successfully verify the credentials. The user can successfully log in and the main UI will show. |
| **Notes** | All upper-case characters will automatically be converted to lower-case characters when tested against the record in the DB. |

|  |  |
| --- | --- |
| **Test ID** | 02 |
| **Category** | Adding a Trading Broker |
| **Requirements Coverage** | UC2. Adding and Removing a Trading Broker |
| **Initial Condition** | The user has successfully logged in and the main UI is presented. The default row has been filled. |
| **Procedure** | 1. The user clicks *Add Row*  2. A second fillable row shows up.  3. The user enters a broker name that has not been previously entered.  3. The user enters a list of valid cryptocurrency abbreviations separated by commas.  4. The user selects a strategy from the drop-down list of strategies. |
| **Expected Outcome** | All verifications are successful.  A new row with corresponding information shows up in the main UI. |
| **Notes** | Broker name is verified against DB and cryptocurrency’s names is verified against the list fetched from online. |

|  |  |
| --- | --- |
| **Test ID** | 03 |
| **Category** | Performing Trade |
| **Requirements Coverage** | UC3. Performing Trading |
| **Initial Condition** | The user has logged in. The main UI is in display. There is a least one broker entry with a not none strategy been selected. |
| **Procedure** | 1. Select the broker.  2. Select the strategy in the drop-down list.  3. Click *Perform Trade* |
| **Expected Outcome** | The program can correctly evaluate the conditions in the selected strategy by fetching real-time prices. If the conditions were not met, the program will return a message indicating failed transaction. If the conditions were met, the program will return a message indicating the transaction has been successful. |
| **Notes** | If the program cannot successfully fetch the price from the internet (internet connection issue or the cryptocurrency website is offline), the program will return a connection error message. |

|  |  |
| --- | --- |
| **Test ID** | 04 |
| **Category** | Displaying the Trading Action for All Trading Clients |
| **Requirements Coverage** | UC3. Performing Trading  UC4. Displaying the Trading Action for All Trading Clients |
| **Initial Condition** | The user has logged in. The main UI is in display. There is a least one transaction been successfully processed. |
| **Procedure** | 1. Select the broker.  2. Select the strategy in the drop-down list.  3. Click *Perform Trade*  4. Observe how *Trade Actions* and *Actions Performed* *by Traders So Far* have changed. |
| **Expected Outcome** | The new transaction has been correctly shown in the *Trade Actions,* and the *Actions Performed* *by Traders So Far has changed accordingly.* |
| **Notes** | N/A |

# Group Meeting Logs

## Project Backlog

|  |  |
| --- | --- |
| Backlog item | Estimate(hours) |
| Structure Design | 2 |
| Design Pattern Application | 4 |
| Login System Design and Coding | 4 |
| Use Case 2 Pattern Design | 2 |
| Use Case 2 Pattern Implementation | 10 |
| Use Case 3 Pattern Design | 2 |
| Use Case 3 Pattern Implementation | 14 |
| Use Case 4 Pattern Design | 2 |
| Use Case 4 Pattern Implementation | 8 |
| Create Necessary Utility Classes | 6 |
| Overall Debug and Optimization | 4 |
| Test Cases | 2 |
| Group Meeting Logs | 2 |

## Sprint Backlog

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tasks | Mar.27 | Mar.28 | Mar.29 | Mar. 30 | Mar. 31 | Apr.2 | Apr.3 | Apr.4 |
| Structure Design | 2 |  |  |  |  |  |  |  |
| Design Pattern Application | 3 | 1 |  |  |  |  |  |  |
| Login System Design and Coding |  | 3 | 1 |  |  |  |  |  |
| Use Case 2 Pattern Design |  | 2 |  |  |  |  |  |  |
| Use Case 2 Pattern Implementation |  |  | 6 | 3 | 1 |  |  |  |
| Use Case 3 Pattern Design |  |  |  | 2 |  |  |  |  |
| Use Case 3 Pattern Implementation |  |  |  | 4 | 8 | 2 |  |  |
| Use Case 4 Pattern Design |  |  |  |  |  | 2 |  |  |
| Use Case 4 Pattern Implementation |  |  |  |  |  | 6 | 2 |  |
| Create Necessary Utility Classes |  | 1 | 2 | 1 | 2 | 2 |  |  |
| Overall Optimization and Commenting |  |  |  |  |  |  | 4 |  |
| Test Cases |  |  |  |  |  |  | 1 | 1 |
| Group Meeting Logs |  |  |  |  |  |  |  | 2 |

## Meeting Logs

Sihui He: SH

Ziyuan Li: ZL

Yuhan Zhang: YZ

Mingkai Yang: MY

|  |  |  |
| --- | --- | --- |
| **Present Group Members** | **Meeting Date** | **Issues Discussed / Resolved** |
| All members | Mar. 27th  (6 hours) | Structure Design & Design Pattern Application, and Assignment splitting |
| All members | Mar. 29th  (8 hours) | Use Case 2 Pattern Implementation, Login System Coding review and revision, UC3 Briefing |
| All members | Mar. 31st  (8 hours) | Use Case 3 Pattern Implementation, Use Case 2 Coding review and revision, UC4 Briefing |
| All members | Apr. 2nd  (7 hours) | Use Case 4 Pattern Implementation, Use Case 3 Coding review and revision |
| All members | Apr. 3rd  (4 hours) | Overall Optimization and Commenting, Test Cases |
| All members | Apr. 4th  (4 hours) | Group Meeting Logs, Final Review and Submission |

|  |  |  |
| --- | --- | --- |
| Backlog item | Estimate (hours) | Members |
| Structure Design | 2 | YZ, SH, ZL, MY |
| Design Pattern Application | 4 | YZ, SH, ZL, MY |
| Login System Design and Coding | 4 | ZL, MY |
| Use Case 2 Pattern Design | 2 | YZ, SH |
| Use Case 2 Pattern Implementation | 10 | YZ, SH |
| Use Case 3 Pattern Design | 2 | YZ, SH |
| Use Case 3 Pattern Implementation | 14 | YZ, SH |
| Use Case 4 Pattern Design | 2 | ZL, MY |
| Use Case 4 Pattern Implementation | 8 | ZL, MY |
| Create Necessary Utility Classes | 6 | YZ, SH, ZL, MY |
| Overall Optimization and Commenting | 4 | YZ, SH, ZL, MY |
| Test Cases | 2 | ZL, SH |
| Group Meeting Logs | 2 | MY, ZH |