



islington college
(इस्लिंग्टन कॉलेज)

CS6P05 Final Year Project Computing

Final Report

Samanghar – An auction and shop based Ecommerce solution

Student Name: Kushal Bhattarai

London Met ID: 17031137

College ID: NP01CP4A170221

Internal Supervisor: Roshan Tandukar

External Supervisor: Amulya Lohani

Due Date: 5th June 2020

Submission Date: 5th June 2020

Word Count: 8083

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and marks of zero

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Abstract

In the introduction of topic section the topic of eCommerce was discussed , problems were stated and solution were described as well as current scenario and motivation for the project to be done was included . In Background part of the report includes the background information about the project like what is it how it works and how was the logic and architecture how it is being made is described. The background portion contains review of similar systems, survey review. The development part includes the process of choosing methodology. Three methodologies were considered and one of them was selected and described how and why it was selected. In the Progress section the overall progress was explained. Use case, wireframes, ER as well as activity diagram was include in the progress section. In Analysis of Progress, the progress of the project was analysed and a new plan was considered to continue the project. In Further Work what is to be done was discussed and reported. In Appendix portion, the some contents of proposal are included like, previous plan and project Gantt chart. Milestones of the project Risks and contingency plans were included.

Acknowledgment

I would like to express their sincere gratitude to our module teachers Mr. Roshan Tandukar and Mr. Amulya Lohani supporting and guiding the outcome of this coursework. The tasks on hand were challenging and often confusing but due to their insightful narrative of the scenario, I have been able to accomplish our final year project. Their opinions and constructive criticism were very valuable to do all the tasks successfully. Furthermore, I am thankful and fortunate enough to get constant encouragement, support, and guidance from all the staff and teachers who took the time to review our work and provide us with their genuine suggestions.

Table of Contents

START OF THE REPORT	1
CHAPTER 1: INTRODUCTION.....	1
1.1 PROJECT DESCRIPTION.....	1
1.2 CURRENT SCENARIO	2
1.3 PROBLEM DOMAIN AND PROJECT AS A SOLUTION	4
1.4 AIM AND OBJECTIVES.....	5
1.5 STRUCTURE OF THE REPORT	6
1.5.1 BACKGROUND	6
1.5.2 DEVELOPMENT	6
1.5.3 TESTING AND ANALYSIS.....	6
1.5.4 CONCLUSION	6
CHAPTER 2: BACKGROUND	7
2.1 ABOUT THE END USERS.....	7
2.2 UNDERSTANDING THE SOLUTION.....	8
2.3 SIMILAR PROJECTS	10
2.4 COMPARISONS	13
CHAPTER 3: DEVELOPMENT	14
3.1 CONSIDERED METHODOLOGIES	14
3.1.1 Methodology 1- RUP.....	14
3.1.2 Methodology 2- Spiral.....	15
3.1.3 Methodology 1- Waterfall.....	15
3.2 SELECTED METHODOLOGY.....	16
3.3 PHASES OF METHODOLGY	17

3.4 SURVEY RESULTS.....	21
3.3.1 PRE-SURVEY RESULTS	21
3.3.2 POST-SURVEY RESULTS	26
3.5 REQUIREMENT ANALYSIS	29
3.6 DESIGN TECHNIQUES	30
3.7 IMPLEMENATION.....	33
CHAPTER 4: TESTING AND ANALYSIS.....	44
4.1 TEST PLAN.....	44
4.1.1 UNIT TESTING, TEST PLAN	44
4.1.2 SYSTEM TEST PLAN	46
4.2 UNIT TESTING	47
4.3 SYSTEM TESTING	63
4.4 CRITICAL ANALYSIS	94
CHAPTER 5: CONCLUSION	95
5.1 LEGAL, SOCIAL AND ETHICAL ISSUES.....	95
5.1.1 LEGAL ISSUES	95
5.1.2 SOCIAL ISSUES	95
5.1.3 ETHICAL ISSUES	95
5.2 ADVANTAGES	96
5.4 LIMITATIONS	96
5.3 FUTURE WORK	96
CHAPTER 6: REFERENCES.....	97
CHAPTER 7: BIBLIOGRAPHY	99
CHAPTER 8: APPENDIX.....	100

8.1 APPENDIX A: PRE-SURVEY	100
8.1.1 PRE-SURVEY FORM.....	100
8.1.2 SAMPLE OF FILLED PRE-SURVEY FORMS.....	100
8.1.3 PRE-SURVEY RESULT	104
8.2 APPENDIX B: POST-SURVEY	109
8.2.1 POST-SURVEY FORM	109
8.2.2 SAMPLE OF FILLED POST-SURVEY FORMS.....	110
8.2.3 POST-SURVEY RESULT.....	112
8.3 APPENDIX C: SAMPLE CODES.....	115
8.3.1 SAMPLE CODE OF THE UI	115
8.3.2 SAMPLE CODE FOR THE SYSTEM	120
8.4 APPENDIX D: DESIGNS	128
8.4.1 GANTT CHART.....	128
8.4.2 WORK BREAKDOWN STRUCTURE.....	130
8.4.3 SEQUENCE DIAGRAMS.....	131
8.4.4 ER-DIAGRAM	137
8.4.5 ACTIVITY DIAGRAM.....	140
8.4.6 DATA FLOW DIAGRAMS	141
8.4.7 USE CASE	148
8.4.8 WIREFRAME	149
8.5 APPENDIX E: SCREENSHOTS OF THE SYSTEM.....	154
8.6 APPENDIX F: USER FEEDBACK.....	161
8.6.1 USER FEEDBACK FORM.....	161
8.6.2 SAMPLE OF FILLED USER FEEDBACK FORMS	163

8.7: APPENDIX G: FUTURE WORK 165

8.7.1 READINGS FOR FUTURE WORK 165

Table of figures

<i>Figure 1:Ebay workings.....</i>	11
<i>Figure 2:Ebay.....</i>	11
<i>Figure 3:Hamrobazar site (hamrobazaar, 2020)</i>	12
<i>Figure 4:Daraz (Daraz, 2020)</i>	12
<i>Figure 5: RUP development framework cycle (Study.com, 2019)</i>	14
<i>Figure 6:Survery question 1 & 2</i>	21
<i>Figure 7:Server Question 3 and 4</i>	22
<i>Figure 8:Survey question 5 and 6.....</i>	23
<i>Figure 9: Survey Question 7& 8.....</i>	24
<i>Figure 10: Survey Question 9& 10.....</i>	25
<i>Figure 11: Post survey result 1.....</i>	26
<i>Figure 12: Post survey result 2.....</i>	27
<i>Figure 13:Post survey result 3</i>	28
<i>Figure 14: ER diagram</i>	30
<i>Figure 15: Use Case Diagram</i>	31
<i>Figure 16: Iteration Workflow.....</i>	32
<i>Figure 17:System architecture</i>	33
<i>Figure 18: Login.....</i>	34
<i>Figure 19:Profile</i>	34
<i>Figure 20:Order.....</i>	35
<i>Figure 21: Shop product</i>	35
<i>Figure 22:Auction Product detail.....</i>	36
<i>Figure 23:Cart.....</i>	36

<i>Figure 24:Admin page</i>	37
<i>Figure 25:Shop</i>	37
<i>Figure 26:Search</i>	37
<i>Figure 27:Functions.....</i>	38
<i>Figure 28:Expired/ purchased.....</i>	38
<i>Figure 29: Favourite.....</i>	38
<i>Figure 30:Placed order.....</i>	39
<i>Figure 31:Notifications</i>	39
<i>Figure 32:Stripe payment.....</i>	39
<i>Figure 33:Payemnt cash on delivery</i>	40
<i>Figure 34:Order details</i>	40
<i>Figure 35: running unit test in system</i>	47
<i>Figure 36: test completed.....</i>	47
<i>Figure 37: testing error of save bid because of.....</i>	47
<i>Figure 38 :error code.....</i>	48
<i>Figure 39:unit test code for testing user creating</i>	48
<i>Figure 40: register user and checking object name</i>	49
<i>Figure 41: Testing adding product in both shop and auction.....</i>	49
<i>Figure 42: HomeView and ShopView test without login</i>	50
<i>Figure 43: Auction , expired without login and login testing</i>	51
<i>Figure 44: Notification, and bids without login test.....</i>	51
<i>Figure 45: Login testing with credentials and with login url</i>	52
<i>Figure 46: testing login , home website home url in same session after logged in.....</i>	52
<i>Figure 47:testing in same logged in session</i>	53

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 48:same logged in session with different url and creating product test.....</i>	54
<i>Figure 49: testing other urls in same logged in session</i>	55
<i>Figure 50: testing other urls, and updating sending message in same logged in session</i>	56
<i>Figure 51: testing urls in the same logged in session and login out</i>	57
<i>Figure 52: Testing Adding to cart</i>	57
<i>Figure 53: Testing bidding.....</i>	57
<i>Figure 54: Test rebidding.....</i>	58
<i>Figure 55: Register test or user Admin, SuperAdmin ,Admin and wrong password.....</i>	59
<i>Figure 56: Test for verifying login.....</i>	59
<i>Figure 57: Testing different urls.....</i>	60
<i>Figure 58: Testing funcitons like end, history, favourites.....</i>	61
<i>Figure 59: testing login, and testing if staff can bid or not.....</i>	62
<i>Figure 60: login test.....</i>	63
<i>Figure 61: test result.....</i>	64
<i>Figure 62: testing 2.....</i>	64
<i>Figure 63: test result -2.....</i>	65
<i>Figure 64 : login form.....</i>	65
<i>Figure 65:test result</i>	66
<i>Figure 66: add product form.....</i>	66
<i>Figure 67:added products</i>	67
<i>Figure 68:displayed new products.....</i>	67
<i>Figure 69:Product update 1</i>	68
<i>Figure 70:Product update 2</i>	69
<i>Figure 71:Product update 3</i>	69

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 72: add 12 hr</i>	70
<i>Figure 73: added 12 hr 11:27 am to 11:27 pm</i>	71
<i>Figure 74: end auction.....</i>	71
<i>Figure 75:Auction ended.....</i>	72
<i>Figure 76: Add to Cart.....</i>	73
<i>Figure 77: Added to cart</i>	73
<i>Figure 78: Test 9.....</i>	74
<i>Figure 79: cart cash on delivery.....</i>	74
<i>Figure 80: filling form.....</i>	74
<i>Figure 81:From filled</i>	75
<i>Figure 82:Order placed</i>	75
<i>Figure 83:Order page</i>	76
<i>Figure 84: Order detail page</i>	76
<i>Figure 85:Adding shop products</i>	77
<i>Figure 86: Successfully add shop product</i>	77
<i>Figure 87:Search category products</i>	78
<i>Figure 88:search.....</i>	79
<i>Figure 89: View notification.....</i>	80
<i>Figure 90: Viewed notification.....</i>	80
<i>Figure 91: dispute.....</i>	81
<i>Figure 92: won auctions.....</i>	82
<i>Figure 93:Filling form</i>	83
<i>Figure 94: filling form.....</i>	83
<i>Figure 95:successfully order place.....</i>	84

<i>Figure 96: conversation</i>	85
<i>Figure 97: added converdation</i>	86
<i>Figure 98: logout.....</i>	87
<i>Figure 99: after logout.....</i>	87
<i>Figure 100: Change Password.....</i>	88
<i>Figure 101:Reset Pasword</i>	88
<i>Figure 102: reset password sent.....</i>	89
<i>Figure 103: link to reset password</i>	89
<i>Figure 104: form for reset password</i>	90
<i>Figure 105: new password formed</i>	90
<i>Figure 106:Profile update.....</i>	91
<i>Figure 107:Form for profile update</i>	91
<i>Figure 108:Updated profile.....</i>	91
<i>Figure 109: Cart</i>	92
<i>Figure 110: Stripe payment.....</i>	92
<i>Figure 111:Stripe payment continued.....</i>	93
<i>Figure 112:Order completed.....</i>	93
<i>Figure 113:pre survey form.....</i>	100
<i>Figure 114:Survey -1</i>	100
<i>Figure 115: Survey -2</i>	101
<i>Figure 116: Survey -3</i>	101
<i>Figure 117: Survey -4</i>	102
<i>Figure 118: Survey- 5</i>	102
<i>Figure 119: Survey -6</i>	103

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 120:Result -1</i>	104
<i>Figure 121: Result 2.....</i>	105
<i>Figure 122: Result -3</i>	106
<i>Figure 123: Result 4.....</i>	107
<i>Figure 124: Result 5.....</i>	108
<i>Figure 125: Form1.....</i>	109
<i>Figure 126:Form2.....</i>	109
<i>Figure 127:Form 3.....</i>	109
<i>Figure 128: filled form1</i>	110
<i>Figure 129: filled form-2.....</i>	111
<i>Figure 130: Filled form-3</i>	111
<i>Figure 131:Result-1</i>	112
<i>Figure 132:Result 2</i>	113
<i>Figure 133:Result 3</i>	114
<i>Figure 134:base code.....</i>	115
<i>Figure 135:auction</i>	116
<i>Figure 136:Cart.....</i>	117
<i>Figure 137:checkout</i>	117
<i>Figure 138:Product detail.....</i>	118
<i>Figure 139:My Products</i>	119
<i>Figure 140:Home.....</i>	120
<i>Figure 141: Django Settings</i>	120
<i>Figure 142:User Views</i>	121
<i>Figure 143:User urls.....</i>	121

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 144: Signals.....</i>	122
<i>Figure 145: Models.....</i>	122
<i>Figure 146: Forms.....</i>	123
<i>Figure 147: Admin.....</i>	123
<i>Figure 148: class Based Views.....</i>	124
<i>Figure 149: Models.....</i>	124
<i>Figure 150: Tests</i>	125
<i>Figure 151: Website Urls</i>	126
<i>Figure 152: Function based website views</i>	127
<i>Figure 153: Gantt chart 1</i>	128
<i>Figure 154: Gantt chart 2</i>	129
<i>Figure 155: Manage Products.....</i>	131
<i>Figure 156: messages</i>	131
<i>Figure 157: View Order Products</i>	132
<i>Figure 158: Register</i>	133
<i>Figure 159: SearchProducts</i>	134
<i>Figure 160: Update Products.....</i>	135
<i>Figure 161: Login.....</i>	136
<i>Figure 162: Final ER daigram</i>	137
<i>Figure 163: 1st Draft Diagram</i>	138
<i>Figure 164: 2nd Draft</i>	139
<i>Figure 165: Activity diagram</i>	140
<i>Figure 166: DFD level 0.....</i>	141
<i>Figure 167: DFD level 1</i>	142

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 168: Dfd level1 of function 1 and 2</i>	143
<i>Figure 169:Dfd 3 level 2</i>	144
<i>Figure 170: dfd diagram level2 of function 3.....</i>	145
<i>Figure 171: dfd diagram level2 offunction8.....</i>	146
<i>Figure 172: dfd diagram level2 offunction8.....</i>	147
<i>Figure 173:Use Case Diagram.....</i>	148
<i>Figure 174:Wireframe part 1</i>	149
<i>Figure 175: Wireframe part 2</i>	150
<i>Figure 176: Wireframe part 3</i>	151
<i>Figure 177: Wireframe part 4</i>	152
<i>Figure 178: Wireframe part5</i>	153
<i>Figure 179: home page</i>	154
<i>Figure 180:Auction page.....</i>	154
<i>Figure 181: Shop page.....</i>	155
<i>Figure 182: Expired/purchased.....</i>	155
<i>Figure 183: Seller dashboard.....</i>	156
<i>Figure 184:MY uploaded product</i>	156
<i>Figure 185: MY bids page.....</i>	156
<i>Figure 186:History page</i>	157
<i>Figure 187: favourites.....</i>	157
<i>Figure 188: profile.....</i>	158
<i>Figure 189:Admin dashoard</i>	158
<i>Figure 190: Super admin dash board</i>	159
<i>Figure 191:Search product</i>	159

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Figure 192:product deatils.....</i>	160
<i>Figure 193: About page</i>	160
<i>Figure 194: user feedback result -1.....</i>	161
<i>Figure 195: user feedback result -2.....</i>	161
<i>Figure 196: user feedback result -3.....</i>	161
<i>Figure 197: user feedback result - 4.....</i>	162
<i>Figure 198: user feedback result -5.....</i>	162
<i>Figure 199: user feedback result 6</i>	162
<i>Figure 200: sample -1</i>	163
<i>Figure 201: sample -2</i>	163
<i>Figure 202: sample3</i>	164
<i>Figure 203: sample 4</i>	164
<i>Figure 204: recommendation further work sample</i>	165
<i>Figure 205:communication with buyer and supplier.....</i>	165

Table of tables

<i>Table 1: Table for product comparisons.....</i>	<i>13</i>
<i>Table 2: Resourcce Planning</i>	<i>16</i>
<i>Table 3:Inception Phase.....</i>	<i>19</i>
<i>Table 4:Elaboration.....</i>	<i>19</i>
<i>Table 5:Construction</i>	<i>19</i>
<i>Table 6: Construction pasrt2.....</i>	<i>20</i>
<i>Table 7:Transiton fucntion</i>	<i>20</i>
<i>Table 8: MoSCow Prioritization parts.....</i>	<i>29</i>
<i>Table 9: Test 1.....</i>	<i>63</i>
<i>Table 10: Test 2</i>	<i>64</i>
<i>Table 11:Test 3</i>	<i>65</i>
<i>Table 12: Test 4</i>	<i>66</i>
<i>Table 13: Test 5</i>	<i>67</i>
<i>Table 14:Test 6</i>	<i>71</i>
<i>Table 15: Test 8</i>	<i>72</i>
<i>Table 16:Test 10</i>	<i>76</i>
<i>Table 17: Test 11</i>	<i>77</i>
<i>Table 18:Test 12</i>	<i>78</i>
<i>Table 19: TEST 13</i>	<i>79</i>
<i>Table 20: TEST 14</i>	<i>81</i>
<i>Table 21: Test 15</i>	<i>82</i>
<i>Table 22: Test 16</i>	<i>85</i>
<i>Table 23: Test 17</i>	<i>86</i>

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

<i>Table 24: Test18</i>	88
<i>Table 25: Test 19</i>	91
<i>Table 26: Test 20</i>	92
<i>Table 27: Work breakdown structure</i>	130

START OF THE REPORT

CHAPTER 1: INTRODUCTION

1.1 PROJECT DESCRIPTION

The expectation is to develop a website based on the process of the C2C model. My website created will have a connection between the two users through a site where the details are present on the website, and the communication of users is conducted. The login system for all types of users will be created. A proper Dashboard for admins for the customization of different products and managing the users will be provided. The users can buy and sell items through the bidding system. A proper communication system for the buyer and seller will be implemented. Full Transparency will be provided about the bidders, seller's products and final buyer. History of the products sold and bought products can be viewed. Messaging system for the Users when the auction has started or any items that have been added to the selling section of the website is a feature that will be present on the site. The seller will provide all the required details about the products and will upload a video or picture on the website. The Website will have a proper checkout system where the winning bidder can check out. A different section where the products can be bought directly from a seller without bidding and the seller has agreed not to sell the item through a bidding system. The registered users can bid on the item and compete with other bidders. The payment system can be cash upon delivery.

1.2 CURRENT SCENARIO

The e-commerce market in Nepal is worth around \$25 million. The market is seeing 300 percent growth annually. Customers are trusting e-commerce more, and sellers are seeing the value of e-commerce. (**The Kathmandu Post, 2020**)

As of May 2019, there were 31 private ISPs in Nepal, with about 200,000 subscribers, and nearly 16.67 million internet users nationwide that are based on E-commerce. Roughly 40 percent of these accounts are commercial, with businesses promoting their products and services and communicating with foreign companies via the internet. Ecommerce is still in its infancy in Nepal. (**export.gov, 2019**) The country's challenging terrain and lack of street addresses make deliveries a challenge. My project is based on Consumer to consumer deals with used things or products. Here a consumer can post his products, goods, services and another consumer, who is interested, can purchase it by using this E-commerce system with the addition of bidding. This web technology helps us to sell our used properties or assets like bike, car, house, any electronic items or kitchen items by using this model as well as new products. Hamrobazar, Sastoramro, and many other sites can provide you this service, but they do not have to buy now feature and bidding features. In those websites only the information is provided, and the users must communicate with their phone and make specific deals. I want to create a bidding system where the users can bid, and the seller can accept the bid and the potential buyers can not only have information about the products but can buy them with a proper payment system. People use social media sites like Facebook and Instagram to sell their products through Facebook groups and Instagram stories. The problem is that people can't know who to trust and will the person be able to give money. "A recent survey conducted by LocalCircles shows that 38 percent of respondents out of 6,923 have been sold counterfeit products from an eCommerce site in the past year. Among the leading eCommerce sites, 12 percent of the respondents said they received fake products." (**Devansh Sharma, 2018**) There will be spammers who would spam the owner of the product, with that problem in mind I want to create a website where people can display the information of products and place an offer for the product owner.

Industry insiders and government officials said that the e-commerce market would grow manifold due to a swelling middle class and a surge in the number of internet users. The Nepal Telecommunications Authority claimed that internet penetration in Nepal had reached 63 percent of the total population as of 2017 following a smartphone boom. Another reason behind the burgeoning e-commerce market is the large number of Nepalis living abroad—for work, study and business—where they gain exposure to international brands. When they return home, they prefer to purchase international products online. Toya Narayan Gyawali, joint secretary of the Ministry of Industry, Commerce and Supplies, told the Post that the draft regulations would be given final shape within a month. “After the draft is finalized, it will be submitted to the Cabinet for its approval to implement it.” The absence of regulations has led to an increasing number of consumer complaints. One of the key features of the regulation would be consumer protection. E-tailers and marketplaces will be made liable for fraudulent sales and substandard or defective products, Gyawali said. Besides checking fraud and business misconduct, the regulations will properly manage electronic transactions, regulate foreign currency transactions and promote Nepali products through online stores on the global market, said Gyawali. There are no exact statistics of the size of the e-commerce business in Nepal, but market insiders estimate annual turnover to be in the neighbourhood of \$25 million. They said that the number of online buyers had been increasing rapidly in recent years, particularly in the Kathmandu Valley, as they can buy a wide range of national and international products with a few clicks of the mouse. Normally, it’s a cash on delivery business model that reduces hassles and saves time for customers. The government doesn’t have the exact date for online marketplaces. According to a government official, there are a large number of unregistered online shops. Daraz, Sastodeal, eSewaPasal, Metro Tarkari, Bhatbhateni Online, Mero Kirana, Foodmario, Foodmandu and Urban Girl are some of the popular online stores in the country. Several rounds of discussions have been conducted with the private sector to prepare the draft, said Gyawali. The proposed regulation focuses on promoting e-commerce in Nepal by easing the process of doing online business. “Our neighbours India and China have made great strides in the development of e-commerce, and Nepal also needs to move ahead by improving services,” he said. India’s e-commerce market is expected to grow more than fourfold to \$150 billion by 2022, fuelled by rising incomes and a surge in internet users, according to a report by software industry lobby group

Nasscom and consulting firm PwC India. The e-commerce market was valued at \$36 billion in 2017. China's online retail market is expected to hit \$1.8 trillion in 2022, buoyed by local tech titans Alibaba and JD.com, according to a report by Forrester. "E-commerce has massive growth potential, so proper regulations are necessary for the development of this sector," said Gyawali. Nepal lags behind in e-commerce because of a poor information and technology infrastructure, he said. "With regard to IT friendly people, Nepal falls in the average category among the least developed countries." <https://kathmandupost.com/money/2019/03/06/government-to-introduce-e-commerce-regulations>

1.3 PROBLEM DOMAIN AND PROJECT AS A SOLUTION

The Ecommerce website is pretty much everywhere in Nepal but has not been too much profitable as well as the auction system in these platforms have not been implemented. A proper auction website has not been available in Nepal in present context. A competitive e-commerce platform between the users is not available in the country. In present eCommerce websites sometimes only information is provided to customer. The users must communicate with their phones and make specific deals according to the information on the website. I wanted to create a website where the user doesn't have to communicate through their phone but can communicate from my website and buy and bid in the products. I want to create a bidding system where the users can bid, and the seller can accept the bid and the potential buyers can not only have information about the products but can buy them with proper payment system. Even though people use Ecommerce system but have People still use social media sites like Facebook and Instagram to sell their products through Facebook groups and Instagram stories.

The following are some of the disadvantages of an existing auction system

- The traditional method is a time-consuming process
- Date and time play an important role, as they operate for a few hours only.
- There is no separate module for sellers to upload their own Product in an online auction system.

Auctions are used to sell many things in addition to antiques and art. All-round the world there are auctions of commodities such as cattle, racehorses and just above anything elsewhere, there's a market of multiple people interested in buying the same thing that's the key to auction-a bunch of people who are interested in buying the same object and taking turns offering bids on the object. The right to buy that object will go to the highest bidder; It is called traditional auction

While the update of the traditional auction is online auction, companies from various industries are moving to online auction say eBay, on sale provided a worldwide platform for bidder by getting it to masses. In these websites they create their own auction by adding their own product to auction. The buyer can bid the product which he/she can win the product if he applies the winning bid. In my system Date and time will also play an important role, but the auction can be fast slow according to the user. My system has a separate module for sellers to upload their own Product in an online auction system.

1.4 AIM AND OBJECTIVES

The project aims to create a C2C eCommerce creating an opportunity for people to sell, buy products with auction like bidding system implemented to the website. The project aims to sell some rare products quickly. The project's objective is to create shorter and lessened channels of distribution system which reduces operating costs in the project with cheap and quicker communication between buyers and sellers. The project's objective to target is to reach relevant markets.

1.5 STRUCTURE OF THE REPORT

1.5.1 BACKGROUND

This part contains the background portion of the system.

1.5.2 DEVELOPMENT

This part contains methodology, implementation, and screenshots of the system.

1.5.3 TESTING AND ANALYSIS

This part contains testing and critical analysis of the test.

1.5.4 CONCLUSION

This part contains the conclusion of the report advantage, limitations, issues and future word is discussed.

CHAPTER 2: BACKGROUND

2.1 ABOUT THE END USERS

There are three types of users in the system. Superadmin, Admin and User.

Guest User:

The type of users who don't have account and can only view the products and search the products but cannot add products to cart or place a bid on any auction.

User:

The type of user who has an account where they can buy, bid, add to favourite, watch history about watched products, search and add products.

Admin:

The type of users who can view the analytics portion of the system like, who manages disputes and check the order, and can view the reports of the type of users, products added, orders but cannot bid on a project and cannot buy a product. The admin user has the capability to list the products.

SuperAdmin:

The type of users who have absolute control over the products and can view, edit, delete anything from the system. The type of users who can view the analytics portion of the system like, who manages disputes and check the order, and can view the reports of the type of users, products added, orders but cannot bid on a project and cannot buy a product. The admin user has the capability to list the products.

2.2 UNDERSTANDING THE SOLUTION

Django is a high-level Python Web framework that helps in rapid development and clean, pragmatic design. It's free and open-source, fast and exceedingly scalable. Django was intended to help developers take applications from concept to completion as quickly as possible. Django has a great security measure and it is reassuringly secure. One of the most important aspects that have made Django popular is because of its ability to quickly and flexibly develop the web. (**Django Software Foundation , 2020**)

Django with authorization supports the databases: PostgreSQL, MariaDB, MySQL, Oracle SQLite. Django supports many features as possible on all database backends. Django needs mysqlclient, mysqlconnect-c to connect to the MySQL database. (Django Software Foundation , 2020) The tables in the database are created automatically by Django using Django Models and but first, a database must be created to store the tables in Database. MySQL is an Oracle-backed open-source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL is based on a client-server model. The core of MySQL is MySQL server, which handles all of the database instructions (or commands). (**TechTarget, 2020**)

JetBrains is a global software vendor specializing in the creation of intelligent, productivity-enhancing tools for software developers and teams. JetBrains was founded in February 2000. Jet Brains has been continuously updating by automating routine checks and corrections, our tools speed up production, freeing developers to grow, discover, and create. The Python IDE for professional developers. PyCharm offers a unique coding experience for productive Python, Django, and web development. (JetBrains s.r.o., 2018) PyCharm offers great framework-specific support for modern web development frameworks such as Django, Flask, Google App Engine, Pyramid, and web2py, including Django templates debugger, manage.py, and appcfg.py tools, special auto completion and navigation, just to name a few. (**JetBrains s.r.o., 2020**)

The system is defined below: A client places an order or wins the bid. The order placed can directly go to the cart . The bid directly goes to order and is charged through an online payment system the order is completed. If the online payment system is not acceptable to the user can select cash on delivery service. The order is completed an email is sent to both seller and buyer. The placed order is carried out through the buyer and seller. The shipment is collected by the buyer who will collect it. My system user 3 tier application architecture where client interface, application layer and database layer. The client/ user interface includes webserver, browser and internet. Suppose a user wants to buy a product form the website. A user uses a web browser to visit the website connects through webserver to communicate with the e-commerce customer to the seller. The application layer includes the Application layer and the backend layer. The purpose of the application layer is to log in to the shopping system and check products that are available on the site. The database layer is the storehouse of data where all the data are stored in the database like product data, seller orders. The layer is accessible only through the application layer.

2.3 SIMILAR PROJECTS

One of the systems similar to my project is eBay, the company was founded in 1995. eBay is where the world goes shopping, sell, and give items basically a trading spot for people. In 2019 there have been 183 million active buyers. Sellers have the platform to grow their businesses through eBay. eBay measures success by our customers' success. eBay is in the top 10 global retail brands. (eBay Inc., 2020)

There are two types of listings on eBay that are auction-format listings and fixed-price listings. Fixed price listings are the same as online shopping. the user clicks a button to make a purchase, then pay for it. Fixed-price items have a "Buy It Now" button with a price listed next to it. The users can buy after clicking the "Buy It Now" button paying the price listed next to the button. Auction items have a "Place Bid" button. The box for entering bids and shows "current bid" prices are also available. Auction items are open to bids for a fixed amount of time. When time is up, the item is declared "sold" to the highest bidder. eBay auctions accept bids only for a specific amount of time. Users must place a bid that is higher than the current bid. The bid winner can pay less than your bid if the user wins but not less than the 2nd bidder's amount. If the user's bid wins, the payment must be done. Functionality like Use 1-Click Bid can be used to place small bids and 3rd party programs can also be used to bid for automatic bidding system. (Hsiao, 2018)

eBay has built an online person-to-person trading community on the Internet. In the website sellers list items for sale, buyers to bid on items of interest as well as can buy the items. eBay users can browse the items. The items are arranged by topics, where each type of auction has its own category. This facilitates easy exploration for buyers and enables the sellers to list an item for sale immediately after registering. eBay notifies the buyer and seller via e-mail at the end of the auction if a bid exceeds the seller's minimum price, and the seller and buyer finish the transaction independently of eBay. The binding contract of the auction is between the winning bidder and the seller only. (brandeis.edu, 2016)

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

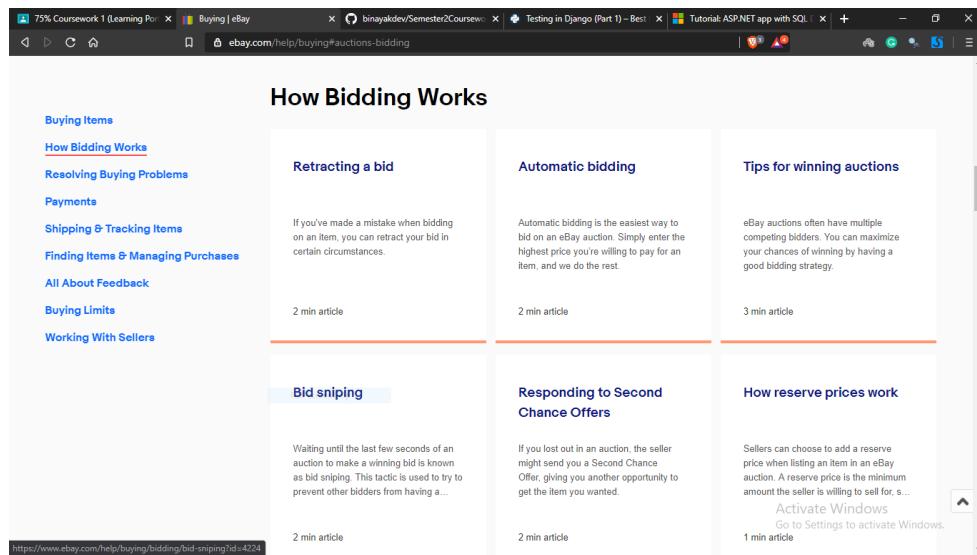


Figure 1: Ebay workings

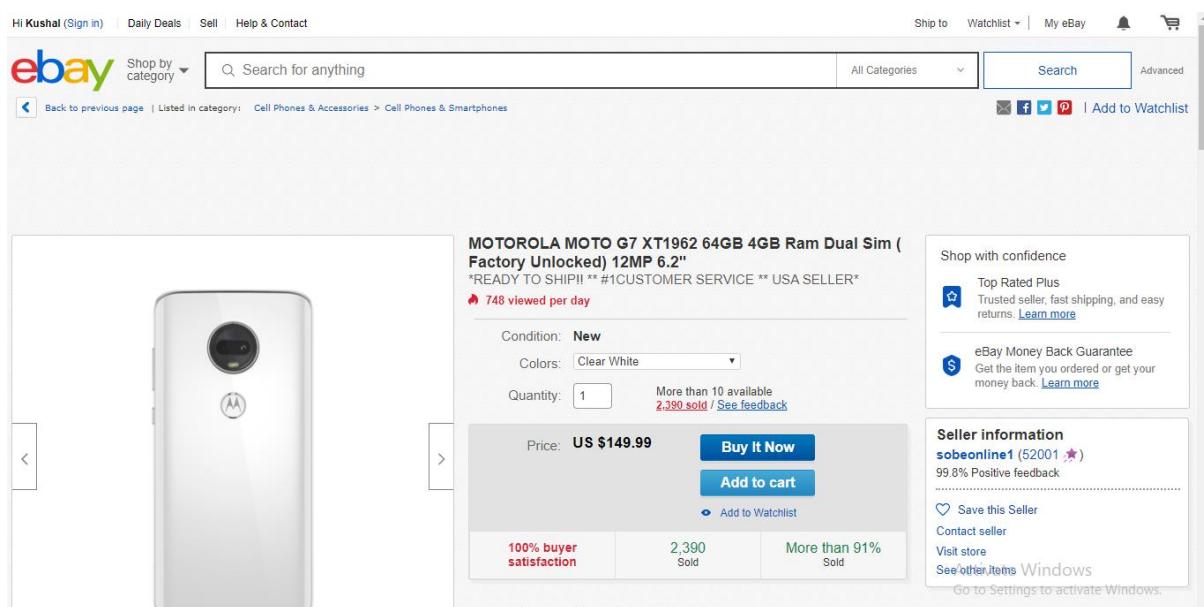


Figure 2:Ebay

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

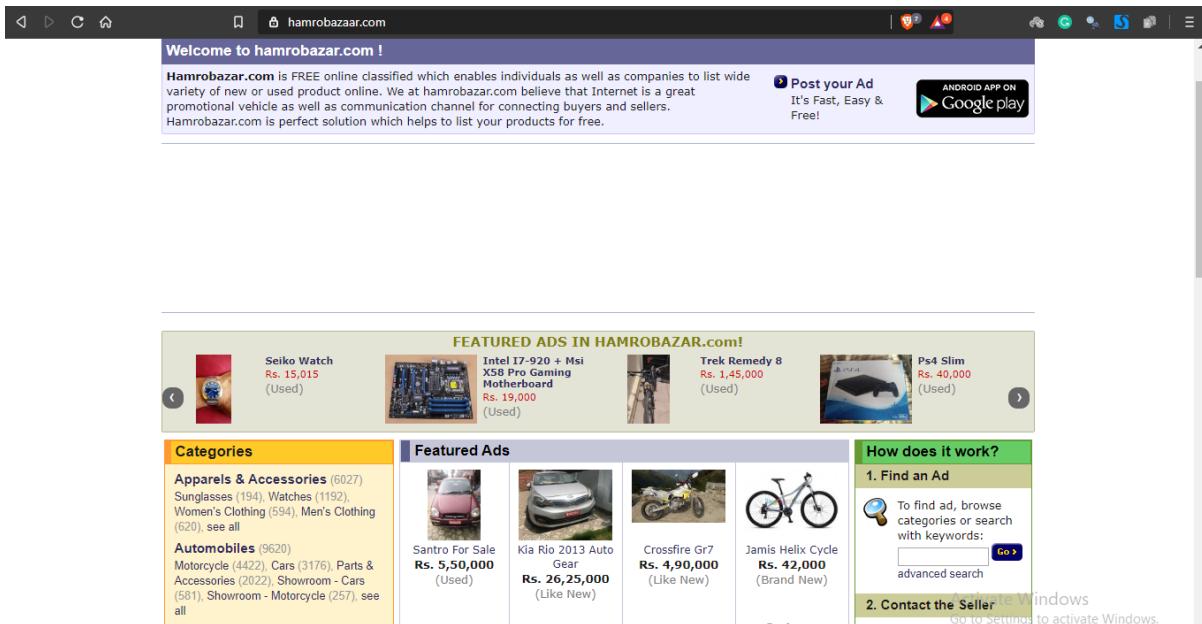


Figure 3:Hamrobazar site (hamrobazaar, 2020)

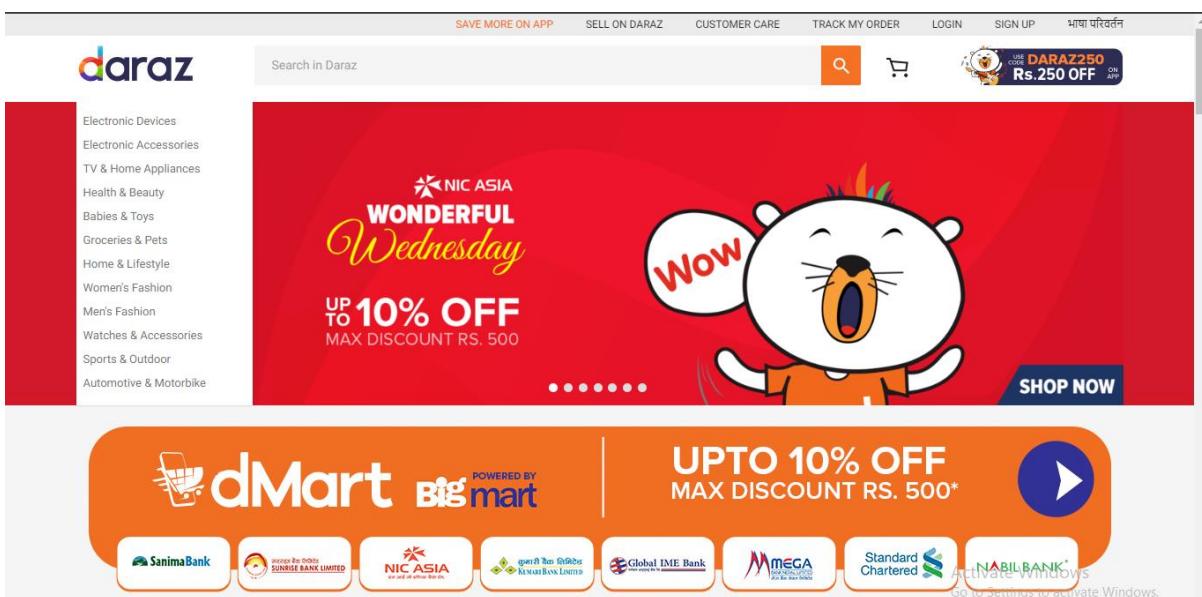


Figure 4:Daraz (Daraz, 2020)

2.4 COMPARISONS

Feature Desc	Samanghar(my project)	Ebay	Daraz	Hamrobazar
Add products	Yes	Yes	Yes	Yes
Add auction/listings	Yes	No	No	Yes
Time based Listings/ auction	Yes	No	No	Yes
Online Payment	Yes	Yes	Yes	Yes
Cash On delivery	Yes	Yes	Yes	Yes

Table 1: Table for product comparisons

Hamrobazar.com is FREE online classified which enables individuals as well as companies to list wide variety of new or used product online. We at hamrobazar.com believe that Internet is a great promotional vehicle as well as communication channel for connecting buyers and sellers. As per NTA April 2019 report, the internet users in Nepal has reached 18.24 million (including 14.24 million mobile data users) which itself indicates that the market for internet advertising is highly lucrative. Hamrobazar is visited by around 800,000 unique visitors monthly who use the site for buying and selling purpose. (hamrobazaar, 2020)

Daraz is the leading online marketplace in South Asia connecting thousands of sellers with millions of customers in Pakistan, Bangladesh, Sri Lanka, Nepal and Myanmar. Daraz has built a robust payment infrastructure that is localised to each market. Daraz have integrated with leading banks and local opened-loop wallets in the five countries to ensure that customers have a number of digital payment options. In Pakistan, the closed-loop Daraz Wallet has been launched as the smartest payment option. Daraz is known to be the best online selling website in Nepal. (Daraz, 2020)

CHAPTER 3: DEVELOPMENT

3.1 CONSIDERED METHODOLOGIES

3.1.1 Methodology 1- RUP

The RUP project is iterative and is measured in milestones. The iterations are used to manage the project by the Project manager. Here the RUP Planning process is developed in a structured way. The plan is made for each step and helps to reduce waste of resources. Inception Elaboration, Construction, and transition are the phases that are repeated in many iterations in this methodology. Each of the four stages has the main objective, which has to be completed before the project can progress to the next step. Each of the life cycle phases can be repeated if needed until the main objectives are met. Once the transition stage is completed successfully, the project is finished. (Study.com, 2019) (Sharpened Productions , 2019)

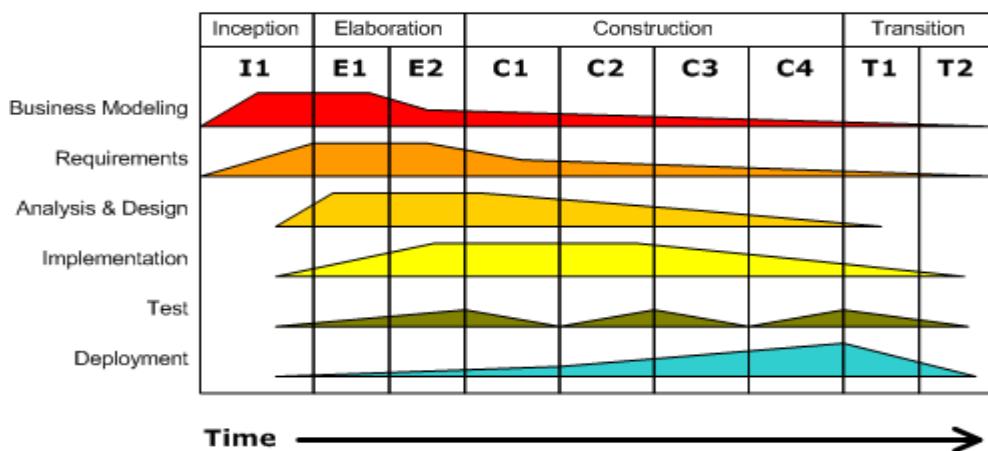


Figure 5: RUP development framework cycle (Study.com, 2019)

Advantages of RUP

1. This is a complete methodology in itself with an emphasis on accurate documentation
2. It is proactively able to resolve the project risks associated with the client's evolving requirements requiring careful change request management
3. Less time is required for integration as the process of integration goes on throughout the software development life cycle.
4. The development time required is less due to reuse of components.

3.1.2 Methodology 2- Spiral

The Spiral model is Meta-model as a result of it subsumes all the initial models. This model is named from its diagrammatic appearance that looks like a spiral with many loops. The exact number of loops of the spiral is not fixed and can vary from project to project. Each loop of the spiral is called a phase of the software process. The exact number of phases through which the product is developed can differ in according to the risk factor. The model helps the developer to understand and resolve the risks at each iteration along the spiral loop. The spiral model uses prototyping is used to produce a risk reduction mechanism. The methodology also uses the systematic stepwise approach of the waterfall model in the spiral loops. Disadvantages of the Spiral Model is that the process is risky, time-consuming costly, and because of many strict protocols that need to follow, the risk factor of the process is high. A risk analysis expert is required and could be costly which could again affect the project budget. Even though it is used in large scale projects Because of its disadvantages and risk factors affect it, I did not choose this methodology. (Learntek, 2019)

3.1.3 Methodology 1- Waterfall

The classical waterfall model divides the SDLC life cycle into a set of phases which is only started after the previous phase is completed. The output of one phase will be the input to the next phase. The phase in the methodology is as follows: Feasibility Study, Requirements analysis and specification, Design, Coding and Unit testing, Integration and System Testing and Maintenance. The development process is in a sequence that flows like a waterfall where the phases do not overlap over one another and don't go from down to up. The waterfall model is not practicable as there are not any feedback path. The process flows one after another as if an error is not committed in any phases and the project was done in 100% accuracy.

This doesn't incorporate any mechanism for error correction so it's not practicable. The requirements can change during the process and the change requests can be difficult to accommodate after the requirement specifications. This can't be maintained. To increase the efficiency and reduce the cost, phases may overlap but in the waterfall the phases do not overlap so even though I considered doing the project in waterfall methodology because of its disadvantages I did not choose this methodology. (geeksforgeeks, 2019)

3.2 SELECTED METHODOLOGY

The methodology that is going to be used is the Rational Unified Process RUP. Each phase is finalized with a milestone. A milestone is a point in time where decisions of critical importance must be made. In order to be able to make those decisions, the objectives must have been accomplished. For example, a milestone from the first two phases is the progress of the use case. A use case is a description of a system's behaviour and describes who can do what using a system. This is an important component in the development of software. Normally, a product will already have to be completed by then. That is because this involves prototypes and test models.

Resource	Worker	Activities
Kushal	Manager	Plans iteration
	Designer	Define prototype and operations
	Use Case Specifier	Details the use case
	System Analyst	Finds actor and Use cases
	Architect	Identify design mechanisms
	Coder	Builds the system
	Implementer	Performs tests

Table 2: Resource Planning

3.3 PHASES OF METHODOLOGY

The Stages/Phases of RUP are as follows:

1. Inception

The idea for the project is stated. The development team determines if the project is worth pursuing and what resources will be needed. During the first phase, the basic idea and structure of the project are determined. In this phase, the team meets regularly to determine the project's necessity, but also its viability and suitability. Viability and suitability also include the expected costs and the means needed to complete the project after the green light has been given (Rational, 2011) (Sharpened Productions , 2019)

2. Elaboration

The project's architecture and required resources are further evaluated. Developers consider possible applications of the software and costs associated with the development. During the elaboration phase, the system's requirements and its required architecture are assessed and analyzed. This is where the project begins to take shape. The objective of the elaboration phase is to analyze products and to lay a foundation for future architecture. (Rational, 2011) (Sharpened Productions , 2019)

3. Construction

The project is developed and completed. The software is designed, written, and tested. In the construction phase of the Rational Unified Process (RUP), the software system is constructed in its entirety. The emphasis is on the development of components and other features of the system. The majority of coding also takes place in this phase. In this production process, the emphasis is on managing costs and means, as well as ensuring quality. Results from the production phase include Fully completed software system User manuals. (Rational, 2011) (Sharpened Productions , 2019)

4. Transition

The software is released to the public. Final adjustments or updates are made based on feedback from end-users. The objective of the transition phase is to transfer the product to its new user. As soon as the user starts using the system, problems almost always arise that require changes to be made to the system. The goal, however, is to ensure a positive and smooth transition to the user. Results and activities in the last phase: Beta testing Conversion of existing user databases Training new users Rolling out of the project to marketing and distribution Input from the new users should guide the assessment here. (Sharpened Productions , 2019) (Rational, 2011)

AS seen, RUP methodology has a highly flexible development path. The best practice in RUP is Developing iteratively, managing proper requirements, use components of large projects to reduce production time, visualizing models using UML diagrams. In the methodology the software requirements specification (SRS) keeps on evolving throughout the development process and loops are created to add them without affecting the cost of development which helps in testing and using different components in development as well as testing. (My-Project-Management-Expert, 2009)

Phase	Inception	
Iteration Number	Iteration1	Iteration2
Objective	Research	Research
Implemented	Basic idea of the project was created	Researched about the project viability and necessity an research completed

Table 3:Inception Phase

Phase	Elaboration				
Iteration Number	Iteration 3	Iteration 4	Iteration 5	Iteration 6	
Objective	Create	Analyse	Update	New create	
Implemented	Find System requirements	Analysing the system requirements	Updating the requirements	Create new and updated requirements after update	

Table 4:Elaboration

Phase	Construction				
	Iteration 7	Iteration 8	Iteration 9	Iteration 10	Iteration 11
Objective	System update	System update	System update	System update	System update
Implemented	Logins, register, adding products	Authentication, dashboard	Updating UI system,	Cart , orders,	Search

Table 5:Construction

Phase	Construction				
	Iteration 12	Iteration 13	Iteration 14	Iteration 15	Iteration 16
Objective	System update	System update	System update	System update	System update
Implemented	cash on delivery payment	History fucniton	Notification	Updating searching	Admin Panel

Table 6: Construction part2

Phase	Transition		
Iteration Number	Iteration 12	Iteration 13	Iteration 14
Objective	Test	Update	Finalize
Implemented	Tested using unit testing coverage	Updation bugs	Finalized project

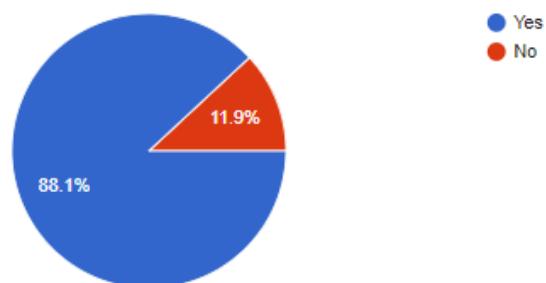
Table 7: Transition function

3.4 SURVEY RESULTS

3.3.1 PRE-SURVEY RESULTS

Have you heard about Customer to Customer E-commerce Platform?

42 responses



Have you used any Ecommerce platform to buy products ?

42 responses

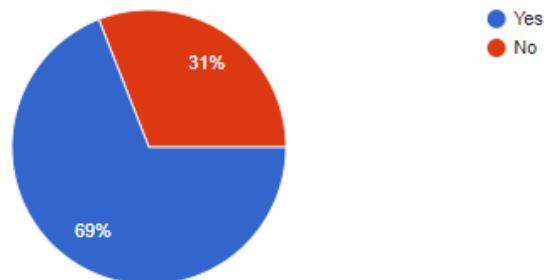
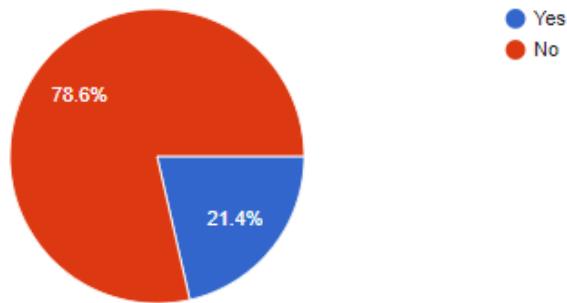


Figure 6:Survey question 1 & 2

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Have you sold your products in E-commerce platform?

42 responses



Have you ever been a part of auction of different products?

42 responses

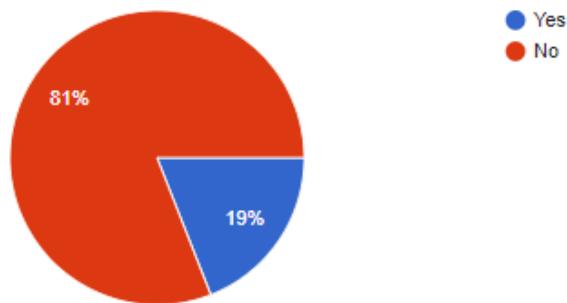
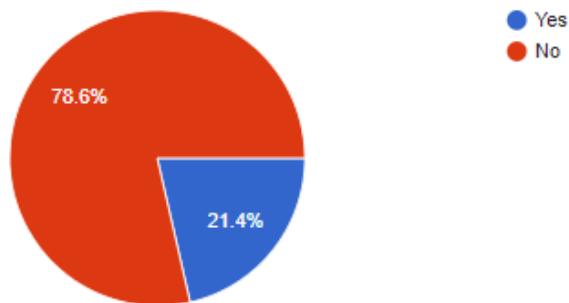


Figure 7:Server Question 3 and 4

Have you used an E-commerce Platform which implements auction system to buy products?



42 responses



Would you buy products from E-commerce platform?

42 responses

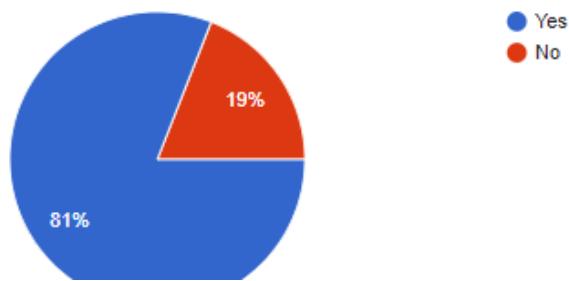
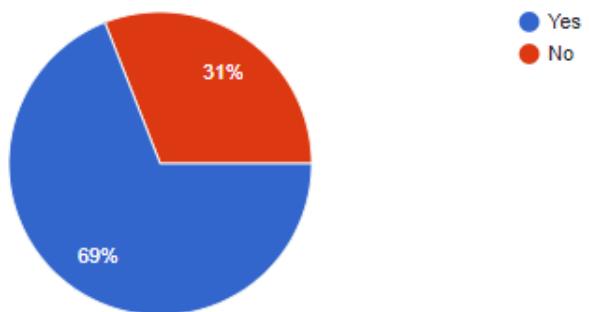


Figure 8:Survey question 5 and 6

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

would this online auction website be useful to you?

42 responses



What would you prefer for payment options?

42 responses

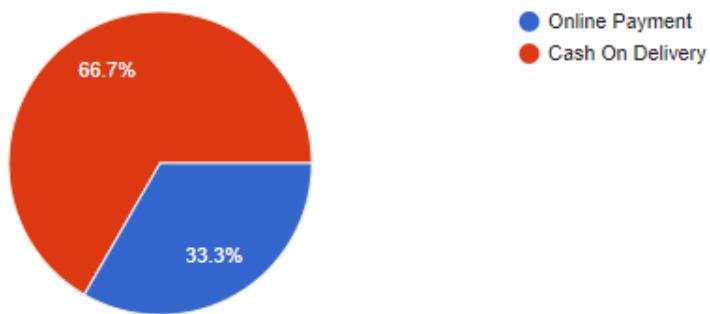
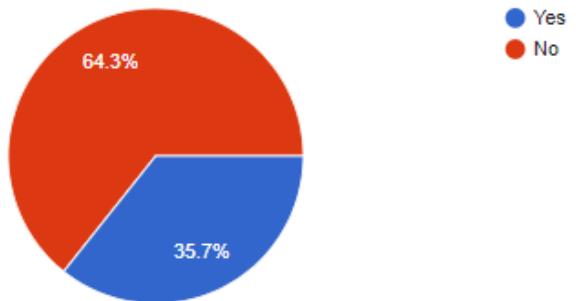


Figure 9: Survey Question 7& 8

Would you prefer to buy 2nd hand products from other user?

42 responses



Would you prefer to buy products from stores or through e commerce platforms?

42 responses

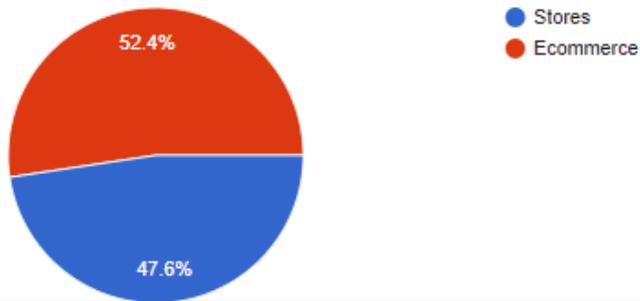


Figure 10: Survey Question 9& 10

Mainly the survey gained positive reflection and encouraged to add certain features and gave idea about how to progress and create the system

3.3.2 POST-SURVEY RESULTS

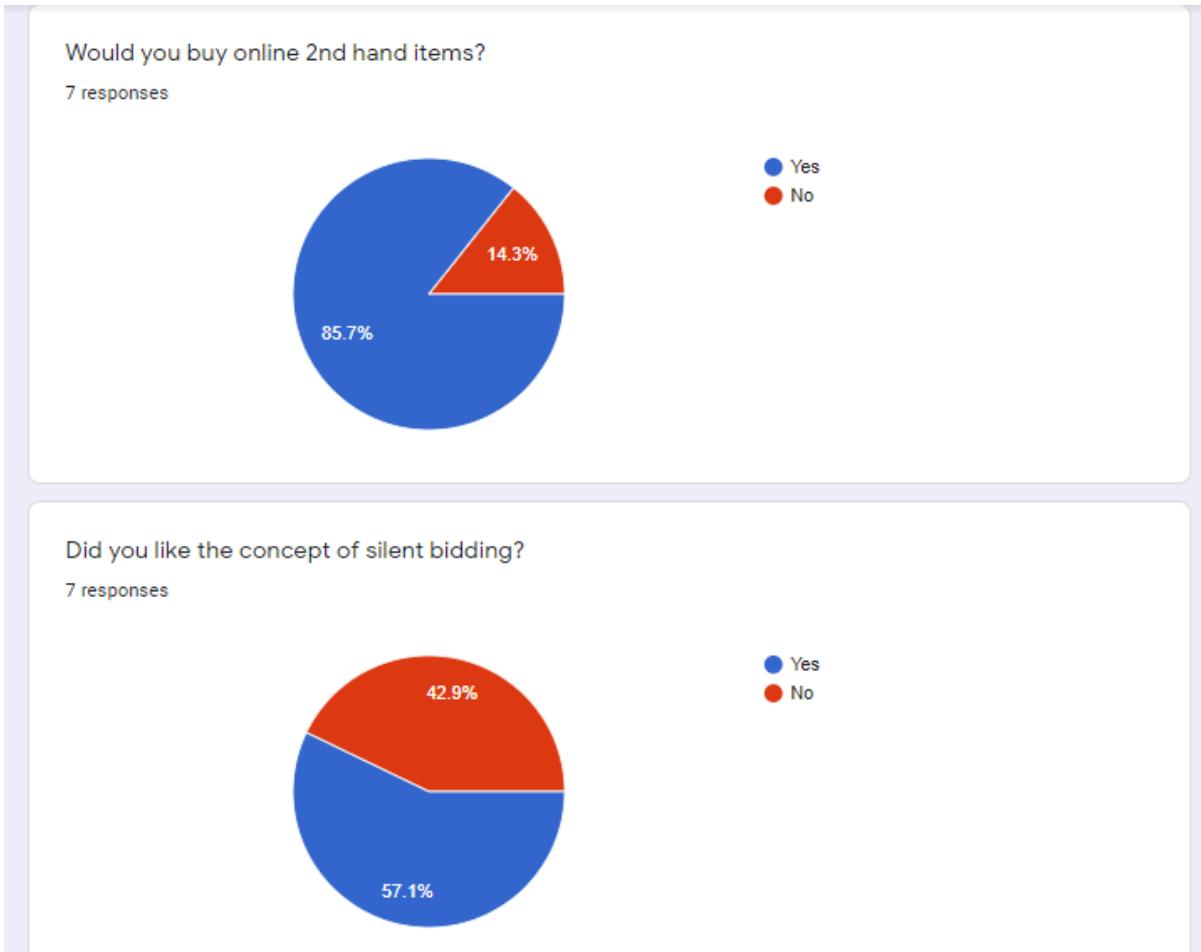
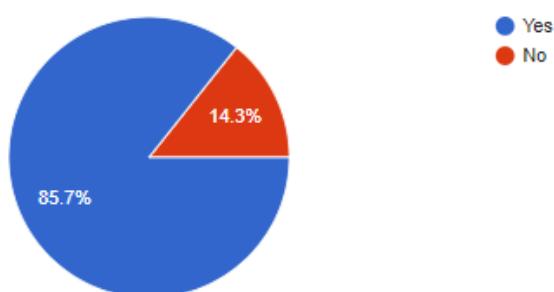


Figure 11: Post survey result 1

Do you want online payment in the system?

7 responses



do you want cash on delivery system ?

7 responses

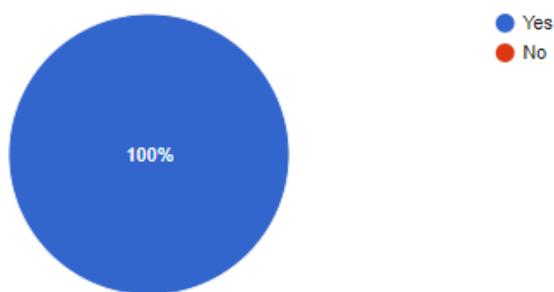
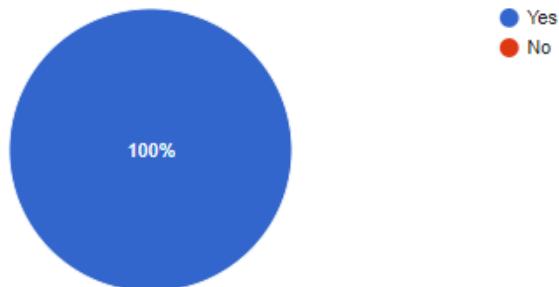


Figure 12: Post survey result 2

would you want to converse with owner during your buy?

7 responses



Would you like recommendation system of products?

7 responses

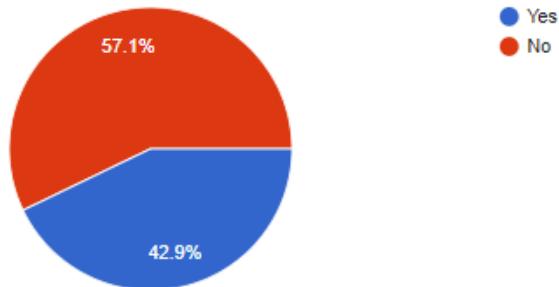


Figure 13:Post survey result 3

3.5 REQUIREMENT ANALYSIS

MoSCoW prioritization, also known as the MoSCoW method or MoSCoW analysis, is a popular prioritization technique for managing requirements. The method is commonly used to help key stakeholders understand the significance of initiatives in a specific release. The acronym, MoSCoW, stands for 4 different categories of initiatives: must-haves, should-haves, could-haves, and will not have at this time. Sometimes, the “W” in MoSCoW is used to stand for “wish” instead of “will not have right now.” (ProductPlan , 2020)

MoSCoW Prioritization parts	Functionality
Must Have (M)	Registration Login portal View products without login Add products Bid in products Fill out the shipping and billing details Payment through cash on delivery option
Should Have (S)	Shop - Cart system Update products Add time frame for auction Auction cart system
Could Have (C)	Add favourite products View history Notification system Online payment Conversation about the products in product page.
Will Not Have (W)	Overall messaging system for all the user Forum to request products Buy/ products as guest user

Table 8: MoSCoW Prioritization parts

3.6 DESIGN TECHNIQUES

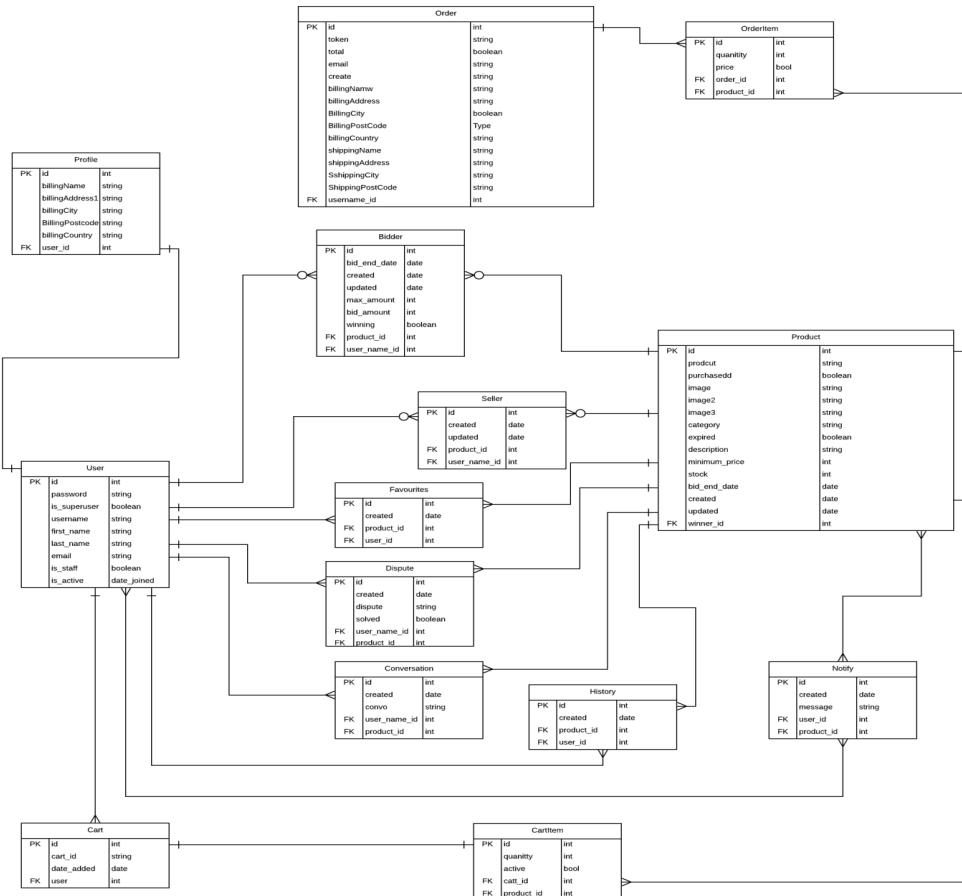


Figure 14: ER diagram

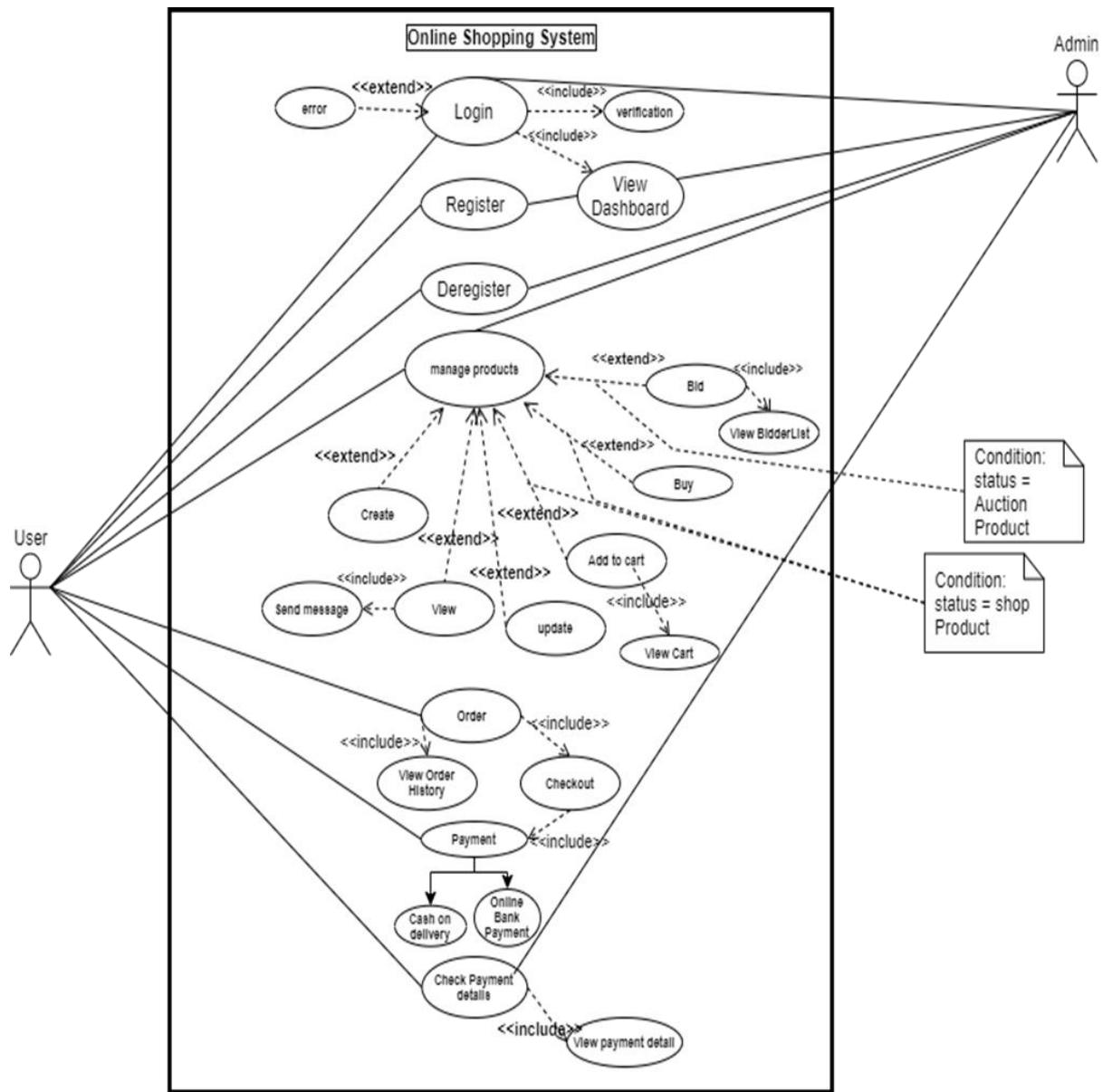


Figure 15: Use Case Diagram

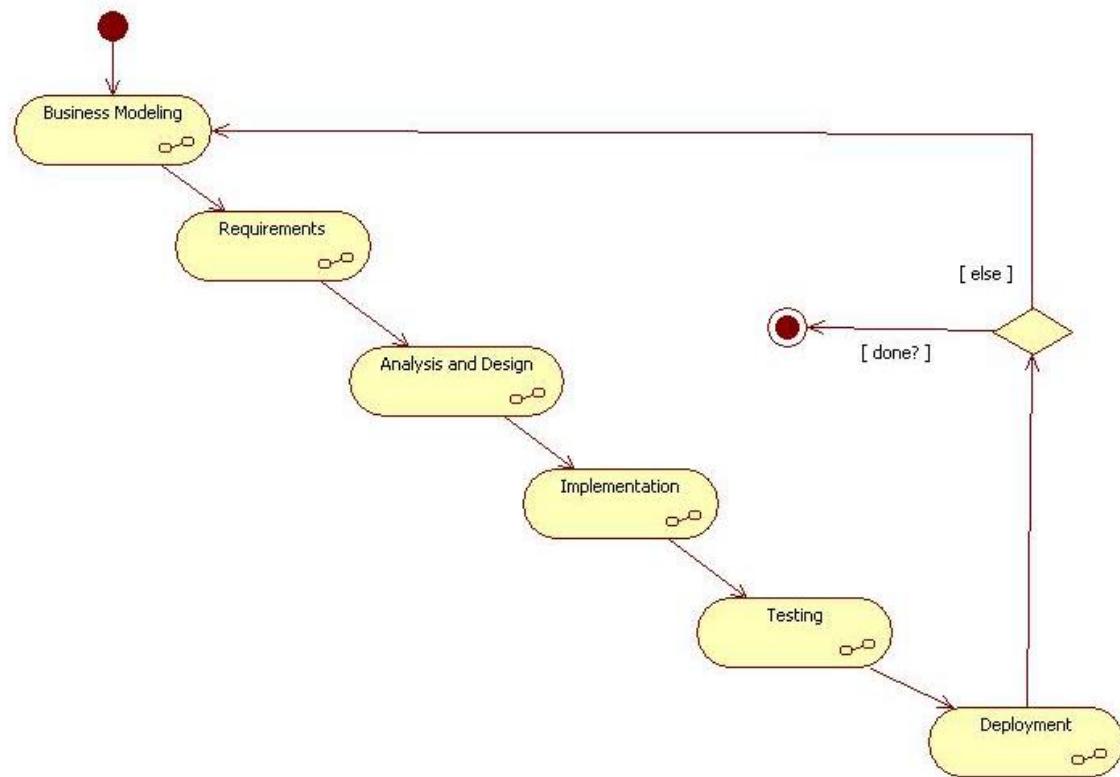


Figure 16: Iteration Workflow

3.7 IMPLEMENTATION

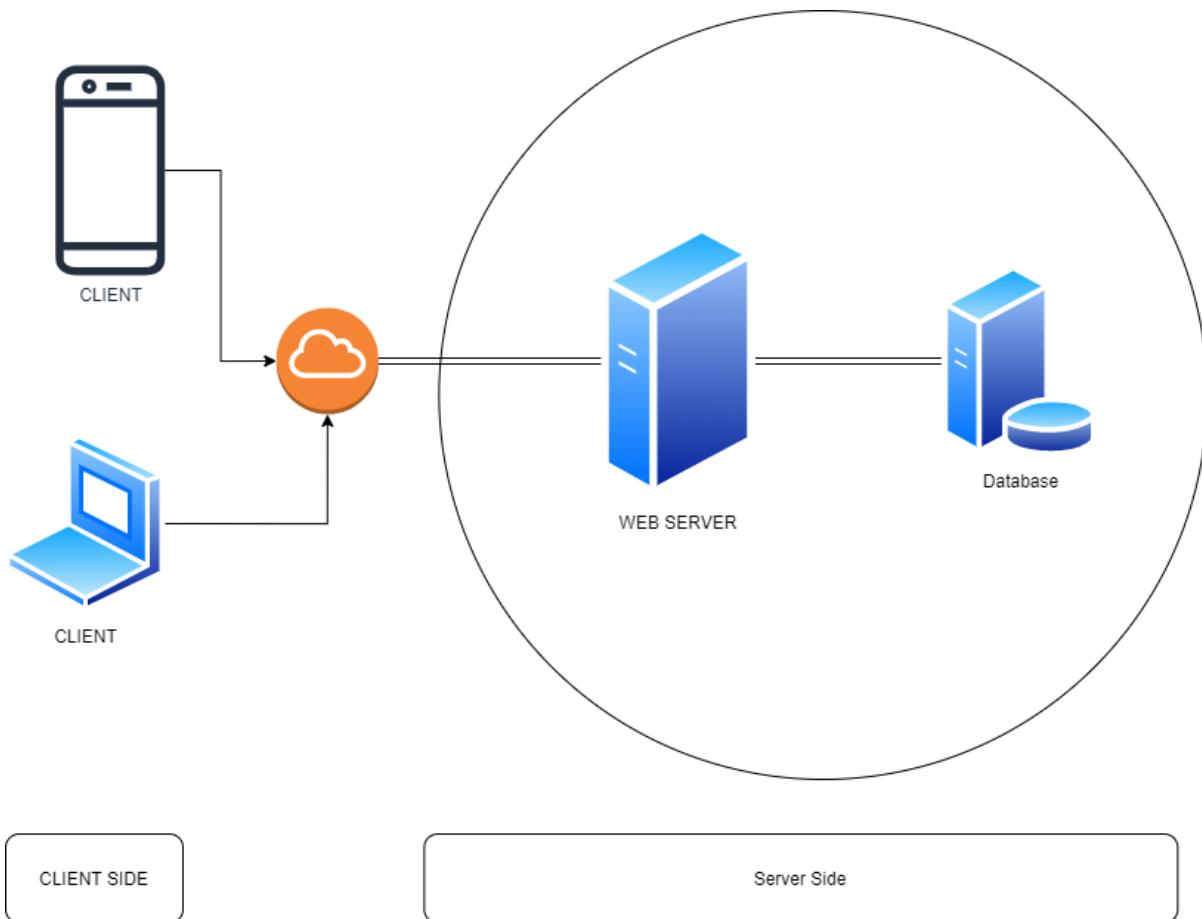


Figure 17: System architecture

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows the login and registration interface for the Samanghar platform. At the top, there is a navigation bar with links for Home, About, Products, Product Search, Login, and Register. The main area contains fields for Username, First name, Last name, Email, and Password. Below the password field is a list of password requirements. On the right side, there is a promotional message for activating Windows.

Username: SiddharthaB
First name: Sid
Last name: Bhattarai
Email: siddharthaB@gmail.com
Password: *****

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Activate Windows
Go to Settings to activate Windows.

Figure 18: Login

The screenshot shows the user profile page for 'SiddharthaB'. It displays basic user information (Name, Email, Role) and a section for managing the billing address. The billing address is set to 'Mid Baneshwor, Kathamndu, 44600'. There is a link to update the billing address.

updated

SiddharthaB
Name Sid Bhattarai
[Change Password](#)
Email siddharthaB@gmail.com
Role User

Billing Name:	Siddhartha
Billing Address:	Mid Baneshwor
Billing City:	Kathamndu
Post Code:	44600
Update Billing Address	

Figure 19:Profile

Order Number	Order Date	Total Amount	Status	Action
27	04 Jun 2020	22.00	✓ Complete	View Order27

Figure 20:Order

The screenshot shows a product detail page for 'Garnier Men NEW AcnoFight Pimple Clearing Whitening Cream'. The page includes a thumbnail image of the product packaging, an 'Add to Cart' button, and a detailed product description table.

Product Name :	Garnier Men acno fight
Category :	Beauty Products
STATUS :	shop product
Price :	22
Description :	sfd

On the right side of the screen, there is a 'Conversations' section with a 'Send Message' input field and a 'Send Message' button. A watermark for 'Activate Windows Go to Settings to activate Windows.' is visible in the bottom right corner.

Figure 21: Shop product

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows a product detail page for a 'Bed'. The product image is a double bed with blue bedding in a room with wooden furniture and framed pictures. Below the image, product details are listed:

- Product Name :** Bed
- Category :** Home Appliances
- STATUS :** auction product
- Minimum Price :** 4334
- Bid End Date :** Jan. 10, 2021, midnight
- Winner :** None

The page also shows a 'Bidderlist' table:

UserName	Bid Amount	Winning
dummy44444	43361	✓
SAMANGHAR23	43355	✗
kushal2	43353	✗
SAMANGHAR	43352	✗
kushal	43350	✗

Below the bidderlist is a 'Conversations 1' section with a message from 'User' to 'Message'.

Figure 22:Auction Product detail

The screenshot shows a shopping cart page. At the top, it says 'Your Shopping Cart !'

Your Items	Payment
 ID:23 Unit Price : \$22	\$ 22 Remove Pay with Card Pay Cash on Delivery

A large 'Continue Shopping' button is at the bottom of the cart area.

At the bottom right, there is an 'Activate Windows' message: 'Activate Windows Go to Settings to activate Windows.'

Figure 23:Cart

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows the SAMANGHAR Super Admin Panel. The top navigation bar includes 'WELCOME, KUSHAL' with links for 'VIEW SITE / VIEW DASHBOARD CHANGE PASSWORD / LOG OUT'. The main content area is divided into three main sections: 'AUTHENTICATION AND AUTHORIZATION', 'USERS', and 'WEBSITE'. Each section contains a list of items with 'Add' and 'Change' buttons.

- AUTHENTICATION AND AUTHORIZATION:**
 - Groups: + Add, Change
 - Users: + Add, Change
- USERS:**
 - Conversations: + Add, Change
 - Favouritess: + Add, Change
 - Historys: + Add, Change
 - Notifys: + Add, Change
 - Profiles: + Add, Change
- WEBSITE:**
 - Bidders: + Add, Change
 - Carts: + Add, Change
 - Order items: + Add, Change
 - Orders: + Add, Change

On the right side, there's a sidebar titled 'Recent actions' listing various actions like 'dummy14 User', 'Product object (13) Product', etc. A message at the bottom right says 'Activate Windows Go to Settings to activate Windows.'

Figure 24:Admin page

The screenshot shows the SAMANGHAR Shop page. The top navigation bar includes 'Home', 'About', 'Products', 'Seller Dashboard', 'Product Search' with a search icon, and other links. Below the navigation, there are three product cards:

- World War Game**: Category Games, 3 items. Image: Call of Duty: World War II. Buttons: 'VIEW'.
- Air Pods**: Category Mobile & Accessories, 23 items. Image: AirPods. Buttons: 'VIEW'.
- Dell Laptop**: Category Computer Peripherals, 233 items. Image: Dell laptop. Buttons: 'VIEW'.

Pagination controls at the bottom show pages 1, 2, and 'NEXT »'.

Figure 25:Shop

The screenshot shows the SAMANGHAR Search results page. The top navigation bar includes 'Home', 'About', 'Products', 'Seller Dashboard', 'Product Search' with a search icon, and other links. On the left, there's a search bar for 'Search Category' set to 'Games' with a 'Search' button. On the right, there's another 'Product Search' bar with a search icon. Below these, there is one product card:

- Bed**: Item ID 4334. Image: A double bed in a bedroom. Buttons: 'VIEW'.

Figure 26:Search

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

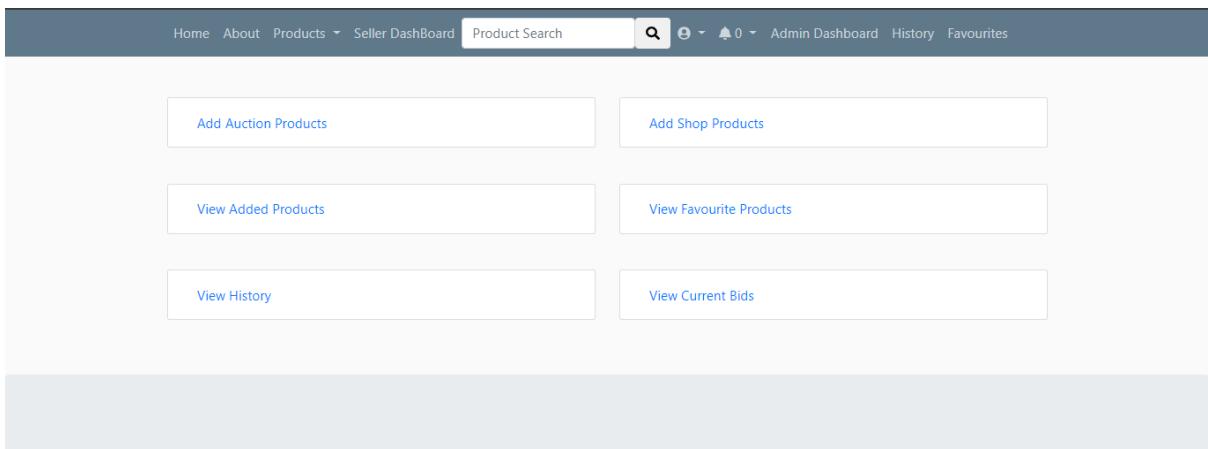


Figure 27: Functions

A screenshot of the SAMANGHAR platform's interface. At the top, there is a dark blue header bar with white text and icons. Below the header, there are two sections: 'Auction Products' and 'Shop Products'. The 'Auction Products' section displays four items with images, names, categories, and view buttons. The 'Shop Products' section shows a single item with a large blue placeholder image and a 'Activate Windows' button. A green progress bar is located between the two sections.

Product	Time	Action
PS4	May 23, 2020, 12:50 p.m.	View

Figure 28: Expired/purchased

A screenshot of the SAMANGHAR platform's interface. At the top, there is a dark blue header bar with white text and icons. Below the header, there is a table titled 'Favourite' with three columns: 'Product', 'Time', and 'View'. The table contains one row for a PS4, with the time being May 23, 2020, 12:50 p.m. and a 'View' button.

Product	Time	Action
PS4	May 23, 2020, 12:50 p.m.	View

Figure 29: Favourite

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

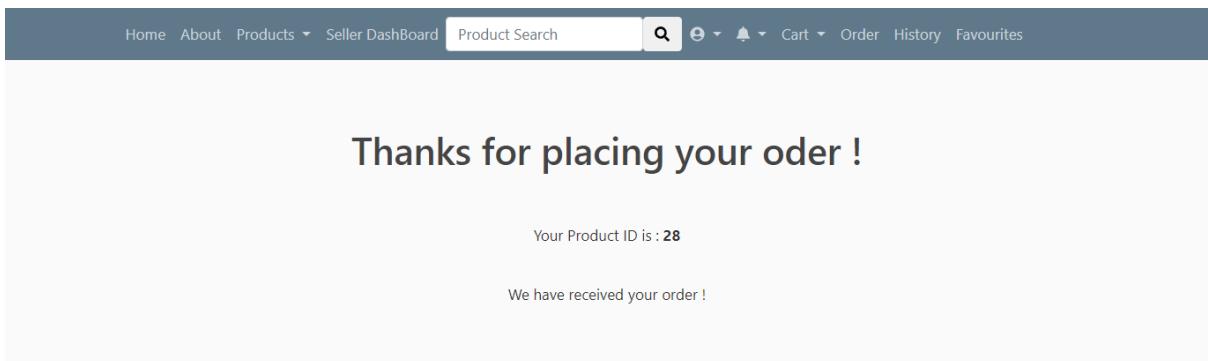


Figure 30:Placed order

#	Notification	Viewed	Product
1	You have a new bid amount for your product	Read	View Product
2	You have a new bid amount for your products	Read	View Product
3	You have bidder	Read	View Product
4	You have a lost bid amount for your products	Read	View Product
5	You have a lost a bid	Read	View Product
6	You have a lost a bid	Read	View Product

Figure 31:Notifications

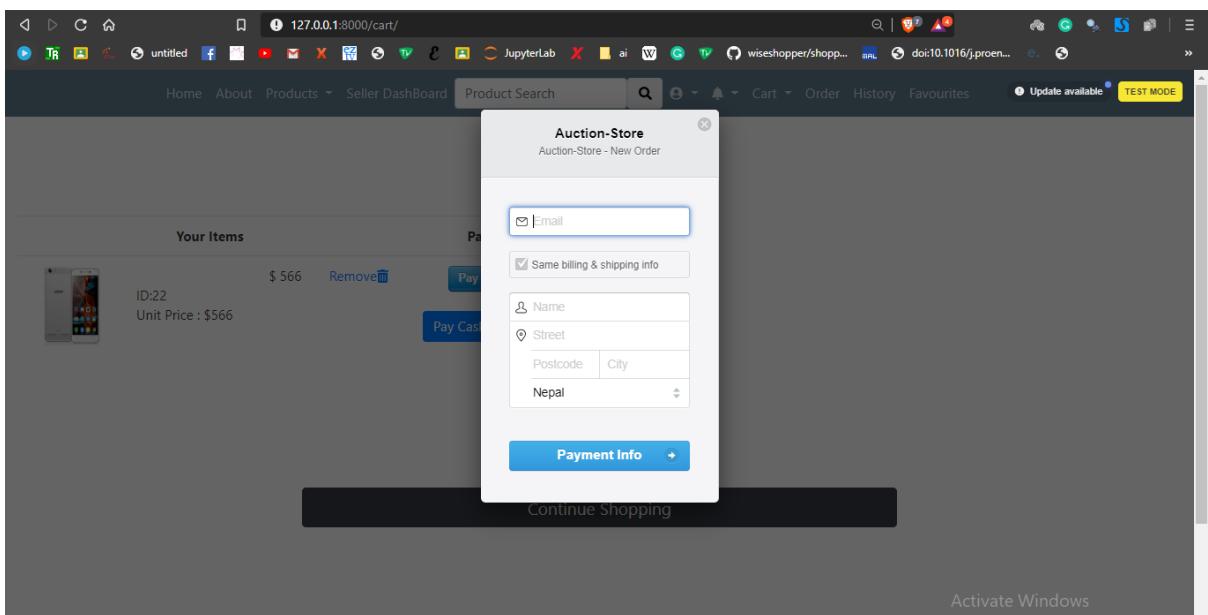


Figure 32:Stripe payment

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows a payment page for an item. On the left, there's a section titled "Your Items" showing a smartphone with ID:22 and a unit price of \$566, with a "Remove" button. To the right, there are two sets of shipping address fields. The top set is for "Billing" and the bottom set is for "Shipping". Both sets include fields for City (Baneshwor, Kathamdu), Postcode (44600), Country (Nepal), and Name (SAMANGHAR). A checkbox labeled "Same Shipping Address" is checked. At the bottom right is a blue "Order" button.

Figure 33: Payment cash on delivery

The screenshot shows an "Order Details" page. At the top, it displays order information: Order #26, Date: 24 Apr 2020, Order Total: \$67.00, and Order Status: Complete. Below this is a "Billing Address" section with the entry "Nepal". The main part of the page is a table showing the order items. The table has columns for Product Description, View, Picture, and Sub-Total. One item listed is "marvel vs capcom" with a "View" link, a small image thumbnail, and a Sub-Total of Rs67.00. At the bottom of the table, it shows a Total of \$67.00 and a Total Paid of \$67.00. The "Payment Details" section at the bottom states: "The order #26 has been paid successfully".

Order: #26	Billing Address:		
Date: 24 Apr 2020	Nepal		
Order Total: \$67.00			
Order Status: Complete.			
Product Description	View	Picture	Sub-Total
marvel vs capcom	View		Rs67.00
			Total \$67.00
			Total Paid \$67.00
Shipping Address:	Payment Details: The order #26 has been paid successfully		

Figure 34: Order details

Develop iteratively

It is best to know all requirements in advance; however, often this is not the case. Several software development processes exist that deal with providing solutions to minimize cost in terms of development phases.

Manage requirements

Always keep in mind the requirements set by users.

Use components

Breaking down an advanced project is not only suggested but in fact unavoidable. This promotes ability to test individual components before they are integrated into a larger system. Also, code reuse is a big plus and can be accomplished more easily through the use of object-oriented programming.

Model visually

Use diagrams to represent all major components, users, and their interaction. "UML", short for Unified Modeling Language, is one tool that can be used to make this task more feasible.

Verify quality

Always make testing a major part of the project at any point of time. Testing becomes heavier as the project progresses but should be a constant factor in any software product creation.

Control changes

Many projects are created by many teams, sometimes in various locations, different platforms may be used, etc. As a result, it is essential to make sure that changes made to a system are synchronized and verified constantly.

(IBM Developer, 2020)

During the start of the project, I researched about many ecommerce platforms of world like , ebay, amazon, olx. I took surveys about the ecommerce system , researched about the bidding application topic, so that I would gain information about the system and how can it be implemented.

Firstly I learned how to create a django application searching tutorials and book and seeing the documentation.

Django

Views

A view function, or view for short, is a Python function that takes a Web request and returns a Web response. This response can be the HTML contents of a Web page, or a redirect, or a 404 error, or an XML document, or an image . . . or anything, really. The view itself contains whatever arbitrary logic is necessary to return that response. (Django Software Foundation, 2020)

I have created 3 views.py file where the view function are created one in classviews.py, website/view.py and users/view.py. class views have class based views whereas other two views are function based views. Generic views was used to adding and updating the products . the generic views that was used was CreateView and UpdateView. Class based view is a callable which takes a request and returns a response. This can be more than just a function. Django provides an example of some classes which can be used as views. These allow you to structure your views. Function based Views : The view returns an HttpResponseRedirect object that contains the generated response. Each view function is responsible for returning an HttpResponseRedirect object. this view function returns an HTML page that includes the current date and time. To display this view at a particular URL. That is where urls.py file is needed

Models

A model is the single, definitive source of information about your data. It contains the essential fields and behaviours of the data you’re storing. Generally, each model maps to a single database table. Each model is a Python class that subclasses `django.db.models.Model`. Each attribute of the model represents a database field. (Django Software Foundation., 2020)

Template

Being a web framework, Django needs a convenient way to generate HTML dynamically. The most common approach relies on templates. A template contains the static parts of the desired HTML output as well as some special syntax describing how dynamic content will be inserted. (Django Software Foundation, 2020)

The RUP uses four elements to describe processes

Workers ➔ people with roles

Activities ➔ tasks that need to be done

Artefacts ➔ use cases , plans, code, test case, test result

Workflows ➔ sequence of activities

Since I am only the developer the worker is always me ‘Kushal Bhattacharai’ whether it be database developer, frontend developer , backend developer , tester. The activities that need to be done are function based using the artefacts like of use case diagram.

The workflows of RUP in each iterations are:

Business Modeling:

Requirements,

Analysis and design

Implementation

Test

Deployment

(Jacko & Stephanidis, 2003)

CHAPTER 4: TESTING AND ANALYSIS

4.1 TEST PLAN

4.1.1 UNIT TESTING, TEST PLAN

Features to be tested

1. Login
2. Register
3. Adding Products
4. Updating products
5. Adding Auction time
6. Ending Auction Products
7. Adding to carts
8. Buying Products from cash on delivery method
9. Ordering
10. Buying Products from stripe
11. Searching
12. Reading notification
13. Buying
14. Filing Dispute
15. Ordering won auction products
16. Adding conversation
17. Logging out

Test type Unit testing

Test objective The test objectives are to verify the Functionality of the system.

Test environment Anaconda environment with shell using converge

Test deliverables

Before testing phase

1. Test plans document.

2. Test cases
3. Test Design specifications.

During the testing

1. Test Tool
2. Simulators.
3. Test Data
4. Error logs

After the testing cycles is over

1. Test Results/reports

4.1.2 SYSTEM TEST PLAN

Features to be tested

18. Login
19. Adding Products
20. Updating products
21. Adding Auction time
22. Ending Auction Products
23. Adding to carts
24. Buying Products from cash on delivery method
25. Ordering
26. Buying Products from stripe
27. Searching
28. Reading notification
29. Buying
30. Filing Dispute
31. Ordering won auction products
32. Adding conversation
33. Logging out
34. Updating profile

Test type System testing

Test objective The test objectives are to verify the system.

Test environment

Objectives , Expected output , Gained output and test result will be created to store the testing document while the testing is done in debug mode.

TEST deliverables

Before testing phase

Test plans document.

During the testing

Test Data stored in form of table

After the testing cycles is over

Test Results/report

4.2 UNIT TESTING

The plan is to use Using coverage to test the system. First models are to be tested and the functions of the models need to be tested. Then the views are tested. The test that were run using shell.

```
(project) C:\Users\DELL\Desktop\FYP\django_project>coverage run manage.py test website
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
```

Figure 35: running unit test in system

```
Ran 17 tests in 26.131s

OK

Destroying test database for alias 'default'...
```

Figure 36: test completed

```
Terminal: Local × Local (2) × +
File "C:\Users\DELL\Desktop\FYP\django_project\website\views.py", line 223, in save_bid
    max = Bidder.objects.get(product_id=request.POST.get('product_id')).first()
File "c:\users\deLL\anaconda3\envs\project\lib\site-packages\django\db\models\manager.py", line 82, in manager_method
    return getattr(self.get_queryset(), name)(*args, **kwargs)
File "c:\users\deLL\anaconda3\envs\project\lib\site-packages\django\db\models\query.py", line 406, in get
    raise self.model.DoesNotExist(
website.models.Bidder.DoesNotExist: Bidder matching query does not exist.

-----
Ran 16 tests in 5.959s

FAILED (errors=1)
Destroying test database for alias 'default'...

(project) C:\Users\DELL\Desktop\FYP\django_project>
```

Figure 37: testing error of save bid because of

```
if request.method == 'POST':
    if int(request.POST['bid_amount']) <= Product.objects.get(id=request.POST['product_id']).minimum_price:
        messages.error(request, "bid amount is low")
        return HttpResponseRedirect(request.META.get('HTTP_REFERER', '/'))

    max = Bidder.objects.filter(product_id=id=request.POST.get('product_id')).first()
    if int(request.POST['bid_amount']) >= max.max_amount:
        pass
    else:
        messages.error(request, "bid amount must be higher than the latest bid")
        return HttpResponseRedirect(request.META.get('HTTP_REFERER', '/'))
```

Figure 38 :error code

```
1  from django.test import TestCase
2
3  # Create your tests here.
4  from django.urls import reverse
5
6  from users.forms import UserRegisterForm
7  from .models import *
8
9  from users.models import Profile, Notify
10
11
12  class RegisterProfile2Test(TestCase):
13
14      def create_user(self):
15          return User.objects.create(username='kushal')
16
17  def test_create_user(self):
18      a = self.create_user()
19      b = Profile.objects.last()
20      self.assertTrue(isinstance(a, User))
21      self.assertTrue(isinstance(b, Profile))
22      self.assertEqual(a, b.user)
23
```

Figure 39:unit test code for testing user creating

Here testing of creating users and profile are done.

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

```
class RegisterProfileTest(TestCase):

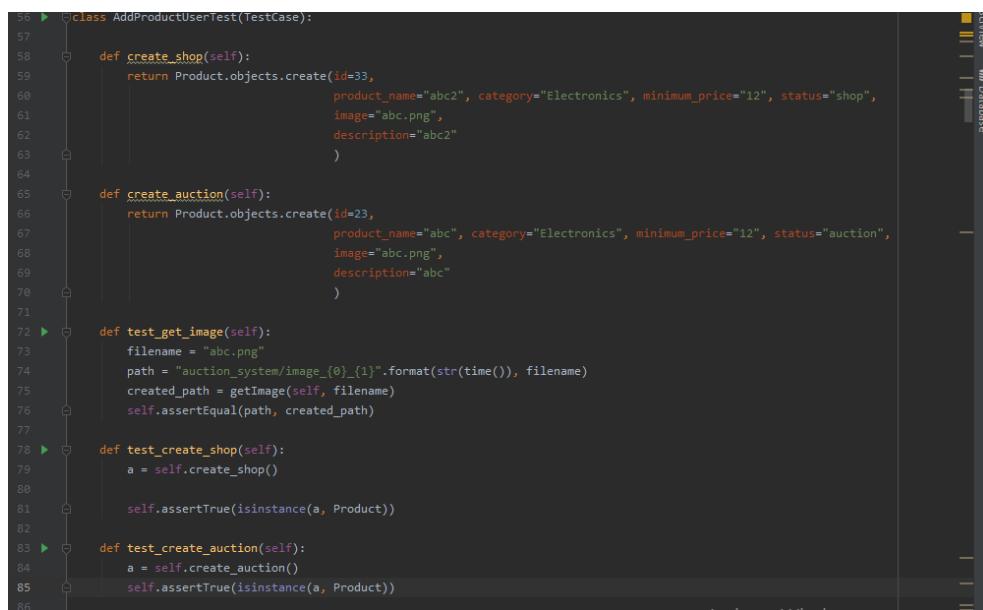
    def create_user(self):
        return User.objects.create(username='kushal')

    def test_create_user(self):
        a = self.create_user()
        b = Profile.objects.last()
        self.assertTrue(isinstance(a, User))
        self.assertTrue(isinstance(b, Profile))
        self.assertEqual(a, b.user)

    def test_object_name(self):
        user = self.create_user()

        b = Profile.objects.last()
        expected_object_name = f'{user.username} Profile'
        self.assertEquals(expected_object_name, str(b))
        print(expected_object_name)
```

Figure 40: register user and checking object name



A screenshot of the PyCharm IDE interface. The main area shows Python test code for adding products. The code defines a class `AddProductUserTest` with methods for creating shop and auction products, and tests for getting images and creating shop/auction objects. The code uses assertions to check if objects are instances of `Product`. The right side of the interface features standard PyCharm toolbars for 'File', 'Edit', 'View', 'Database', and others.

```
56 ► class AddProductUserTest(TestCase):
57
58     def create_shop(self):
59         return Product.objects.create(id=33,
60                                         product_name="abc2", category="Electronics", minimum_price="12", status="shop",
61                                         image="abc.png",
62                                         description="abc2")
63
64     def create_auction(self):
65         return Product.objects.create(id=23,
66                                         product_name="abc", category="Electronics", minimum_price="12", status="auction",
67                                         image="abc.png",
68                                         description="abc")
69
70     def test_get_image(self):
71         filename = "abc.png"
72         path = "auction_system/image_{0}_{1}".format(str(time()), filename)
73         created_path = getImage(self, filename)
74         self.assertEqual(path, created_path)
75
76     def test_create_shop(self):
77         a = self.create_shop()
78
79         self.assertTrue(isinstance(a, Product))
80
81     def test_create_auction(self):
82         a = self.create_auction()
83
84         self.assertTrue(isinstance(a, Product))
```

Figure 41: Testing adding product in both shop and auction

Here the testing of creating shop products and auctions

```
88 ►  class HomeViewTest(TestCase):
89
90 ►  def test_home(self):
91     url = reverse("home")
92     resp = self.client.get(url)
93     print(resp, 'a')
94     self.assertEqual(resp.status_code, 200)
95
96 ►  def test_website_home(self):
97     url = reverse("website-home")
98     resp = self.client.get(url)
99     print(resp, 'b')
100    self.assertEqual(resp.status_code, 200)
101
102
103 ►  class ShopViewTest(TestCase):
104 ►  def test_shop(self):
105     url = reverse("shop")
106     resp = self.client.get(url)
107     print(resp, 'c')
108     self.assertEqual(resp.status_code, 200)
109
```

Figure 42: HomeView and ShopView test without login

The test of home, website-home, shop without login.

```

111 ►  class AuctionViewTest(TestCase):
112
113 ►      def test_auction(self):
114          url = reverse("auction")
115          resp = self.client.get(url)
116          print(resp, 'd')
117          self.assertEqual(resp.status_code, 200)
118
119
120 ►  class ExpiredHomeViewTest(TestCase):
121
122 ►      def test_expiredHome(self):
123          url = reverse("expired")
124          resp = self.client.get(url)
125          print(resp, 'e')
126          self.assertEqual(resp.status_code, 200)
127
128
129 ►  class LoginViewTest(TestCase):
130
131 ►      def test_login(self):
132          url = reverse("login")
133          resp = self.client.get(url)
134          print(resp, 'f')
135          self.assertEqual(resp.status_code, 200)
136

```

Figure 43: Auction , expired without login and login testing

```

class NotificationViewTest(TestCase):

    def test_notification(self):
        url = reverse("viewnotify")
        resp = self.client.get(url)
        print(resp, 'h')
        self.assertEqual(resp.status_code, 302)

class MyBidsViewTest(TestCase):

    def test_mybids(self):
        url = reverse("mybids")
        resp = self.client.get(url)
        print(resp, 'i')
        self.assertEqual(resp.status_code, 302)

```

Figure 44: Notification, and bids without login test

Notification, and bids without login test are being tested.

```

class LogInTest(TestCase):

    def setUp(self):
        self.credentials = {
            'username': 'testuser',
            'password': 'secret'}
        User.objects.create_user(**self.credentials)

    def test_login(self):
        # send login data
        response = self.client.post('/login/', self.credentials, follow=True)
        # should be logged in now
        print(response, 'j')
        self.assertTrue(response.context['user'].is_authenticated)

```

Figure 45: Login testing with credentials and with login url

Login in tested here while setting credentials.

```

class AllTestViewTest(TestCase):

    def setUp(self):
        self.credentials = {
            'username': 'testuser',
            'password': 'secret'}
        User.objects.create_user(**self.credentials)

    def test_login(self):
        # send login data
        response = self.client.post('/login/', self.credentials, follow=True)
        # should be logged in now

        self.assertTrue(response.context['user'].is_authenticated)
        url = reverse('home')
        resp = self.client.get(url)
        self.assertEqual(resp.status_code, 200)
        url = reverse("website-home")
        resp = self.client.get(url)
        self.assertEqual(resp.status_code, 200)
        url = reverse("expired")
        resp = self.client.get(url)

```

Figure 46: testing login , home website home url in same session after logged in

Testing of expired, home, website-home and authenticated is done.

```
self.assertEqual(resp.status_code, 200)
url = reverse("shop")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("auction")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("home")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("viewnotify")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("mybids")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("all-product")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("won")
resp = self.client.get(url)
```

Figure 47:testing in same logged in session

Testing of shop, auction, home, viewnotify,mybids, all-prodcut,won is done here.

```
url = reverse("profile")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("post-create")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("shopcreate")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
Product.objects.create(id=1,
                      product_name="abc2", category="Games", minimum_p
                      image="abc.png",
                      description="abc2"
                      )
Product.objects.create(id=2,
                      product_name="abc23", category="Games", minimum_p
                      status="auction",
                      image="abc3.png",
                      description="abc23"
                      )
Product.objects.create(id=3,
                      product_name="abc23", category="Games", minimum_p
                      status="auction",
                      image="abc3.png",
```

Figure 48:same logged in session with different url and creating product test

Adding product testing in login.

```

Seller.objects.create(product_id_id=1, user_name_id=1)
Seller.objects.create(product_id_id=2, user_name_id=1)
Seller.objects.create(product_id_id=3, user_name_id=1)

url = reverse("product_detail", args=[1])
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("updateAcc", args=[1])
resp = self.client.get(url)

self.assertEqual(resp.status_code, 200)
url = reverse("search")
resp = self.client.get(url, data={'data': 'category', 'q': 'Games'})

self.assertEqual(resp.status_code, 200)
url = reverse("search")
resp = self.client.get(url, data={'data': 'product', 'q': 'abc2'})

self.assertEqual(resp.status_code, 200)
url = reverse("search")
resp = self.client.get(url, data={'q': ''})

self.assertEqual(resp.status_code, 200)
url = reverse("addtime", args=[2])

resp = self.client.post(url, data={'time': 'twelve', 'pk': 2})

```

Figure 49: testing other urls in same logged in session

Testing of viewing of product. Updating account is tested, searching category games as well as abc2 product as well as empty search. The testing of adding time is also done.

```

        resp = self.client.post(url, data={'time': 'twelve', 'pk': 2})

    self.assertEqual(resp.status_code, 302)
    url = reverse("addtime", args=[2])
    resp = self.client.post(url, data={'time': 'one', 'pk': 2})

    self.assertEqual(resp.status_code, 302)
    url = reverse("addtime", args=[2])
    resp = self.client.post(url, data={'time': 'two', 'pk': 2})

    self.assertEqual(resp.status_code, 302)
    url = reverse("message")
    resp = self.client.post(url, data={'p_id': 2, 'convo': "hi please bid faster i need money ASAP"})

    self.assertEqual(resp.status_code, 302)
    url = reverse("update", args=[2])
    resp = self.client.get(url)

    self.assertEqual(resp.status_code, 200)
    url = reverse("updateshop", args=[1])
    resp = self.client.get(url)

    self.assertEqual(resp.status_code, 200)

    url = reverse("addfavourite")
    resp = self.client.post(url, data={'pk': 2})
    self.assertEqual(resp.status_code, 302)

```

Figure 50: testing other urls, and updating sending message in same logged in session

Testing of updating sop, add-favourite, orderhistory, addcart, cart, cart remove sending message is done here.

```

    self.assertEqual(resp.status_code, 302)
    url = reverse("order_history")
    resp = self.client.post(url)
    self.assertEqual(resp.status_code, 200)
    url = reverse("cart")
    resp = self.client.get(url)
    self.assertEqual(resp.status_code, 200)
    url = reverse("add_cart", args=[1])
    resp = self.client.post(url)
    self.assertEqual(resp.status_code, 302)
    url = reverse("cart")
    resp = self.client.get(url)
    self.assertEqual(resp.status_code, 200)
    url = reverse("cart_remove", args=[1])
    resp = self.client.get(url)
    self.assertEqual(resp.status_code, 302)

    url = reverse("logout")
    resp = self.client.get(url)

```

Figure 51: testing urls in the same logged in session and login out

```
self.credentials2 = {
    'username': 'testusess',
    'password': 'secretss',
    'is_staff': True,
    'is_superuser': True,
}

User.objects.create_user(**self.credentials2)
resp = self.client.post('/verifylogin/', self.credentials2, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("add_cart", args=[1])
resp = self.client.post(url)
self.assertEqual(resp.status_code, 302)
url = reverse("sellDashboard")
resp = self.client.post(url)
self.assertEqual(resp.status_code, 200)
url = reverse("logout")
resp = self.client.get(url)
```

Figure 52: Testing Adding to cart

Testing logged in with superadmin adding cart and sellerdashboard. Logging in with simple credentials and doing save_bid.

```
self.assertEqual(resp.status_code, 302)
self.credentialsne = {
    'username': 'testy',
    'password': 'secretss',
}
User.objects.create_user(**self.credentialsne)
resp = self.client.post('/verifylogin/', self.credentialsne, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 4000})
self.assertEqual(resp.status_code, 302)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 3000})
self.assertEqual(resp.status_code, 302)
url = reverse("logout")
resp = self.client.get(url)
```

Figure 53: Testing bidding

```
self.credentials4 = {
    'username': 't',
    'password': 'secretss',
}

User.objects.create_user(**self.credentials4)
resp = self.client.post('/verifylogin/', self.credentials4, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 4000})
self.assertEqual(resp.status_code, 302)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 4009})
self.assertEqual(resp.status_code, 302)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 6000})
self.assertEqual(resp.status_code, 302)
url = reverse("logout")
resp = self.client.get(url)

self.assertEqual(resp.status_code, 302)
```

Figure 54: Test rebidding

```

self.assertEqual(resp.status_code, 302)
url = reverse("register")

resp = self.client.post(url,
                       data={'email': "samanghar@gmail.com", 'username': "saman", 'first_name': "kushal",
                             'last_name': "bhattacharai", 'role': "Admin", 'password1': "asid1234",
                             'password2': "asid1234", 'is_staff': True}, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("register")
resp = self.client.post(url,
                       data={'email': "samanghar3@gmail.com", 'username': "saman3", 'first_name': "kushal",
                             'last_name': "bhattacharai", 'role': "SuperAdmin", 'password1': "asid1234",
                             'password2': "asid1234", 'is_staff': True}, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("register")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("register")
resp = self.client.post(url,
                       data={'email': "samanghar2@gmail.com", 'username': "saman2", 'first_name': "kushal",
                             'last_name': "bhattacharai", 'role': "User", 'password1': "asid1234",
                             'password2': "asid1234", 'is_staff': False}, follow=True)
self.assertEqual(resp.status_code, 200)
resp = self.client.post(url,
                       data={'email': "samanghar@gmail.com", 'username': "saman", 'first_name': "kushal",
                             'last_name': "bhattacharai", 'role': "User", 'password1': "asid1234",
                             'password2': "asid123", 'is_staff': False}, follow=True)
self.assertEqual(resp.status_code, 200)

```

Figure 55: Register test or user Admin ,SuperAdmin ,Admin and wrong password

Testing of registration of different roles like user, superadmin and admin. Testing of verifylogin function.

```

self.credentials55 = {
    'username': 'saman3',
    'password': 'asid1234',

}
resp = self.client.post('/verifylogin/', self.credentials55, follow=True)
self.assertEqual(resp.status_code, 200)
resp = self.client.get('/verifylogin/')
self.assertEqual(resp.status_code, 302)
self.credentials4 = {
    'username': 'kushalnew',
    'password': 'kushal1234',

}
resp = self.client.post('/verifylogin/', self.credentials4, follow=True)
self.assertEqual(resp.status_code, 200)

```

Figure 56: Test for verifying login

```
self.credentials5 = {
    'username': 'testusersss',
    'password': 'secretss',
}

User.objects.create_user(**self.credentials5)
resp = self.client.post('/verifylogin/', self.credentials5, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 5000})
self.assertEqual(resp.status_code, 302)
url = reverse("winning")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("cart")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("add_cart", args=[1])
resp = self.client.post(url)
self.assertEqual(resp.status_code, 302)
url = reverse("admin-home")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 302)
url = reverse("logout")
resp = self.client.get(url)
```

Figure 57: Testing different urls

Testing savebid, winning, cart, adiing to cart ,admin- home functions.

```
resp = self.client.post('/verifylogin/', self.credentials2, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("end", args=[3])
resp = self.client.post(url)
self.assertEqual(resp.status_code, 302)
url = reverse("end", args=[2])
resp = self.client.post(url)
self.assertEqual(resp.status_code, 302)
url = reverse("message")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 302)
url = reverse("favourites")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("history")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("admin-home")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("winner")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("home")
resp = self.client.get(url)
self.assertEqual(resp.status_code, 200)
url = reverse("logout")
resp = self.client.get(url)
```

Figure 58: Testing functions like end, history, favourites

Testing of ending products auction, sending empty message. History , favoutire, winner, admin home and home and testing of logout.

```
self.assertEqual(resp.status_code, 302)
self.credentialsnew = {
    'username': 'testuses',
    'password': 'secretss',
}
response = self.client.post('/verifylogin/', self.credentialsnew, follow=True)

resp = self.client.get(url)
self.assertEqual(resp.status_code, 302)
self.credentials3 = {
    'username': 'test',
    'password': 'secretss',
    'is_staff': True,
}
User.objects.create_user(**self.credentials3)
resp = self.client.post('/verifylogin/', self.credentials3, follow=True)
self.assertEqual(resp.status_code, 200)
url = reverse("save_bid")
resp = self.client.post(url, data={'product_id': 2, 'bid_amount': 5000}, follow=True)
self.assertEqual(resp.status_code, 200)

url = reverse("logout")
resp = self.client.get(url)
```

Figure 59: testing login, and testing if staff can bid or not.

Testing of credentiils and saving bid of staff where the staff cannot bid.

4.3 SYSTEM TESTING

TEST 1	
Action	Enter invalid username and password
Expected output	Error and not being able to login. Redirects to login page
Actual output	Error and not being able to login. Redirects to login page
Test Result	Pass

Table 9: Test 1

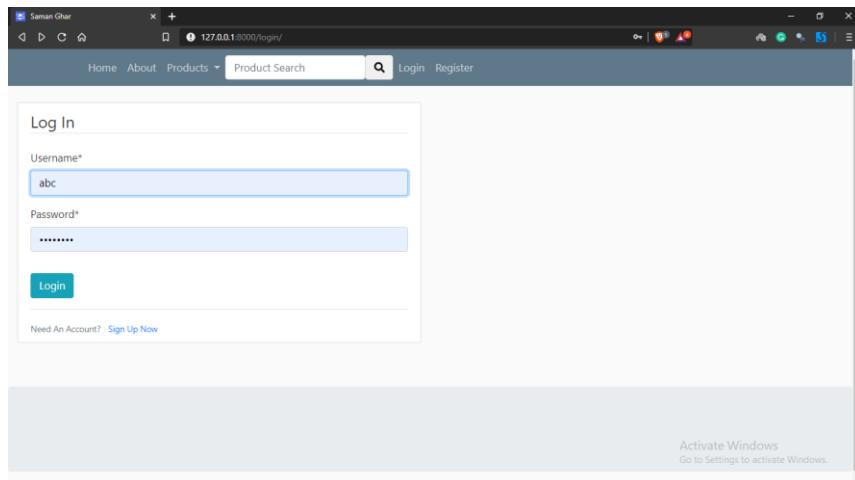


Figure 60: login test

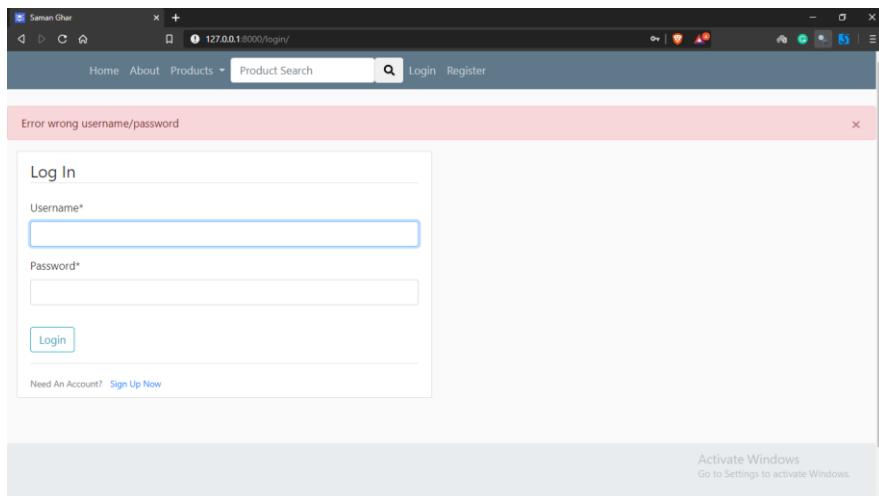


Figure 61: test result

TEST 2	
Action	Enter deactivated profile username and password.
Expected output	Not being able to login and redirect to login page
Actual output	Not being able to login and redirect to home page
Test Result	Pass

Table 10: Test 2

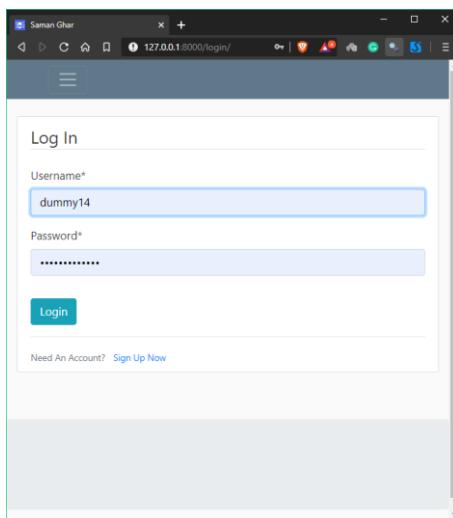


Figure 62: testing 2

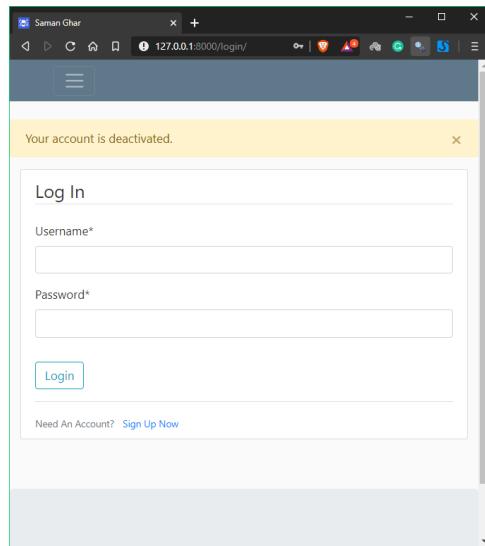


Figure 63: test result -2

TEST 3	
Action	Enter valid username and password
Expected output	Being able to login. Redirects to login page
Actual output	Being able to login. Redirects to home page
Test Result	Pass

Table 11:Test 3

Figure 64 : login form

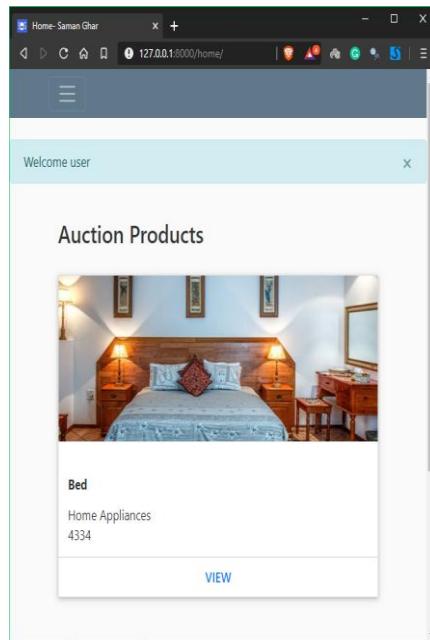


Figure 65:test result

TEST 4	
Action	Add products
Expected output	New Product Added
Actual output	New Product Added
Test Result	Pass

Table 12: Test 4

Figure 66: add product form

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

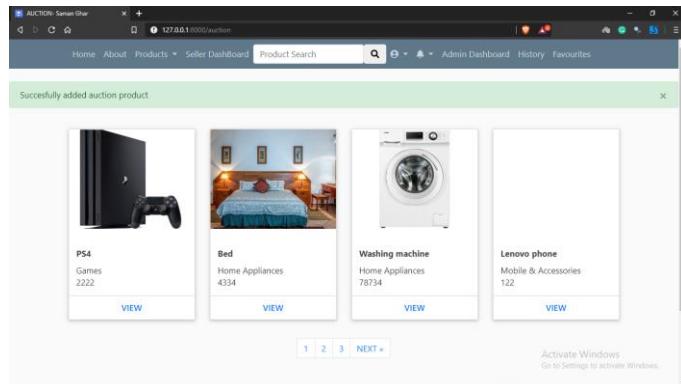


Figure 67:added products

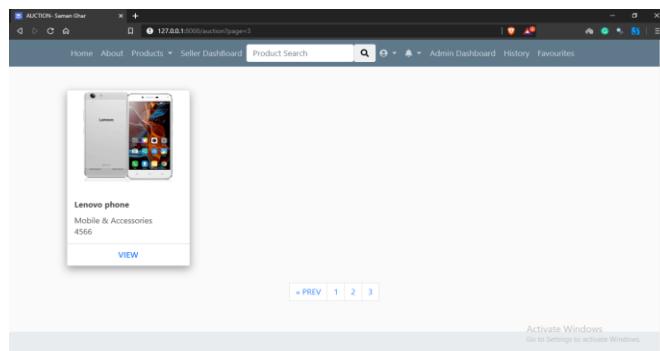


Figure 68:displayed new products

TEST 5	
Action	Update products
Expected output	Product updated
Actual output	Product updated
Test Result	Pass

Table 13: Test 5

The screenshot displays a web-based application interface for managing products and bids. On the left, a modal window titled "Product update" shows a thumbnail of a Garnier Men AcneFight product and its details: Product Name: Lenovo phone, Category: Beauty Products, Status: auction product, Minimum Price: 449, Bid End Date: June 4, 2020, 11:27 a.m., and Winner: dummy44444. On the right, there are two sections: "Bidderlist" which shows a single entry for dummy44444 with a bid amount of 449 and a checked winning status; and "Conversations" which includes a "Send Message" input field and a "Send Message" button.

UserName	Bid Amount	Winning
dummy44444	449	✓

Figure 69:Product update 1

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Product name:
Lenovo phone

Minimum price:
449

Bid end date:
2020-06-25 19:42:27

Image: Currently: auction_system/image_1587811443.7398207_image_1574944478.0575385_2.PNG

Change:
 No file chosen

Image2:
 No file chosen

Image3:
 No file chosen

jb

Activate Windows
Go to Settings to activate Windows.

Figure 70:Product update 2

updated

Product update

GARNIER MEN NEW AcneFight Pimple Clearing Cream

Product Name : Acne Cream
Category : Beauty Products
STATUS : auction product

UserName	Bid Amount	Winning
dummy44444	449	✓

Bidderlist

Conversations

Send Message

Send Message

Activate Windows
Go to Settings to activate Windows.

Figure 71:Product update 3

TEST 6	
Action	Add Auction time
Expected output	Auction time added
Actual output	Auction time added
Test Result	Pass

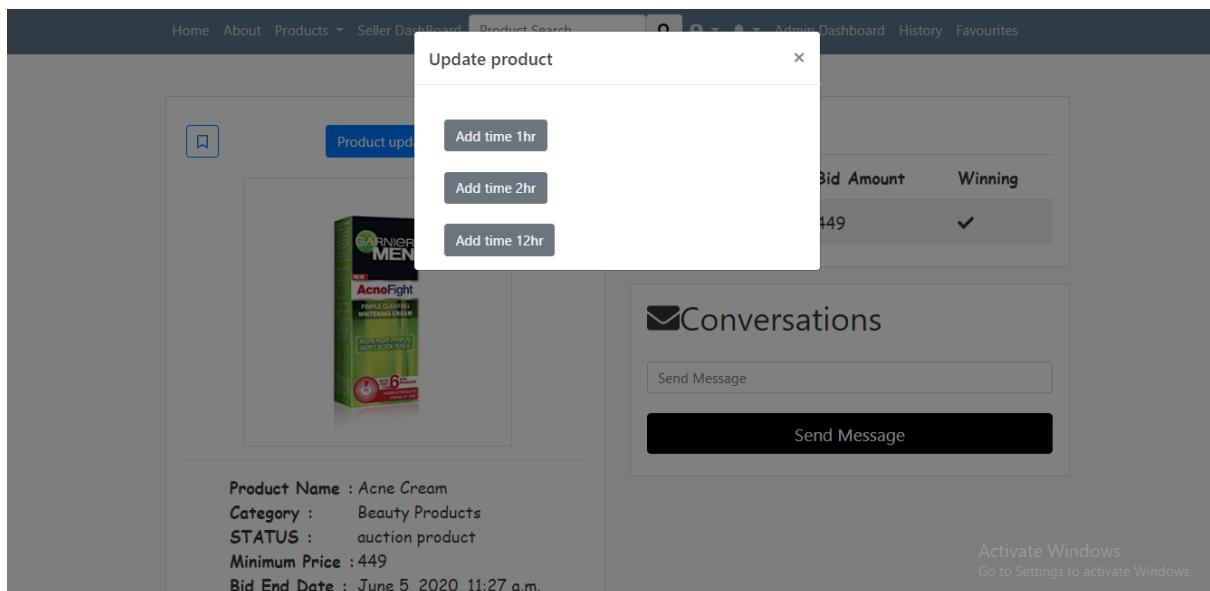


Figure 72:add 12 hr

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows a web-based auction platform. On the left, a product card for 'GARNIER MEN NEW AcnoFight PURPLE CLEANSING WHITENING CREAM' is displayed with a 'Product update' button. Below the card, product details are listed: Product Name: Acne Cream, Category: Beauty Products, STATUS: auction product, Minimum Price: 449, and Bid End Date: June 5, 2020, 11:27 p.m. On the right, a 'Bidderlist' section shows a single entry: dummy44444 with a Bid Amount of 449 and a checkmark in the Winning column. Below it is a 'Conversations' section with a 'Send Message' button. A small 'Activate Windows' message is visible at the bottom right.

Figure 73: added 12 hr 11:27 am to 11:27 pm

TEST 7	
Action	End auction products
Expected output	Auction ended
Actual output	Auction ended
Test Result	Pass

Table 14: Test 6

This screenshot is similar to Figure 73 but includes additional buttons below the product details: 'Update', 'End Auction', and 'Winner'. The 'End Auction' button is highlighted with a red box. The rest of the interface and data are identical to Figure 73.

Figure 74: end auction

The screenshot shows a web-based auction platform. On the left, a 'Product update' card displays a box of 'SAMANGHAR MEN Acne Fight' cream. Below the image are product details: **Product Name**: Acne Cream, **Category**: Beauty Products, **STATUS**: auction product, **Minimum Price**: 449, **Bid End Date**: June 5, 2020, 7:59 a.m., and **Winner**: dummy44444. On the right, a 'Bidderlist' section shows a single entry: dummy44444 with a bid amount of 449 and a checkmark under 'Winning'. Below it is a 'Conversations' section with a 'Send Message' button.

Figure 75:Auction ended

TEST 8	
Action	Add to cart products
Expected output	Product added to cart
Actual output	Product added to cart
Test Result	Pass

Table 15: Test 8

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Product Name : Garnier Men acno fight
 Category : Beauty Products
 STATUS : shop product
 Price : 22
 Description : sfd

Figure 76: Add to Cart

Your Items	Payment
ID:23 Unit Price : \$22	\$ 22 Remove Pay with Card Pay Cash on Delivery

Figure 77: Added to cart

TEST 9	
Action	Payment by cash
Expected output	Order created
Actual output	Order created

Test Result	Pass
--------------------	-------------

Figure 78: Test 9

CASH ON DELIVERY PAYMENT ORDER

USD Order Total:
22

User name:
SiddharthaB

Email Address:
siddharthaB@gmail.com

BillingName:
Siddhartha

BillingAddress1:
MID BANESHWOR

BillingCity:
Kathamndu

BillingPostcode:
44600

BillingCountry:
Nepal

Figure 79: cart cash on delivery

BillingCity:
Kathamndu

BillingPostcode:
44600

BillingCountry:
Nepal

ShippingName: Same Shipping Address
Siddhartha

ShippingAddress1:

ShippingCity:

ShippingPostcode:

ShippingCountry:
Nepal

Order

Figure 80: filling form

The screenshot shows the 'Your Items' section with a total of 22 items. On the right, there is a form for entering shipping details. The 'BillingCity:' field contains 'Kathamndu'. The 'BillingPostcode:' field contains '44600'. The 'BillingCountry:' field contains 'Nepal'. Under 'Shipping', the 'ShippingName:' field contains 'Siddhartha', and the 'Same Shipping Address' checkbox is checked. The 'ShippingAddress1:' field contains 'MID BANESHWOR'. The 'ShippingCity:' field contains 'Kathamndu'. The 'ShippingPostcode:' field contains '44600'. The 'ShippingCountry:' field contains 'Nepal'. A blue 'Order' button is at the bottom.

Figure 8I:From filled

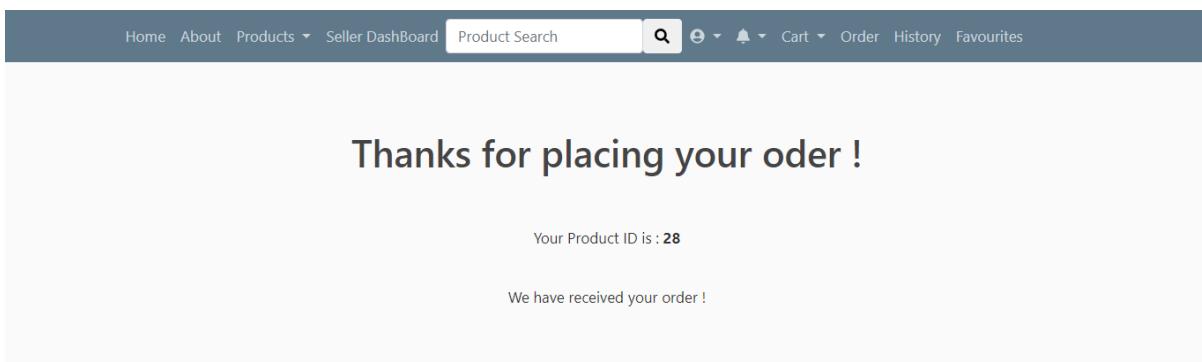


Figure 82:Order placed

TEST 10	
Action	View order detail
Expected output	View order detail
Actual output	Viewed order details
Test Result	Pass

Table 16:Test 10

The screenshot shows a web application interface. At the top, there is a dark blue header bar with navigation links: Home, About, Products (with a dropdown arrow), Seller Dashboard, Product Search, a search icon, and other user-related icons like notifications and cart. Below the header, the main content area has a light gray background. The title "Order History" is centered at the top of this area. A table follows, with columns labeled "Order Number", "Order Date", "Total Amount", "Status", and "Action". There is one row of data: Order Number 28, Order Date 04 Jun 2020, Total Amount 22.00, Status Complete (with a checkmark icon), and Action "View Order" next to it.

Figure 83:Order page

The screenshot shows a detailed view of an order. The top part of the page has a dark blue header with the same navigation links as Figure 83. Below the header, the title "Order Details" is centered. The page contains several sections of information:

- Order Summary:** Order: #28, Date: 04 Jun 2020, Order Total: 22.00, Order Status: ✓ Complete.
- Billing Address:** Siddhartha, Mid Baneshwor, Kathamndu, 44600, Nepal.
- Product Details:** A table showing a single item: Garnier Men acno fight. The table has columns for Product Description, View, Picture, and Sub-Total. The sub-total is 22.00.
- Total Summary:** Total 22.00, Total Paid 22.00.
- Shipping Address:** Siddhartha, Mid Baneshwor.
- Payment Details:** The order #28 has been paid successfully.
- Activation Message:** Activate Windows. Go to Settings to activate Windows.

Figure 84: Order detail page

TEST 11	
Action	Add shop product
Expected output	Added shop product
Actual output	Added shop product
Test Result	Pass

Table 17: Test 11

Product name: Lenovo phone
 Category: Mobile & Accessories
 Price: 2300
 Status: Shop Product
 Image: Choose File image_159058_4_lenovo.PNG
 Image: Choose File No file chosen
 Image: Choose File No file chosen
 Description: Lenevo phone with Dobly Digital Audio and Bluetooth

Submit

Activate Windows
Go to Settings to activate Windows.

Figure 85:Adding shop products

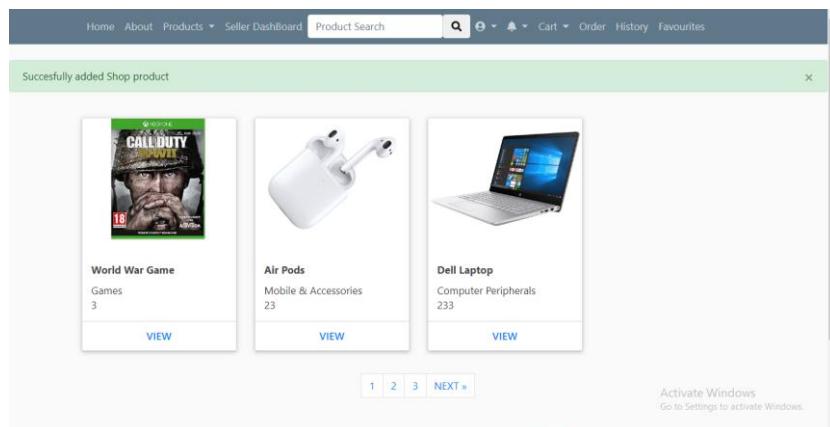


Figure 86: Successfully add shop product

TEST 12	
Action	Searching product
Expected output	Searching Product displays
Actual output	Products displays
Test Result	Pass

Table 18:Test 12

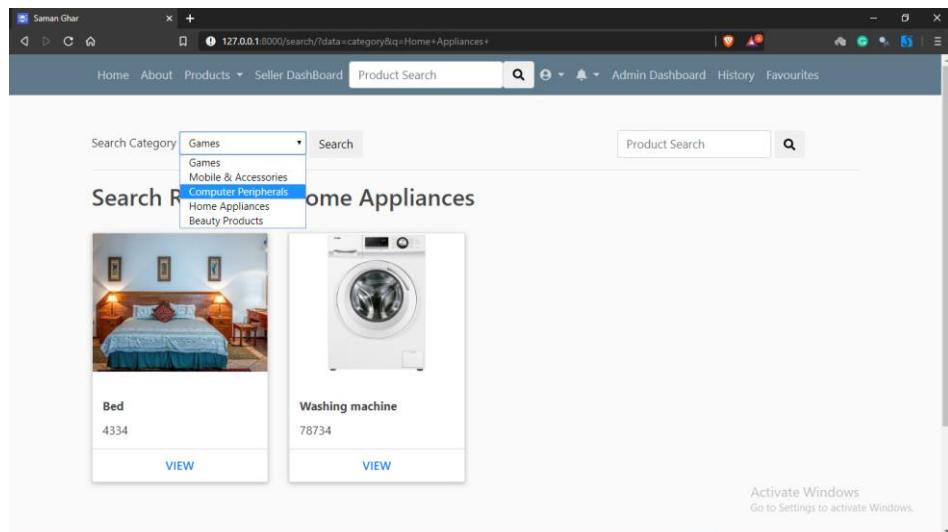


Figure 87:Search category products

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

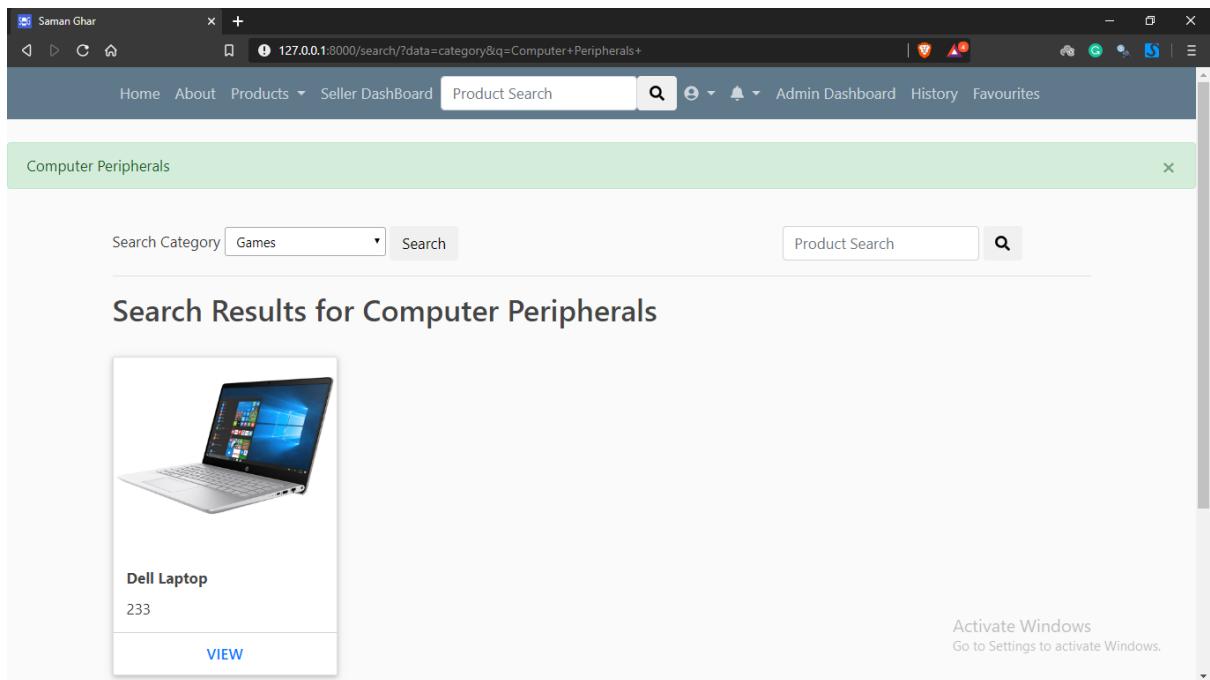


Figure 88:search

TEST 13	
Action	Read Notification
Expected output	Read notification.
Actual output	Notification successfully read.
Test Result	Pass

Table 19: TEST 13

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Home About Products Seller Dashboard Product Search Read New Product

11	You have a new bid amount for your product	Read	View Product
12	You have a new bid amount for your product	Read	View Product
13	You have a new bid amount for your product	Read	View Product
14	You have a new bid amount for your product	Read	View Product
15	You have a new bid amount for your product	Read	View Product
16	You have bidder	Mark as Read	View Product
17	You have a new bid amount for your products	Mark as Read	View Product
18	You have a new bid amount for your products	Mark as Read	View Product
19	You have a new bid amount for your products	Mark as Read	View Product
20	You have a new bid amount for your products	Mark as Read	View Product
21	You have new bidder	Mark as Read	View Product

Activate Windows
Go to Settings to activate Windows.

127.0.0.1:8000/viewed/22/

Figure 89: View notification

Home About Products Seller Dashboard Product Search Read Cart Order History Favourites

13	You have a new bid amount for your product	Read	View Product
14	You have a new bid amount for your product	Read	View Product
15	You have a new bid amount for your product	Read	View Product
16	You have bidder	Read	View Product
17	You have a new bid amount for your products	Mark as Read	View Product
18	You have a new bid amount for your products	Mark as Read	View Product
19	You have a new bid amount for your products	Mark as Read	View Product
20	You have a new bid amount for your products	Mark as Read	View Product
21	You have new bidder	Mark as Read	View Product

Figure 90: Viewed notification

TEST 14	
Action	File a dispute
Expected output	File a dispute
Actual output	Dispute filed
Test Result	Pass

Table 20: TEST 14

The screenshot shows a product listing for 'GARNIER MEN NEW AcnoFight PURPLE CLEANSING WHITENING CREAM'. The product image is a green and black box. Below the image, product details are listed:

- Product Name :** Acne Cream
- Category :** Beauty Products
- STATUS :** auction product
- Minimum Price :** 449
- Bid End Date :** June 5, 2020, 7:59 a.m.
- Winner :** dummy44444

A small note at the bottom left says 'Expired'.

To the right of the product listing is a sidebar with the following sections:

- Bidderlist**: A table showing the winning bidder:

UserName	Bid Amount	Winning
dummy44444	449	✓
- Conversations**: Buttons for 'Send Message' and 'Send Message' (black background).
- File a dispute**: Buttons for 'i did not get any product' and 'File dispute'.
- Activate Windows**: A note saying 'Go to Settings to activate Windows.'

Figure 91: dispute

The screenshot is similar to the previous one, showing the same product listing for 'GARNIER MEN NEW AcnoFight PURPLE CLEANSING WHITENING CREAM'.

A green header bar at the top indicates that a dispute has been filed: **dispute filed**.

The sidebar sections are identical to Figure 91, including the Bidderlist, Conversations, File a dispute, and Activate Windows sections.

TEST 15	
Action	Won Auction order
Expected output	Order auctions items
Actual output	Order successfully
Test Result	Pass

Table 21: Test 15

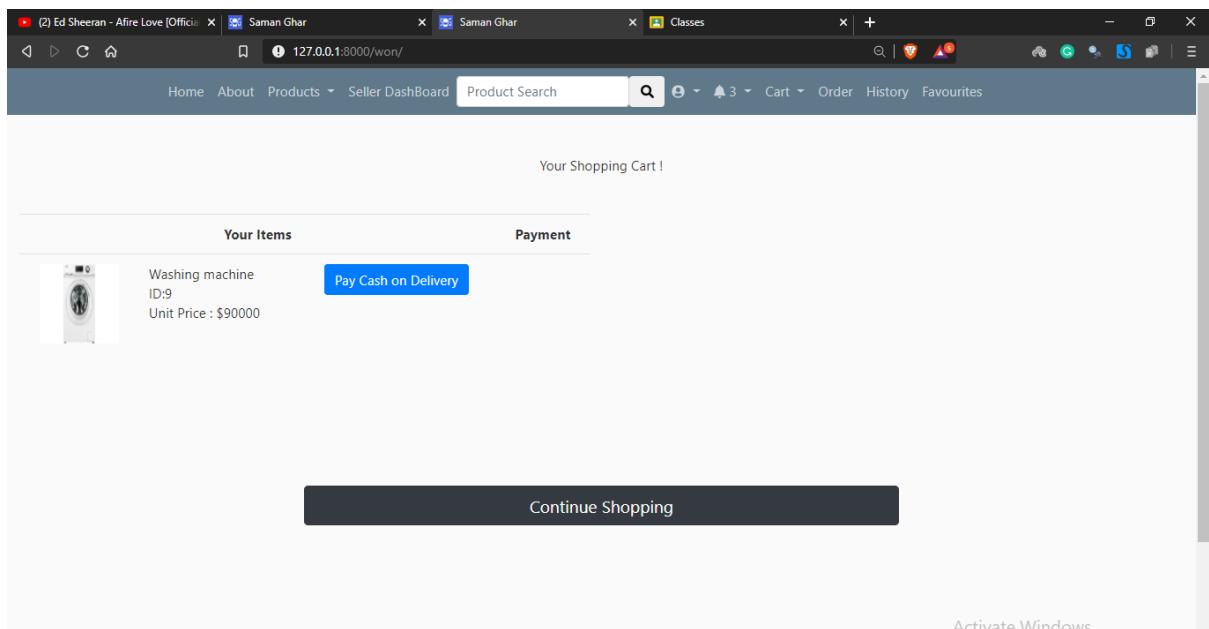


Figure 92: won auctions

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

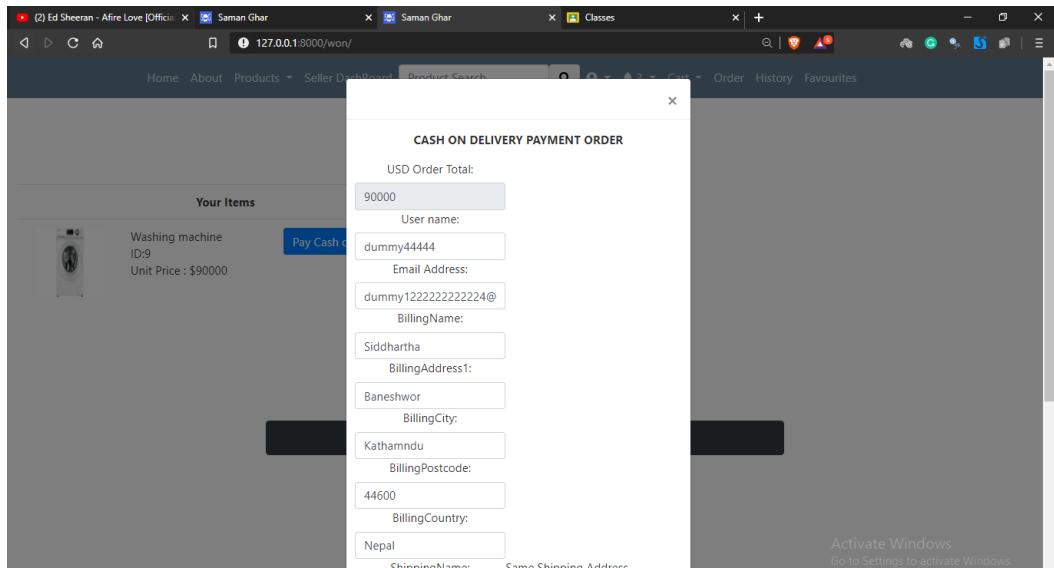


Figure 93: Filling form

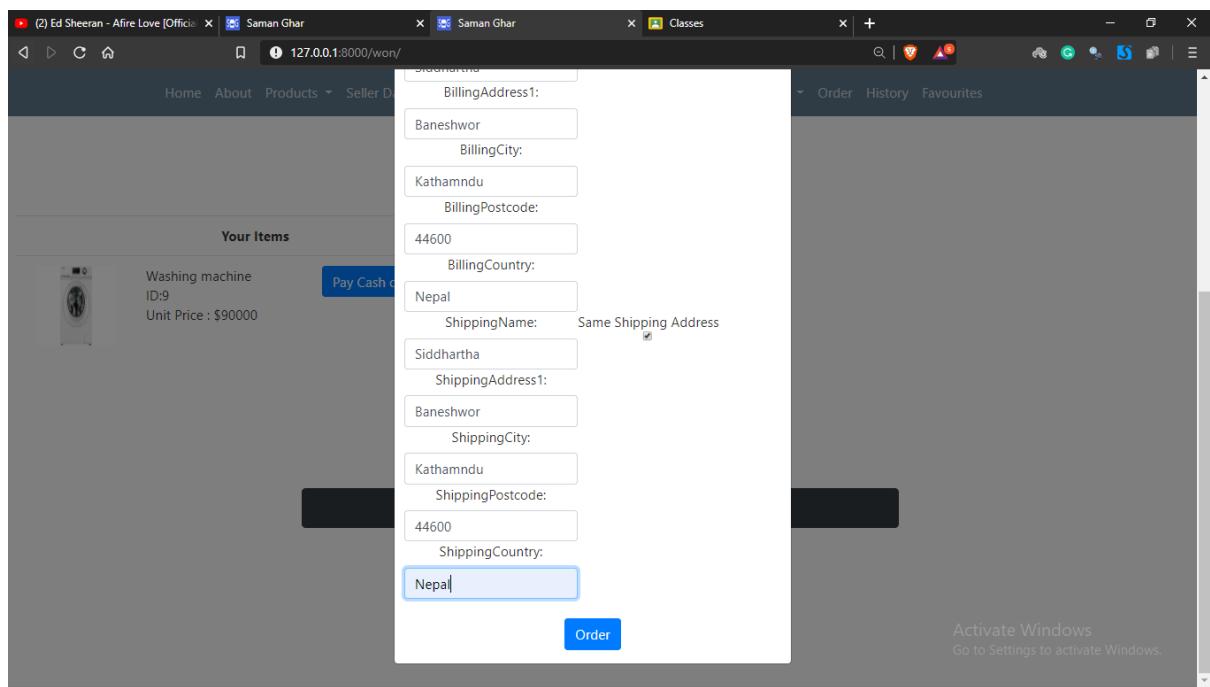


Figure 94: filling form

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

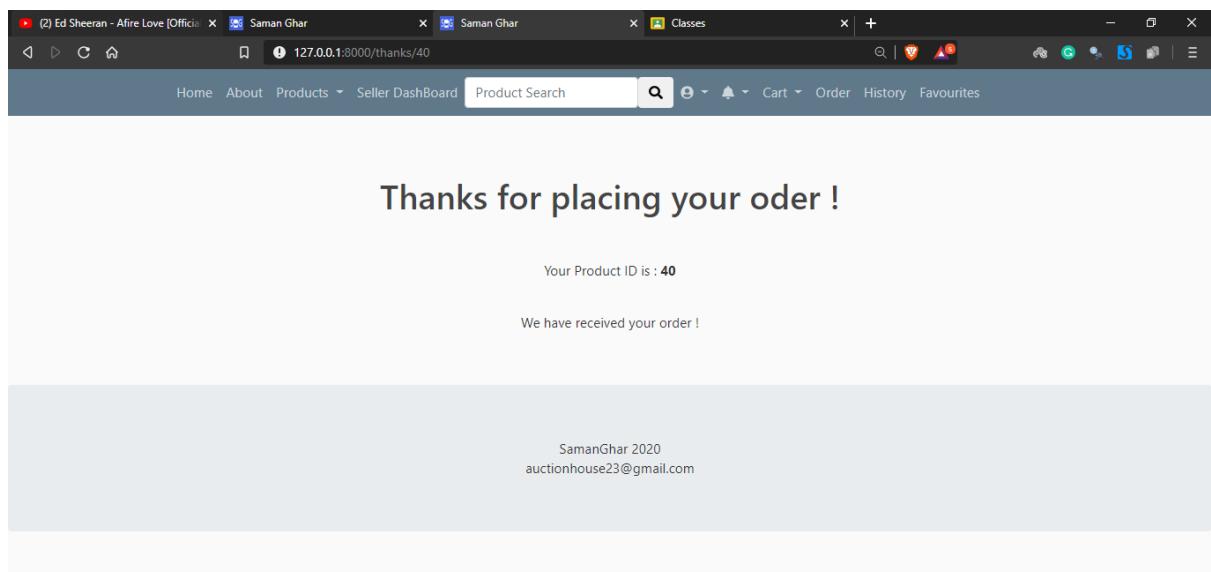


Figure 95:successfully order place

TEST 16	
Action	Adding conversation
Expected output	Conversation displayed
Actual output	Conversation is displayed
Test Result	Pass

Table 22: Test 16

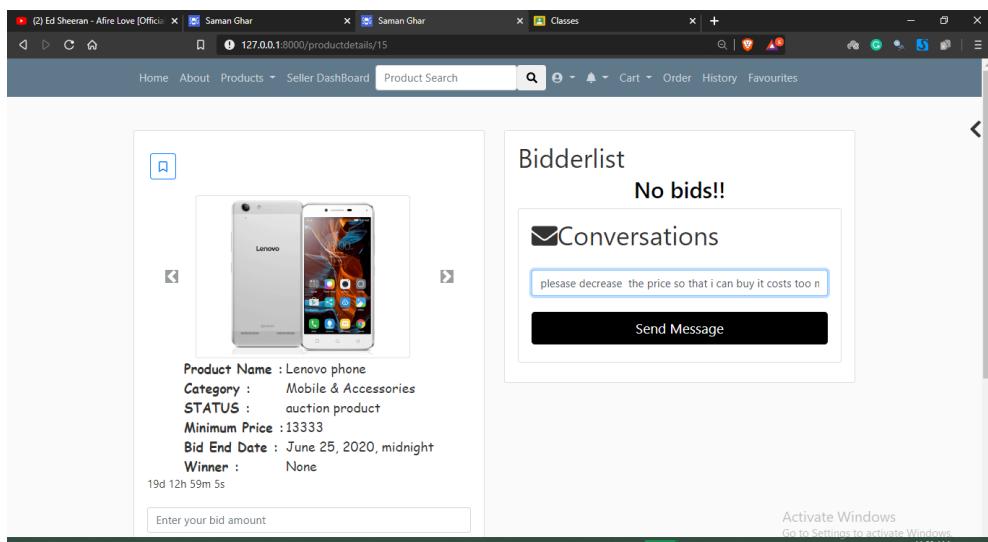


Figure 96: conversation

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

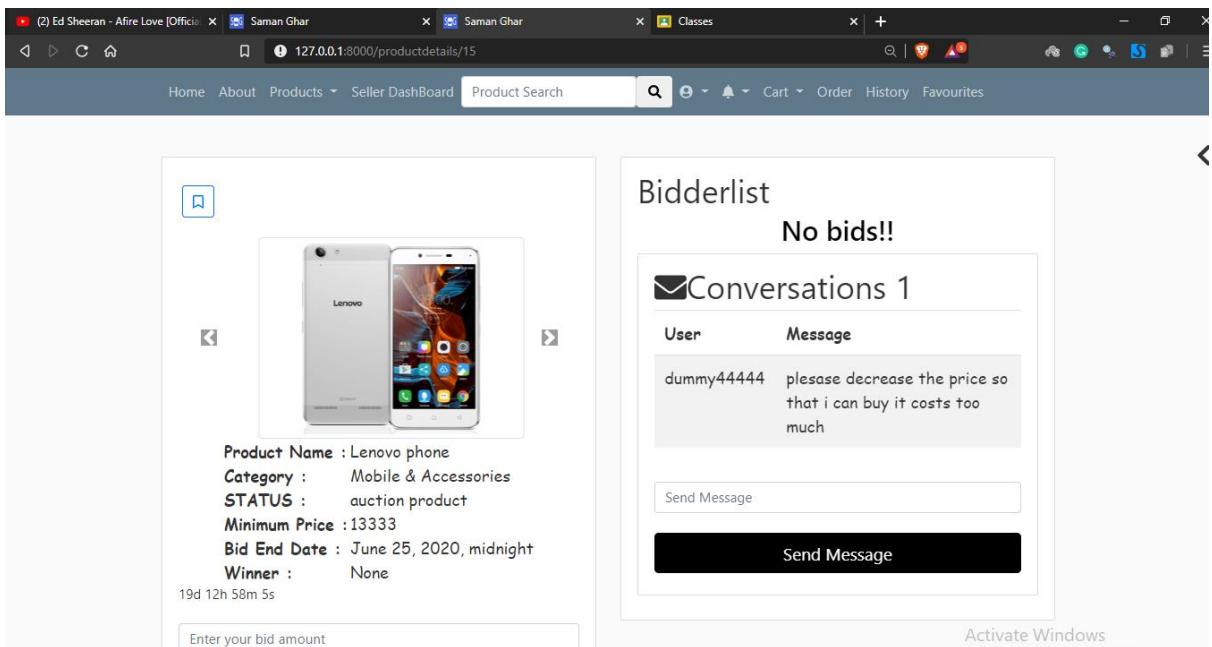


Figure 97: added conversation

TEST 17	
Action	Logging out
Expected output	Successfully log out
Actual output	Logged out successfully
Test Result	Pass

Table 23: Test 17

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

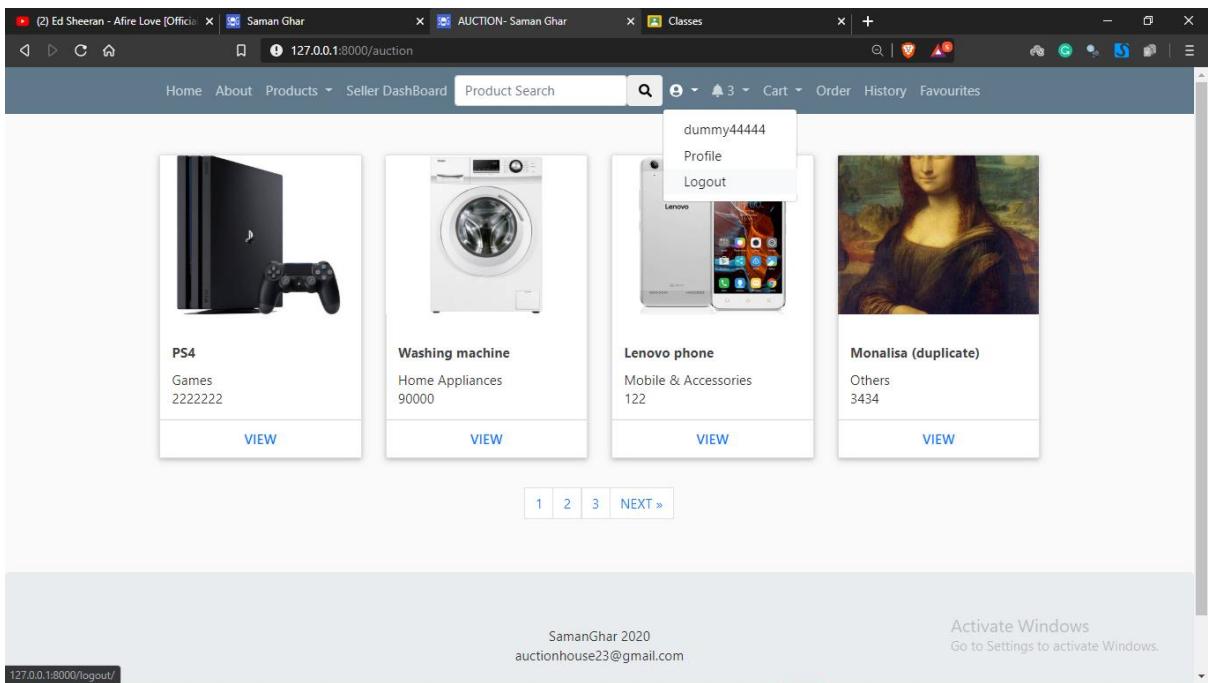


Figure 98: logout

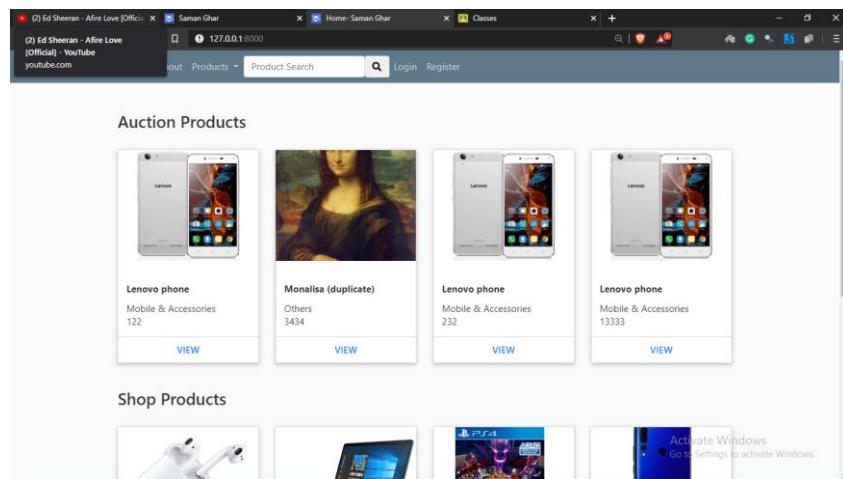


Figure 99: after logout

TEST 18	
Action	Changing Password
Expected output	Password change
Actual output	Changed password
Test Result	Pass

Table 24: Test18

The screenshot shows the user profile section on the left with the following details:

- Name: SiddharthaB
- Name: Sid Bhattacharai
- Email: siddharthaB@gmail.com
- Role: User

On the right, there is a form for updating the billing address:

Billing Name:	Siddhartha
Billing Address:	Mid Baneshwor
Billing City:	Kathmandu
Post Code:	44600
Update Billing Address	

Figure 100: Change Password

The screenshot shows the 'Reset Password' page with the following fields:

- Email*: siddharthaB@gmail.com
- [Request Password Reset](#)

Figure 101:Reset Pasword

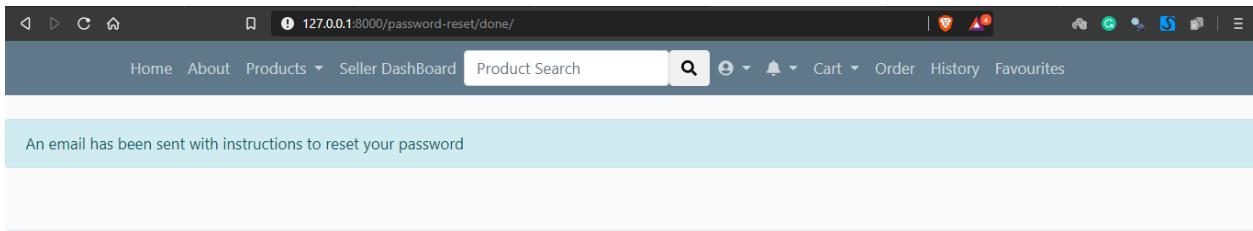


Figure 102: reset password sent

```
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Subject: Password reset on 127.0.0.1:8000
From: Ag <AuctionGhar@example.com>
To: siddharthaB@gmail.com
Date: Thu, 04 Jun 2020 12:48:42 -0000
Message-ID: <159127492210.19780.1273304163808747508@kushal>

You're receiving this email because you requested a password reset for your user account at 127.0.0.1:8000.

Please go to the following page and choose a new password:

http://127.0.0.1:8000/password-reset-confirm/MTY/5h2-0f3c6c27ca023ec59f5b/

Your username, in case you've forgotten: SiddharthaB

Thanks for using our site!

The 127.0.0.1:8000 team
```

Figure 103: link to reset password

The screenshot shows a dark blue header bar with navigation links: Home, About, Products (with a dropdown arrow), Seller Dashboard, Product Search, a magnifying glass icon, a user icon, a bell icon, Cart (with a dropdown arrow), Order, History, and Favourites. Below the header, the page title 'Reset Password' is displayed. A 'New password*' input field is present, containing several bullet points indicating password requirements: 'Your password can't be too similar to your other personal information.', 'Your password must contain at least 8 characters.', 'Your password can't be a commonly used password.', and 'Your password can't be entirely numeric.' Below the input field is a 'New password confirmation*' input field, which is currently empty. At the bottom is a teal-colored 'Reset Password' button.

Figure 104: form for reset password

This screenshot is identical to Figure 104, except the 'New password*' input field now contains five dots ('.....') instead of the password requirements. The rest of the interface, including the header, page title, and other fields, remains the same.

Figure 105: new password formed

TEST 19	
Action	Updating profile
Expected output	Change billing address
Actual output	Address changed
Test Result	Pass

Table 25: Test 19

The screenshot shows a user profile page for 'SiddharthaB'. On the left, there is a sidebar with 'Change Password' and 'Role User' options. The main area displays the user's name, email (siddharthaB@gmail.com), and role. To the right, there is a form for updating the billing address, which includes fields for Billing Name, Billing Address, Billing City, Post Code, and an 'Update Billing Address' button.

Figure 106:Profile update

The screenshot shows a dropdown menu for selecting an address type. The user has selected 'Siddhartha' from the list. The menu also includes 'Mid Baneshwor', 'Kathamndu', and '44600'. Below the dropdown, there is a field for 'Nepal' and a large 'Update' button. A watermark for 'Activate Windows' is visible on the right side of the screen.

Figure 107:Form for profile update

The screenshot shows the user's profile page after updating the address. The address has been successfully updated to 'Siddhartha', 'Mid Baneshwor', 'Kathamndu', '44600', and 'Nepal'. A green banner at the top indicates the update was successful.

Figure 108:Updated profile

TEST 20	
Action	Stripe payment
Expected output	Stripe payemnt
Actual output	Stripe payment successful.
Test Result	Pass

Table 26: Test 20

The screenshot shows a shopping cart page with a dark header bar containing navigation links like Home, About, Products, Seller Dashboard, Product Search, and a search icon. Below the header is a message 'Your Shopping Cart !'. The main content area has two columns: 'Your Items' and 'Payment'. In the 'Your Items' column, there are two product thumbnails, ID:23, and Unit Price : \$22. In the 'Payment' column, there are two buttons: 'Pay with Card' and 'Pay Cash on Delivery'. At the bottom is a large 'Continue Shopping' button.

Figure 109: Cart

This screenshot shows a modal window titled 'Auction-Store - New Order' overlaid on the shopping cart page. The modal contains fields for email (siddharthab@gmail.com), a checked checkbox for 'Same billing & shipping info', and a dropdown for shipping address selection. The dropdown shows 'Kushal' as the recipient, 'Mid-baneshwor' as the location, '44600' as the zip code, and 'Kathmandu' as the city. A 'Payment Info +' button is at the bottom of the modal. The background of the page shows the same shopping cart items and payment options as Figure 109.

Figure 110: Stripe payment

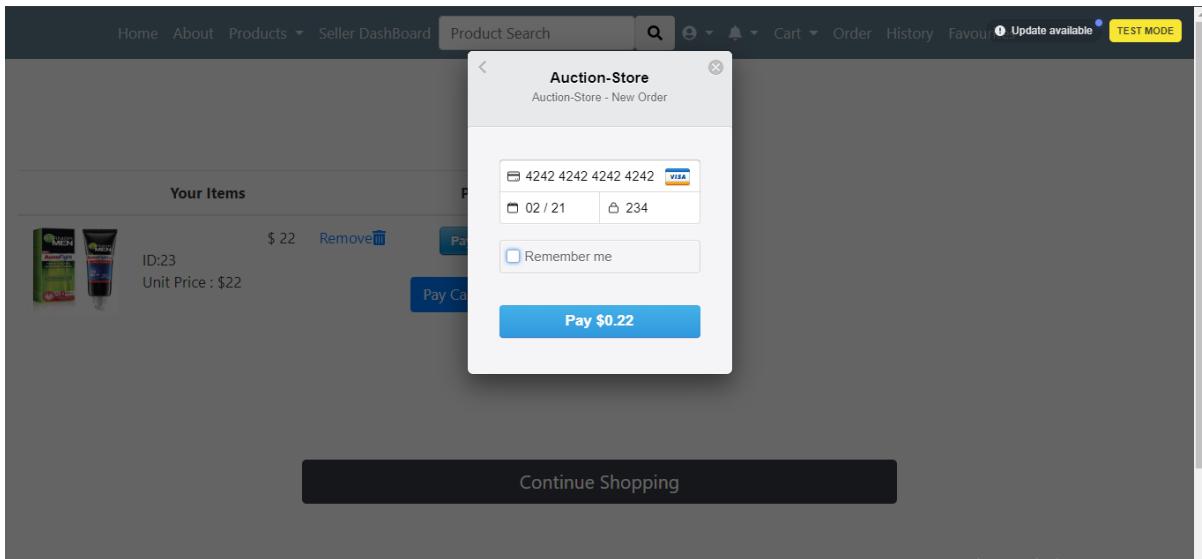


Figure 111:Stripe payment continued

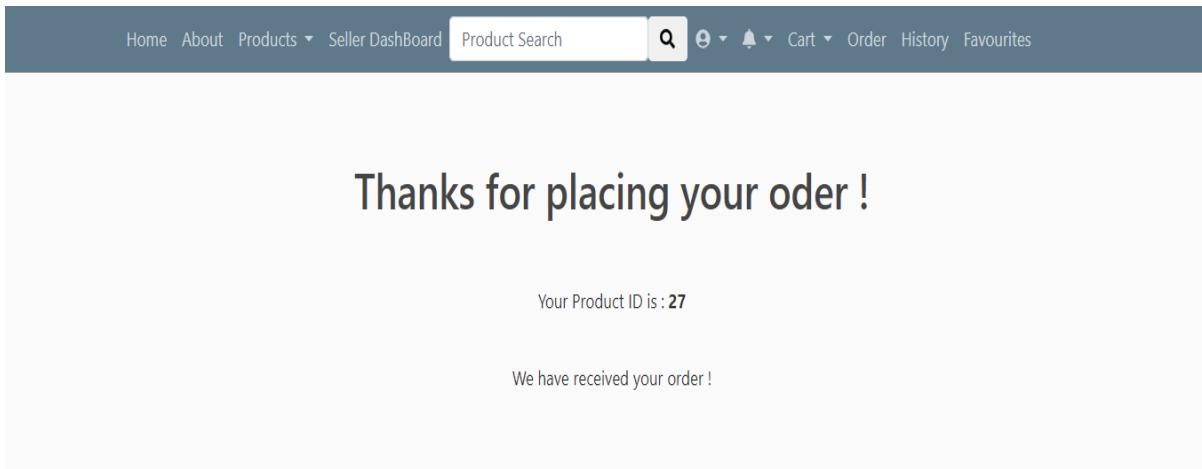


Figure 112:Order completed

4.4 CRITICAL ANALYSIS

The testing was completed with unit as well as system testing. The unit testing was done using coverage , and system testing was done with clear objectives that need to be pass or fail. While running the unit test in the system, every time a new test is created a new test database would be created which would be then deleted after the testing. The difficulty faced during the testing is the proper coding language the test is written in. while using only command prompt , the test were just completed and the data could not be displayed, but by the use of coverage the data test data can be viewed in the htmlcov folder of the system. The code that was used to display in test result in the file was coverage.html.

With all the difficulties of the testing and participation of test in the project , testing is a hard job to do but can also be viewed as a clear way of identifying the errors in the program and help the programmer fix the bug. The testing part gave me confidence to do more coding and improve my coding and be a better coder so that my code may contain less portion of bugs.

Finding about different types of testing gave me an experience as there are many testing phases , methods which helps in making the code better . after doing testing I clearly had the knowledge that testing is also an equal and important portion in system development.

CHAPTER 5: CONCLUSION

5.1 LEGAL, SOCIAL AND ETHICAL ISSUES

5.1.1 LEGAL ISSUES

Incorporation Problem: If you are a company operated merely via a website, not being incorporated is a crucial problem. Any purchase and selling activity related to your products will be considered illegal and you can't claim your right in case of any fraud and corruption. Without incorporation, your business has no shelter. Transaction Issues: There can be transaction issues where the money transfer cannot be done. SO cash on delivery is available . Products issues :The users can post illegal items which can be a problem. (EuroLogo Marketing , 2018)

5.1.2 SOCIAL ISSUES

The user can post images which are socially awkward. Drugs, other harmful products can be sold in the system which can cause social issue. The major social issues related to e-commerce and privacy concern the development of “exception of privacy” or privacy norms, as well as public attitudes. In what areas of should we as a society encourage people to think they are in “private territory” as opposed to public view? The major political issues related to ecommerce and privacy concern the development of statutes that govern the relations between record keepers and individuals. Shipping issues causing complexity in process with very high cost.

5.1.3 ETHICAL ISSUES

Data theft can be one of the ethical issues. Online sellers though have a responsibility to their customers. They must ensure that their eCommerce transactions do not result in data theft or security breaches. Customers give you a lot of sensitive information to complete an eCommerce transaction: sharing wrong pictures for the display and selling another things. inaccurate product data, Selling Counterfeit Products. (Lunka, 2015) Web tracking can also be an issue with Cyber Squatting.

5.2 ADVANTAGES

The advantages of the project are as follows:

1. The products can be sold fast and easily.
2. 2nd hand items can be sold easily
3. Anyone can sell Products , licences not needed.
4. Simple and understandable UI.
5. Easy to Bid
6. Easy to communicate to user
7. Easy to file dispute.

5.4 LIMITATIONS

The limitation of the project are as follows:

1. The background task was not implemented .
2. The search portion only searched if the name matches or the category matched other search like price search is not available.
3. The recommendation of the system can is only simply based on category.
4. The algorithms was not used in the system.
5. Since the system is new recommendation cannot be used properly.

5.3 FUTURE WORK

AI enables an ecommerce website to recommend products uniquely suited to shoppers and enables people to search for products using conversational language or images, as though they were interacting with a person. (Meier, 2020) A forum could be created to add new requests where the users can communicate. A real time communication can be created to help the system progress. A tracking system can be implemented in the system.

CHAPTER 6: REFERENCES

brandeis.edu. (2016) <https://www.cs.brandeis.edu/~magnus/ief248a/eBay/history.html> [Online]. Available from: <https://www.cs.brandeis.edu/~magnus/ief248a/eBay/history.html> [Accessed 2 January 2020].

Daraz. (2020) <https://www.daraz.com.np/> [Online]. Available from: <https://www.daraz.com.np/> [Accessed 22 March 2020].

Django Software Foundation. (2020) <https://docs.djangoproject.com/en/3.0/topics/db/models/> [Online]. Available from: <https://docs.djangoproject.com/en/3.0/topics/db/models/> [Accessed 22 January 2020].

Django Software Foundation. (2020) <https://docs.djangoproject.com/en/3.0/topics/http/views/> [Online]. Available from: <https://docs.djangoproject.com/en/3.0/topics/http/views/> [Accessed 2 January 2020].

Django Software Foundation. (2020) <https://docs.djangoproject.com/en/3.0/topics/templates/> [Online]. Available from: <https://docs.djangoproject.com/en/3.0/topics/templates/> [Accessed 22 January 2020].

eBay Inc. (2020) <https://www.ebayinc.com/company/who-we-are/> [Online]. Available from: <https://www.ebayinc.com/company/who-we-are/> [Accessed 1 January 2020].

EuroLogo Marketing. (2018) [common-legal-issues-faced-by-e-commerce-businesses/](#) [Online]. Available from: [EuroLogo Marketing](#) [Accessed 22 March 2020].

geeksforgeeks. (2019) [software-engineering-classical-waterfall-model](#) [Online]. Available from: <https://www.geeksforgeeks.org/software-engineering-classical-waterfall-model/> [Accessed 2 January 2020].

hamrobazaar. (2020) <https://hamrobazaar.com/> [Online]. Available from: <https://hamrobazaar.com/> [Accessed 3 March 2020].

Hsiao, A. (2018) [understanding-the-ebay-auction-automatic-bidding-system-1140186](#) [Online]. Available from: <https://www.thebalancesmb.com/understanding-the-ebay-auction-automatic-bidding-system-1140186> [Accessed 1 January 2020].

IBM Developer. (2020) <https://www.ibm.com/developerworks/rational/library/4763.html> [Online]. Available from: <https://www.ibm.com/developerworks/rational/library/4763.html> [Accessed 22 March 2020].

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Jacko, J.A. & Stephanidis, C. (2003) *Human-Computer Interaction: Theory and Practice Volume 1 of Human Factors and Ergonomics*. illustrated ed. CRC Press.

Learntek. (2019) *spiral-model* [Online]. Available from: <https://www.learntek.org/blog/spiral-model/> [Accessed 2 January 2020].

Lunka, R. (2015) *ethical-issues-in-ecommerce/* [Online]. Available from: <https://www.nchannel.com/blog/ethical-issues-in-ecommerce/> [Accessed 3 March 2020].

Meier, S. (2020) *the-future-of-ai-retail-and-roi* [Online]. Available from: <https://www.bigcommerce.com/blog/ecommerce-ai/#the-future-of-ai-retail-and-roi> [Accessed 3 March 2020].

My-Project-Management-Expert. (2009) *the-advantages-and-disadvantages-of-rup-software-development.html* [Online]. Available from: <http://www.my-project-management-expert.com/the-advantages-and-disadvantages-of-rup-software-development.html> [Accessed 4 January 2020].

ProductPlan. (2020) *moscow-prioritization/* [Online]. Available from: <https://www.productplan.com/glossary/moscow-prioritization/> [Accessed 2 march 2020].

Rational. (2011) *Rational Unified Process: Best Practices for Software development Teams*. Rational Software White Paper. IBM.

Sharpened Productions. (2019) *rup Definiton* [Online]. Available from: <https://techterms.com/definition/rup> [Accessed 3 September 2019].

Study.com. (2019) *what-is-the-rational-unified-process-methodology-tools-examples.html* [Online]. Available from: <https://study.com/academy/lesson/what-is-the-rational-unified-process-methodology-tools-examples.html> [Accessed 3 September 2019].

CHAPTER 7: BIBLIOGRAPHY

Aldaej, R. et al. (2018) Analyzing, Designing and Implementing a Web-Based Auction online System. *International Journal of Applied Engineering Research*, 12, pp.8005-13.

Arnaudova, M. (2010) *ebay-analysis* [Online]. Available at: <https://www.slideshare.net/MaryArnaudova/ebay-analysis> [Accessed 1 January 2019].

Blackhawk Network. (2020) *4-ideas-real-time-e-commerce* [Online]. Available from: <https://www.hawkcommerce.com/resource/4-ideas-real-time-e-commerce> [Accessed 6 March 2020].

E Radar. (2020) *fair-warning-7-risks-of-the-online-auction/* [Online]. Available from: <https://www.eradar.eu/fair-warning-7-risks-of-the-online-auction/> [Accessed 20 January 2020].

Marasini, M.K. (2019) <https://kathmandupost.com/columns/2019/07/15/digital-development-and-e-commerce-in-nepal> [Online]. Available from: <https://kathmandupost.com/columns/2019/07/15/digital-development-and-e-commerce-in-nepal> [Accessed 3 Feb 2020].

Medium. (2018) <https://medium.com/@boscacci/why-and-how-to-make-a-requirements-txt-f329c685181e> [Online]. Available from: <https://medium.com/@boscacci/why-and-how-to-make-a-requirements-txt-f329c685181e> [Accessed 22 April 2020].

Real Python. (2020) <https://realpython.com/testing-in-django-part-1-best-practices-and-examples/> [Online]. Available from: <https://realpython.com/testing-in-django-part-1-best-practices-and-examples/> [Accessed 2 Jan 2020].

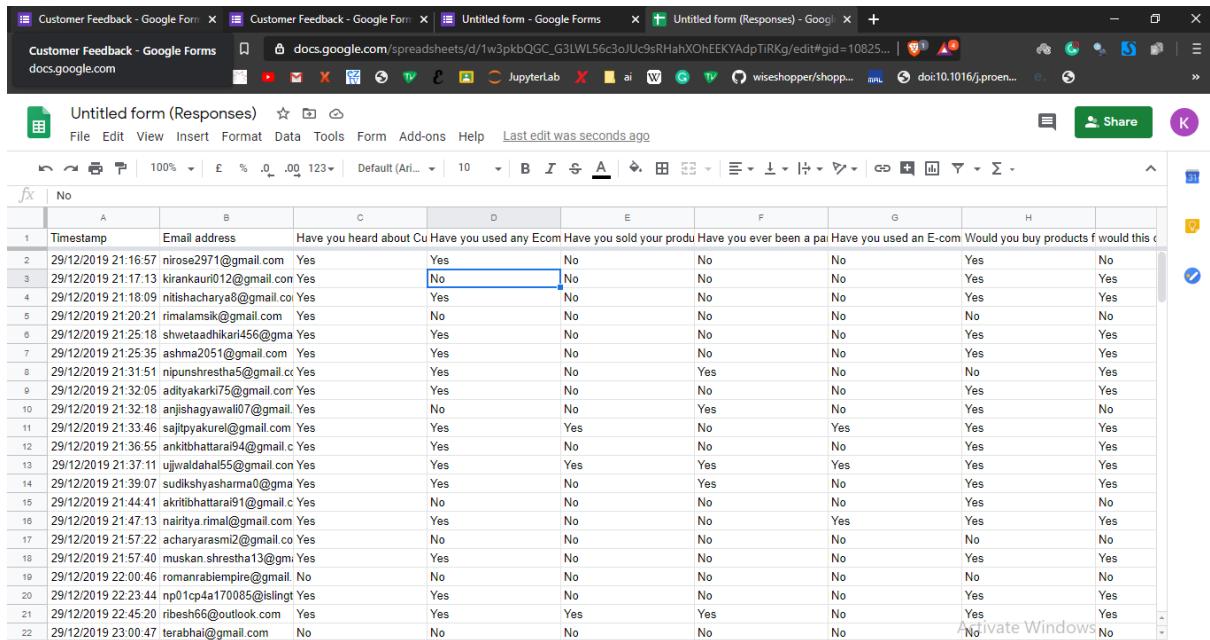
Schafer, J.B., Konstan, J.A. & Riedl, J. (2020) *schafer01ecommerce.pdf* [Online]. GroupLens Research Project Department of Computer Science and Engineering University of Minnesota University of Minnesota Available at: <http://www.cs.umd.edu/~samir/498/schafer01ecommerce.pdf> [Accessed 3 March 2020].

ShipBob, Inc. (2019) <https://www.shipbob.com/blog/ecommerce-order-tracking/> [Online]. Available from: <https://www.shipbob.com/blog/ecommerce-order-tracking/> [Accessed 2 June 2020].

CHAPTER 8: APPENDIX

8.1 APPENDIX A: PRE-SURVEY

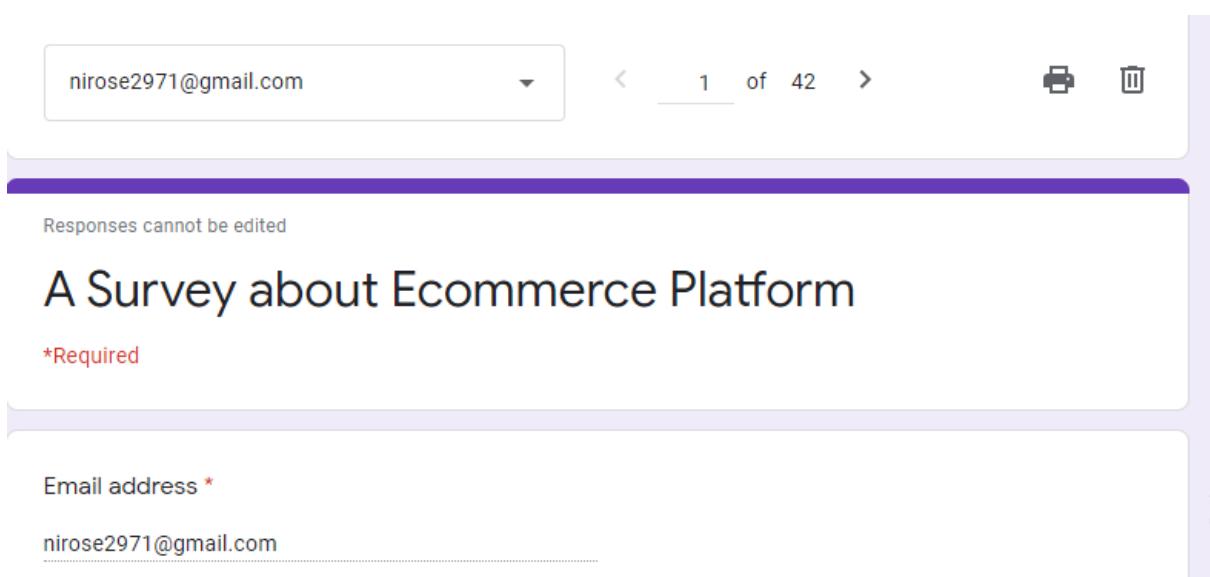
8.1.1 PRE-SURVEY FORM



	A	B	C	D	E	F	G	H
1	Timestamp	Email address	Have you heard about Cu	Have you used any Ecom	Have you sold your produ	Have you ever been a pa	Have you used an E-com	Would you buy products f
2	29/12/2019 21:16:57	nirose2971@gmail.com	Yes	Yes	No	No	Yes	No
3	29/12/2019 21:17:13	kirankauri12@gmail.com	Yes	No	No	No	Yes	Yes
4	29/12/2019 21:18:09	nitishcharyat@gmail.com	Yes	Yes	No	No	Yes	Yes
5	29/12/2019 21:20:21	rimalamisk@gmail.com	Yes	No	No	No	No	No
6	29/12/2019 21:25:18	shwetaadhikar456@gmail.com	Yes	Yes	No	No	Yes	Yes
7	29/12/2019 21:25:35	ashma2051@gmail.com	Yes	Yes	No	No	Yes	Yes
8	29/12/2019 21:31:51	nipunshrestha5@gmail.com	Yes	Yes	No	Yes	No	Yes
9	29/12/2019 21:32:05	adityakarik75@gmail.com	Yes	Yes	No	No	Yes	Yes
10	29/12/2019 21:32:18	anjishagayavali07@gmail.com	Yes	No	No	Yes	Yes	No
11	29/12/2019 21:33:46	saftyakure@gmail.com	Yes	Yes	No	Yes	Yes	Yes
12	29/12/2019 21:36:55	ankithbhatarai94@gmail.com	Yes	Yes	No	No	Yes	Yes
13	29/12/2019 21:37:11	ujjwaldahal55@gmail.com	Yes	Yes	Yes	Yes	Yes	Yes
14	29/12/2019 21:39:07	sudikshayasharma0@gmail.com	Yes	No	Yes	No	Yes	Yes
15	29/12/2019 21:44:41	akritibhatarai91@gmail.com	Yes	No	No	No	Yes	No
16	29/12/2019 21:47:13	nairitya.rimal@gmail.com	Yes	Yes	No	Yes	Yes	Yes
17	29/12/2019 21:57:22	acharyarasmit@gmail.com	Yes	No	No	No	No	No
18	29/12/2019 21:57:40	muskan.shrestha13@gmail.com	Yes	Yes	No	No	Yes	Yes
19	29/12/2019 22:00:46	romanrabiempire@gmail.com	No	No	No	No	No	No
20	29/12/2019 22:23:44	np01cp4a17008@islingt	Yes	Yes	No	No	Yes	Yes
21	29/12/2019 22:45:20	ribesh66@outlook.com	Yes	Yes	Yes	No	Yes	Yes
22	29/12/2019 23:00:47	terabhal@gmail.com	No	No	No	No	No	No

Figure 113:pre survey form

8.1.2 SAMPLE OF FILLED PRE-SURVEY FORMS



Responses cannot be edited

A Survey about Ecommerce Platform

*Required

Email address *

nirose2971@gmail.com

Figure 114:Survey -1

Have you heard about Customer to Customer E-commerce Platform? *

Yes
 No

Have you used any Ecommerce platform to buy products ? *

Yes
 No

Figure 115: Survey -2

Have you sold your products in E-commerce platform? *

Yes
 No

Have you ever been a part of auction of different products? *

Yes
 No

Figure 116: Survey -3

Have you used an E-commerce Platform which implements auction system to buy products? *

Yes
 No

Would you buy products from E-commerce platform? *

Yes
 No

Figure 117: Survey -4

would this online auction website be useful to you? *

Yes
 No

What would you prefer for payment options? *

Online Payment
 Cash On Delivery

Figure 118: Survey- 5

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Would you prefer to buy 2nd hand products from other user? *

Yes

No

Would you prefer to buy products from stores or through e commerce platforms? *

Stores

Ecommerce

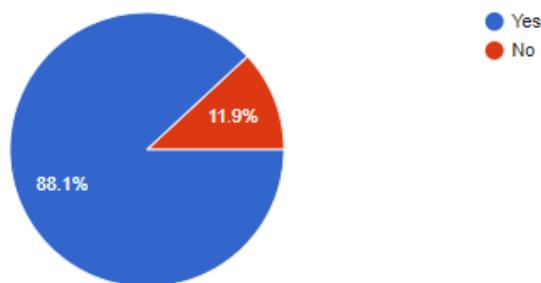
Submitted 29/12/2019, 21:16

Figure 119: Survey -6

8.1.3 PRE-SURVEY RESULT

Have you heard about Customer to Customer E-commerce Platform?

42 responses



Have you used any Ecommerce platform to buy products ?

42 responses

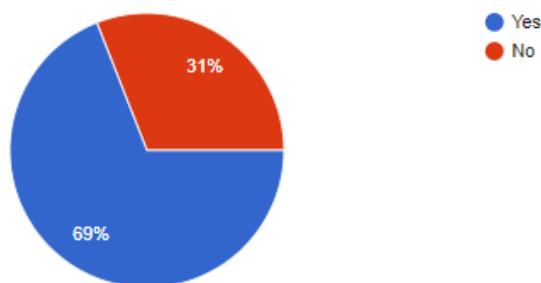
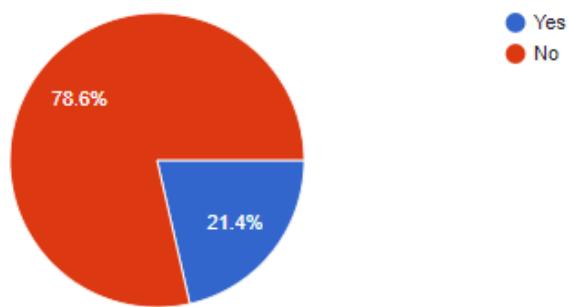


Figure 120:Result -I

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Have you sold your products in E-commerce platform?

42 responses



Have you ever been a part of auction of different products?

42 responses

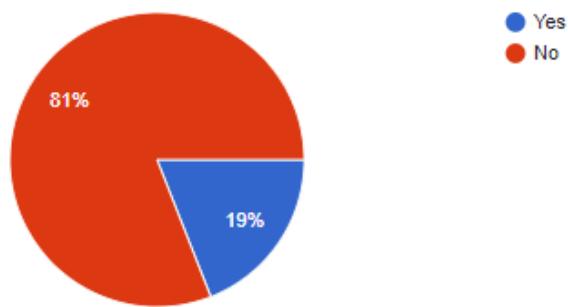
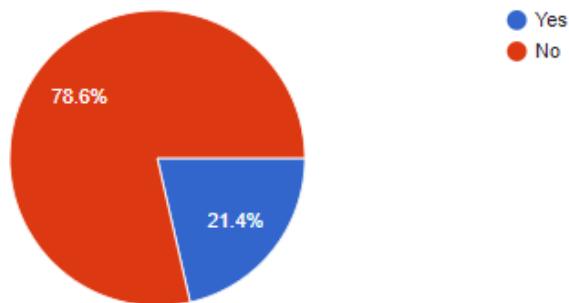


Figure 121: Result 2

Have you used an E-commerce Platform which implements auction system to buy products?



42 responses



Would you buy products from E-commerce platform?

42 responses

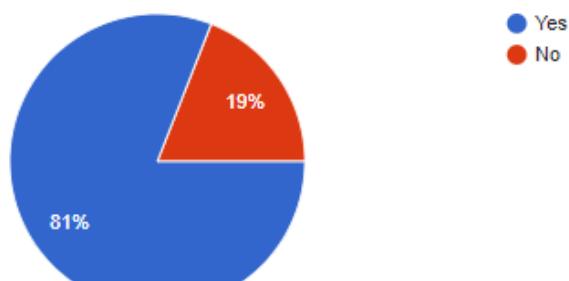
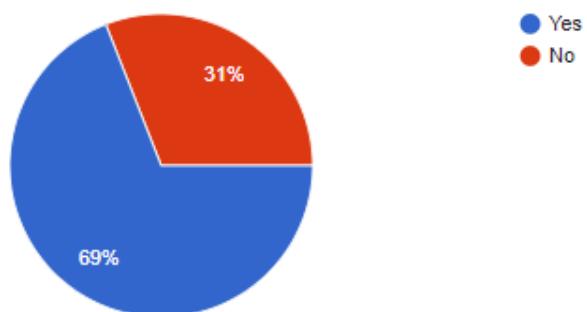


Figure 122: Result -3

would this online auction website be useful to you?

42 responses



What would you prefer for payment options?

42 responses

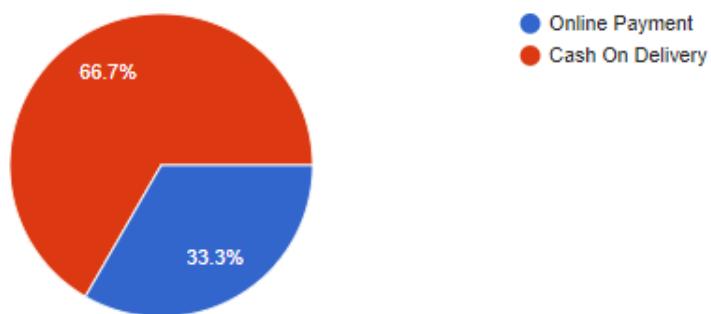
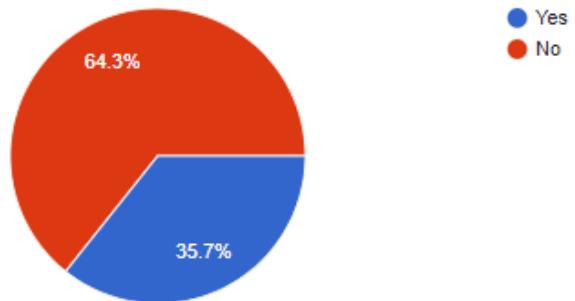


Figure 123: Result 4

Would you prefer to buy 2nd hand products from other user?

42 responses



Would you prefer to buy products from stores or through e commerce platforms?

42 responses

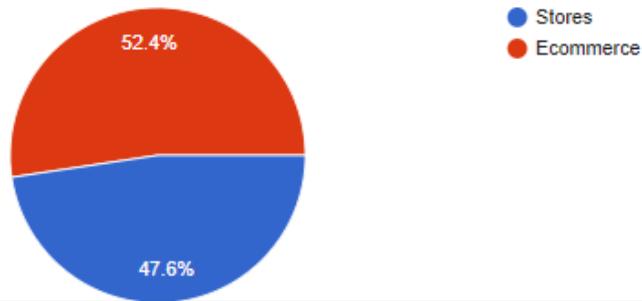


Figure 124: Result 5

8.2 APPENDIX B: POST-SURVEY

8.2.1 POST-SURVEY FORM

A Survey about Ecommerce Platform

Form description

Email address *

Valid email address

This form is collecting email addresses. [Change settings](#)

Would you buy online 2nd hand items? *

Yes

No

Did you like the concept of silent bidding? *

Yes

No

Figure 125: Form1

Do you want online payment in the system? *

Yes

No

do you want cash on delivery system ? *

Yes

No

Figure 126:Form2

would you want to converse with owner during your buy? *

Yes

No

Would you like recommendation system of products?

Yes

No

Figure 127:Form 3

8.2.2 SAMPLE OF FILLED POST-SURVEY FORMS

A Survey about Ecommerce Platform

*Required

Email address *

joshibinayak17@gmail.com

Would you buy online 2nd hand items? *

Yes

No

Did you like the concept of silent bidding? *

Yes

No

Figure 128: filled form1

Do you want online payment in the system? *

Yes
 No

do you want cash on delivery system ? *

Yes
 No

would you want to converse with owner during your buy? *

Yes
 No

Figure 129: filled form-2

Would you like recommendation system of products?

Yes
 No

Figure 130: Filled form-3

8.2.3 POST-SURVEY RESULT

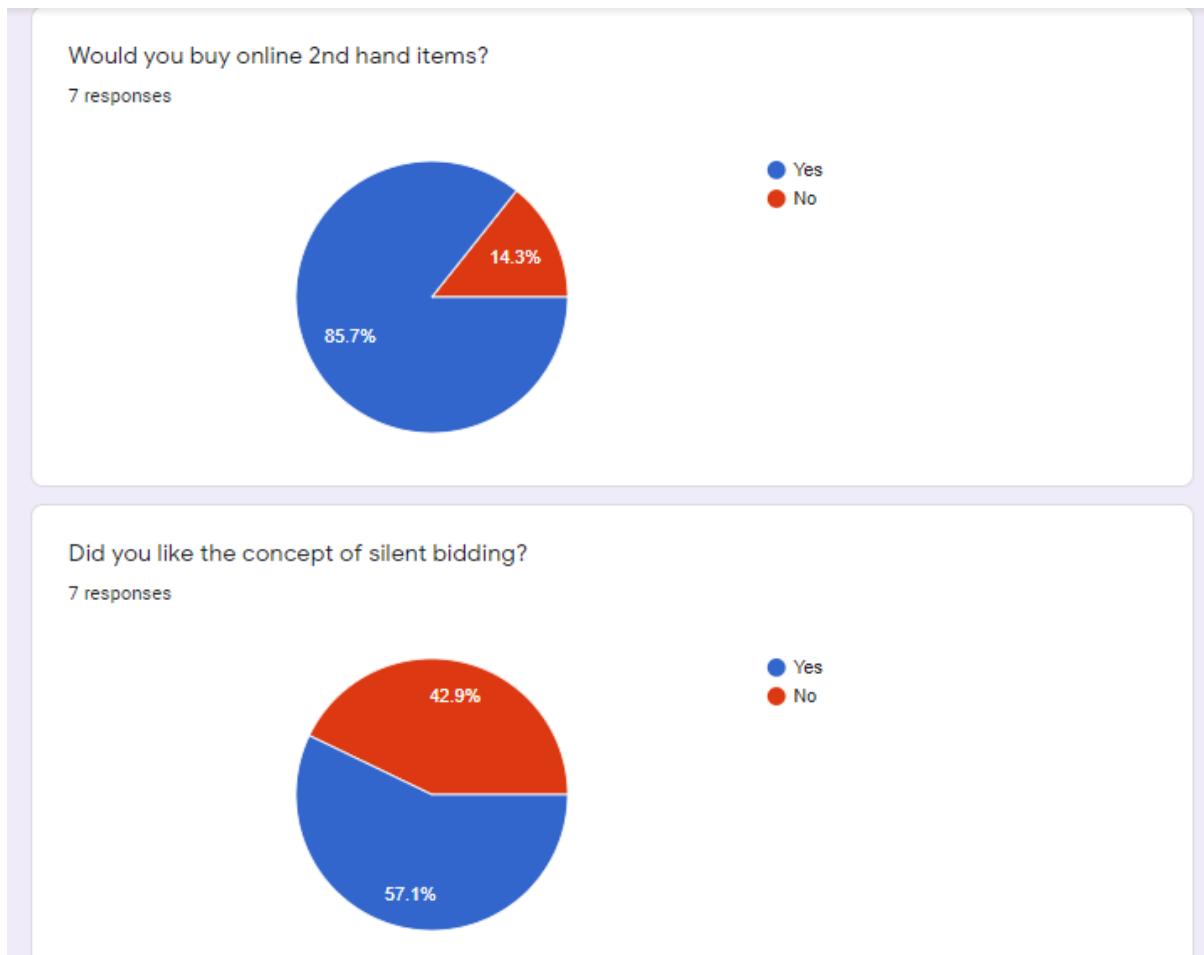
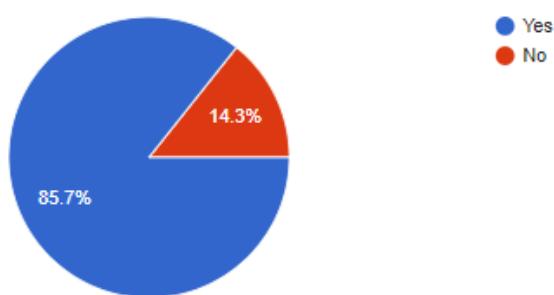


Figure 131:Result-1

Do you want online payment in the system?

7 responses



do you want cash on delivery system ?

7 responses

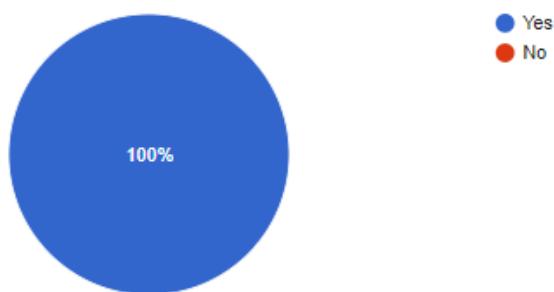
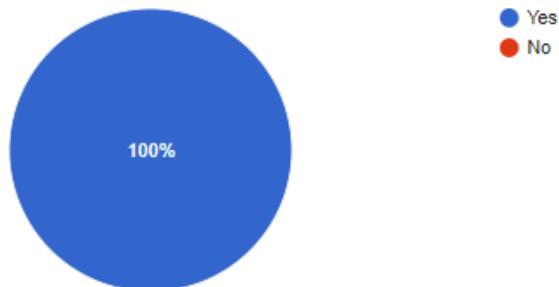


Figure 132:Result 2

would you want to converse with owner during your buy?

7 responses



Would you like recommendation system of products?

7 responses

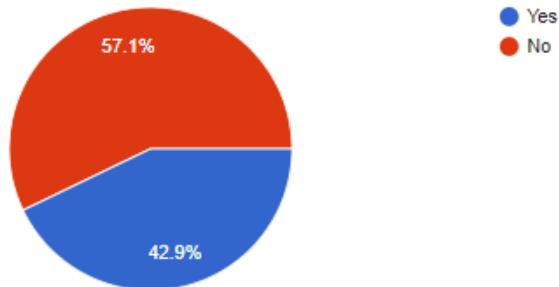
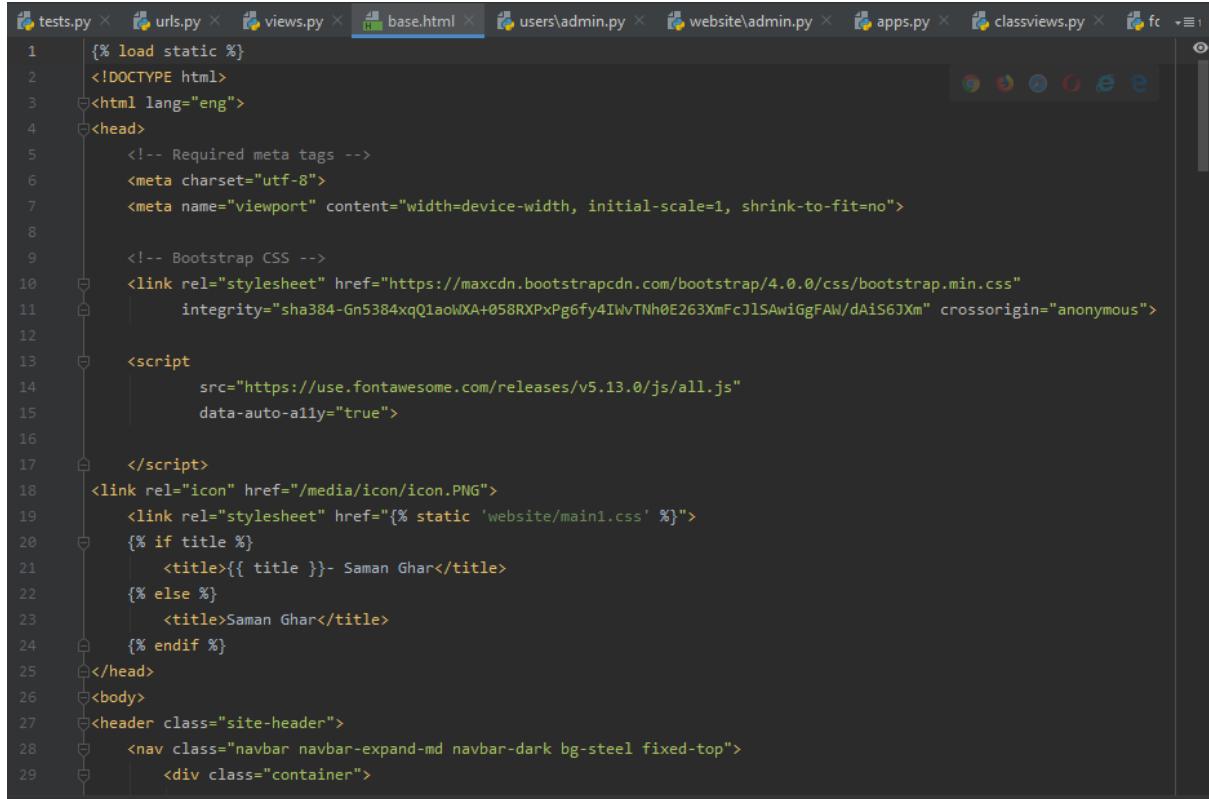


Figure 133:Result 3

8.3 APPENDIX C: SAMPLE CODES

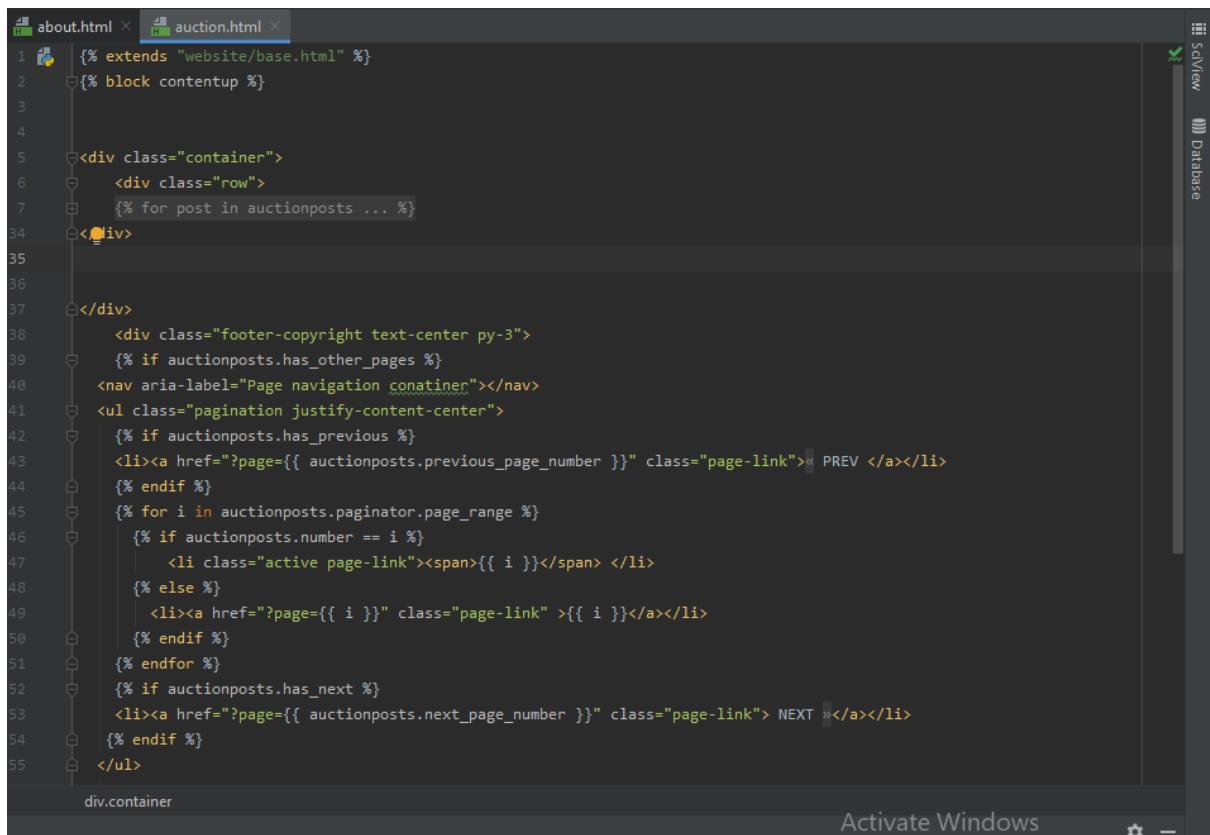
8.3.1 SAMPLE CODE OF THE UI



```
1  {% load static %}          tests.py × urls.py × views.py × base.html × users\admin.py × website\admin.py × apps.py × classviews.py × fc ...
2  <!DOCTYPE html>
3  <html lang="eng">
4  <head>
5      <!-- Required meta tags -->
6      <meta charset="utf-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
8
9      <!-- Bootstrap CSS -->
10     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
11         integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGFAW/dAiS6JXm" crossorigin="anonymous">
12
13     <script
14         src="https://use.fontawesome.com/releases/v5.13.0/js/all.js"
15         data-auto-ally="true">
16
17     </script>
18     <link rel="icon" href="/media/icon/icon.PNG">
19     <link rel="stylesheet" href="{% static 'website/main1.css' %}">
20     {% if title %}
21         <title>{{ title }}- Saman Ghar</title>
22     {% else %}
23         <title>Saman Ghar</title>
24     {% endif %}
25     </head>
26     <body>
27         <header class="site-header">
28             <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
29                 <div class="container">
```

Figure 134:base code

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



The screenshot shows a code editor with two tabs: 'about.html' and 'auction.html'. The 'auction.html' tab is active, displaying the following template code:

```
1  {% extends "website/base.html" %}  
2  {% block contentup %}  
3  
4  
5      <div class="container">  
6          <div class="row">  
7              {% for post in auctionposts ... %}  
8                  <div>  
9  
10                 </div>  
11             </div>  
12             <div class="footer-copyright text-center py-3">  
13                 {% if auctionposts.has_other_pages %}  
14                     <nav aria-label="Page navigation conatiner"></nav>  
15                     <ul class="pagination justify-content-center">  
16                         {% if auctionposts.has_previous %}  
17                             <li><a href="?page={{ auctionposts.previous_page_number }}" class="page-link"> PREV </a></li>  
18                         {% endif %}  
19                         {% for i in auctionposts.paginator.page_range %}  
20                             {% if auctionposts.number == i %}  
21                                 <li class="active page-link"><span>{{ i }}</span> </li>  
22                             {% else %}  
23                                 <li><a href="?page={{ i }}" class="page-link" >{{ i }}</a></li>  
24                             {% endif %}  
25                         {% endfor %}  
26                         {% if auctionposts.has_next %}  
27                             <li><a href="?page={{ auctionposts.next_page_number }}" class="page-link"> NEXT </a></li>  
28                         {% endif %}  
29                     </ul>  
30             </div>  
31         </div>  
32     </div>
```

The code includes logic for rendering a list of posts, a footer copyright notice, and a pagination navigation bar. The pagination bar shows previous and next page links, along with numbered links for each page in the range.

Figure 135:auction

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

```

1  {% extends 'website/base.html' %}

2  {% block contentup %}
3      {% if not cart_items %}
4          <div class="text-center">
5              <br>
6              <h1 class="text-center product_title">Your Shopping Cart is Empty</h1>
7              <br>
8              <p class="text-center">Please Click <a href="{% url 'home' %}>here</a> to continue shopping ! </p>
9          </div>
10     {% else %}
11         <div class="text-center">
12             <br>
13             <div class="text-center product_title">
14                 Your Shopping Cart !
15             </div>
16         </div>
17     <br>
18     <div class="row mx-auto">
19         <div class="col-12 col-sm-12 col-md-12 col-lg-6 text-center">
20             <table class="table cart_table" >
21                 <thead class="cart_head">
22                     <tr>
23                         <th colspan="4">
24                             Your Items
25                         </th>
26                         <th colspan="4">
27                             Payment
28                         </th>
29                     </tr>
30                 </thead>
31                 <tbody>
32                     <tr>
33                         <td>
34                             <div class="form-row" style="border-bottom: 1px solid #ccc; padding-bottom: 10px;">
35                                 <div class="form-group col-md-6">
36                                     <label for="inputname">Name</label>
37                                     <input type="text" class="form-control" id="name" name="name" placeholder="Name">
38                                 </div>
39                                 <div class="form-group col-md-6">
40                                     <label for="inputEmail4">Email</label>
41                                     <input type="email" class="form-control" id="email" name="email" placeholder="Email">
42                                 </div>
43                             </div>
44                             <div class="form-group">
45                                 <label for="inputAddress">Address</label>
46                                 <input type="text" class="form-control" id="address1" name="address1" placeholder="1234 Main St">
47                             </div>
48                             <div class="form-group">
49                                 <label for="inputAddress2">Address line 2</label>
50                                 <input type="text" class="form-control" id="address2" name="address2" placeholder="Apartment, studio, or floor">
51                             </div>
52                             <div class="form-row">
53                                 <div class="form-group col-md-6">
54                                     <label for="inputCity">City</label>
55                                     <input type="text" class="form-control" id="city" name="city">
56                                 </div>
57                                 <div class="form-group col-md-4">
58                                     <label for="inputZip">Zip Code</label>
59                                     <input type="text" class="form-control" id="zip" name="zip">
60                                 </div>
61                             </div>
62                         </td>
63                         <td>
64                             <div class="form-row" style="border-bottom: 1px solid #ccc; padding-bottom: 10px;">
65                                 <div class="form-group col-md-6">
66                                     <label for="inputName2">Name</label>
67                                     <input type="text" class="form-control" id="name2" name="name2" placeholder="Name">
68                                 </div>
69                                 <div class="form-group col-md-6">
70                                     <label for="inputEmail5">Email</label>
71                                     <input type="email" class="form-control" id="email2" name="email2" placeholder="Email">
72                                 </div>
73                             </div>
74                             <div class="form-group">
75                                 <label for="inputAddress3">Address</label>
76                                 <input type="text" class="form-control" id="address3" name="address3" placeholder="1234 Main St">
77                             </div>
78                             <div class="form-group">
79                                 <label for="inputAddress4">Address line 2</label>
80                                 <input type="text" class="form-control" id="address4" name="address4" placeholder="Apartment, studio, or floor">
81                             </div>
82                             <div class="form-row">
83                                 <div class="form-group col-md-6">
84                                     <label for="inputCity2">City</label>
85                                     <input type="text" class="form-control" id="city2" name="city2" placeholder="City">
86                                 </div>
87                                 <div class="form-group col-md-4">
88                                     <label for="inputZip2">Zip Code</label>
89                                     <input type="text" class="form-control" id="zip2" name="zip2" placeholder="Zip Code">
90                                 </div>
91                             </div>
92                         </td>
93                     </tr>
94                 </tbody>
95             </table>
96         </div>
97     </div>
98 
```

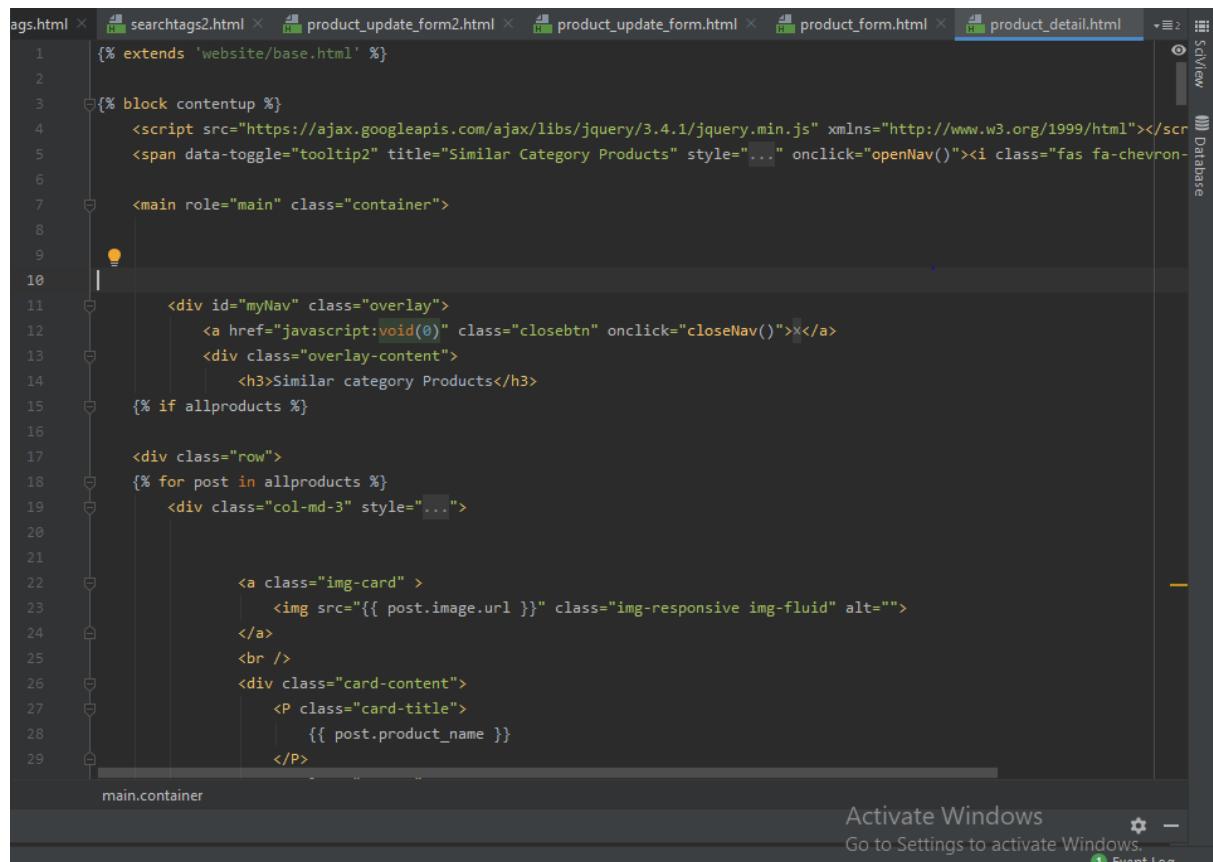
Figure 136:Cart

```

1  <div class="col my-6">
2      <h2>Step 2 - Enter Address & Other Details:</h2>
3      <form method="post" action="/shop/checkout/">{% csrf_token %}</form>
4          <input type="hidden" name="itemsJson" id="itemsJson">
5          <div class="form-row">
6              <div class="form-group col-md-6">
7                  <label for="inputname">Name</label>
8                  <input type="text" class="form-control" id="name" name="name" placeholder="Name">
9              </div>
10             <div class="form-group col-md-6">
11                 <label for="inputEmail4">Email</label>
12                 <input type="email" class="form-control" id="email" name="email" placeholder="Email">
13             </div>
14         </div>
15         <div class="form-group">
16             <label for="inputAddress">Address</label>
17             <input type="text" class="form-control" id="address1" name="address1" placeholder="1234 Main St">
18         </div>
19         <div class="form-group">
20             <label for="inputAddress2">Address line 2</label>
21             <input type="text" class="form-control" id="address2" name="address2" placeholder="Apartment, studio, or floor">
22         </div>
23         <div class="form-row">
24             <div class="form-group col-md-6">
25                 <label for="inputCity">City</label>
26                 <input type="text" class="form-control" id="city" name="city" placeholder="City">
27             </div>
28             <div class="form-group col-md-4">
29                 <label for="inputZip">Zip Code</label>
30                 <input type="text" class="form-control" id="zip" name="zip" placeholder="Zip Code">
31             </div>
32         </div>
33     </div>
34 
```

Figure 137:checkout

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



The screenshot shows a code editor with multiple tabs open at the top, including 'ags.html', 'searchtags2.html', 'product_update_form2.html', 'product_update_form.html', 'product_form.html', and 'product_detail.html'. The 'product_detail.html' tab is active. The code in the editor is a Django template for a product detail page. It starts with an extends tag to inherit from 'website/base.html'. It then defines a content block named 'contentup'. Inside this block, there is a script tag for jQuery and a tooltip for a category products link. The main content area starts with a main container. Inside the main container, there is a div with id 'myNav' which contains a close button and an overlay content section with a heading 'Similar category Products'. A conditional block checks if 'allproducts' is available. If true, it enters a for loop where each product is wrapped in a col-md-3 grid item. Each item contains an img card with a responsive image and a title card containing the product name.

```
{% extends 'website/base.html' %}

{% block contentup %}
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js" xmlns="http://www.w3.org/1999/xhtml"></script>
    <span data-toggle="tooltip2" title="Similar Category Products" style="..." onclick="openNav()"><i class="fas fa-chevron-right" style="font-size: 24px; color: #ccc; cursor: pointer;"></i></span>
<main role="main" class="container">

    <div id="myNav" class="overlay">
        <a href="javascript:void(0)" class="closebtn" onclick="closeNav()">×</a>
        <div class="overlay-content">
            <h3>Similar category Products</h3>
    {% if allproducts %}

        <div class="row">
        {% for post in allproducts %}
            <div class="col-md-3" style="...">

                <a class="img-card" >
                    
                </a>
                <br />
                <div class="card-content">
                    <P class="card-title">
                        {{ post.product_name }}
                    </P>
                </div>
            </div>
        {% endfor %}
    </div>
    {% endif %}

</div>
</main>

```

Figure 138:Product detail

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Figure 139:My Products

```

1  {% extends "website/base.html" %}           main.container > div > div.row > div.col-md-12
2  {% block contentup %}
3
4
5      <main role="main" class="container">
6          <div>
7              <div class="row">
8
9                  <div class="col-md-12">
10
11
12                      <h3 class="h3">Auction Products</h3>
13
14                      <div class="row">
15                          {% if auctionposts %}
16                          {% for post in auctionposts %}
17                              <div class="col-md-3">
18
19                                  <div class="card">
20                                      <a class="img-card" >
21                                          
22                                      </a>
23                                      <br />
24                                      <div class="card-content">
25                                          <p class="card-title">
26                                              {{ post.product_name }}
27                                          </p>
28                                          <p class="center">
29                                              {{ post.category }}

```

Figure 140:Home

8.3.2 SAMPLE CODE FOR THE SYSTEM

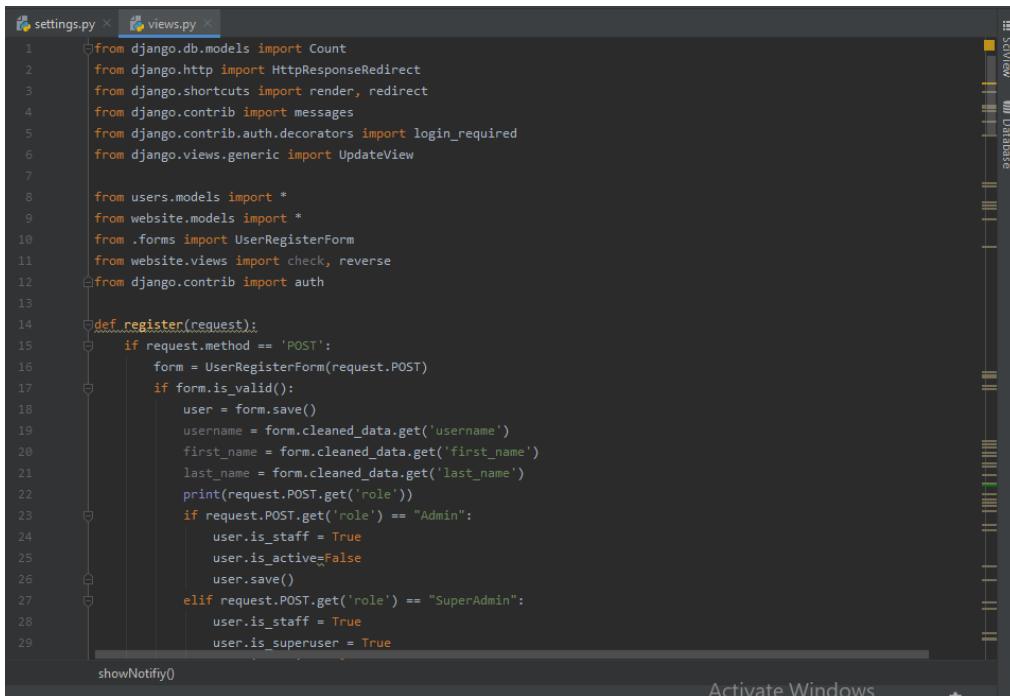
```

1  """
2      Django settings for django_project project.
3
4      Generated by 'django-admin startproject' using Django 2.1.
5
6      For more information on this file, see
7      https://docs.djangoproject.com/en/2.1/topics/settings/
8
9      For the full list of settings and their values, see
10     https://docs.djangoproject.com/en/2.1/ref/settings/
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29

```

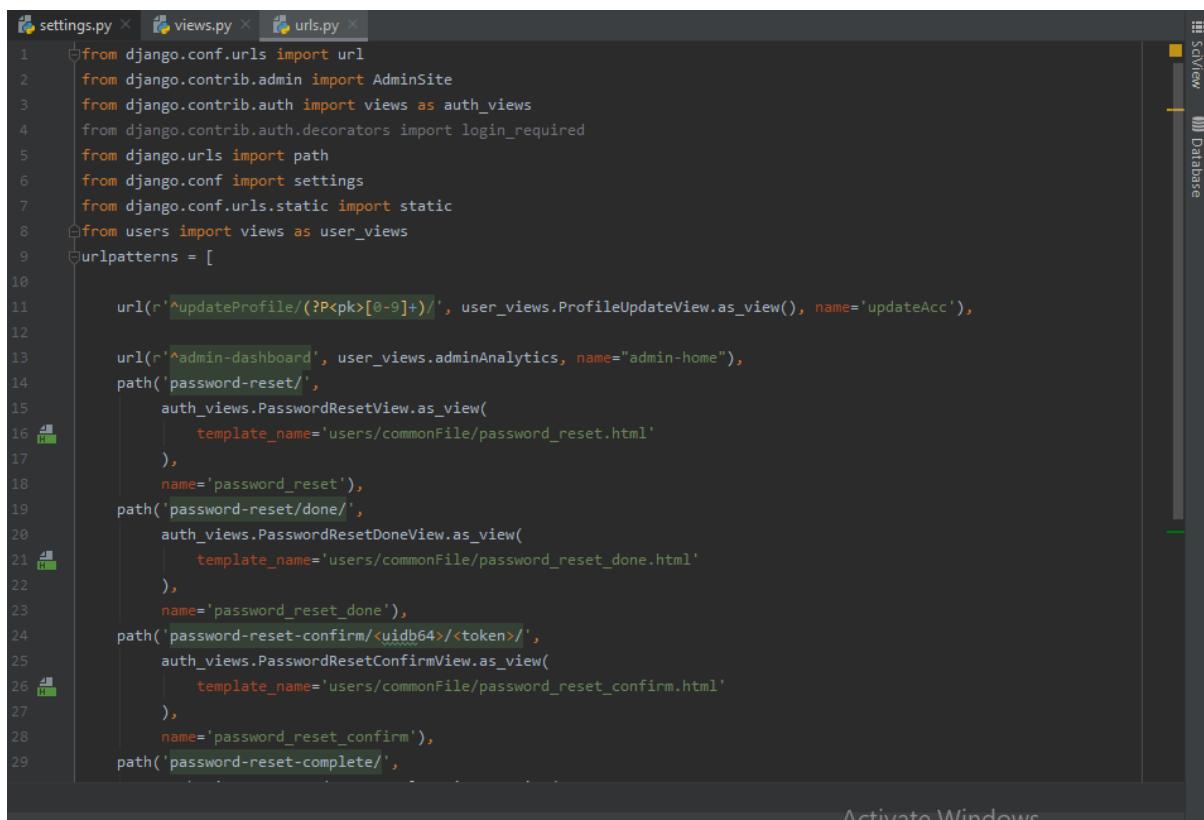
Figure 141: Django Settings

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



```
settings.py x views.py x
1   from django.db.models import Count
2   from django.http import HttpResponseRedirect
3   from django.shortcuts import render, redirect
4   from django.contrib import messages
5   from django.contrib.auth.decorators import login_required
6   from django.views.generic import UpdateView
7
8   from users.models import *
9   from website.models import *
10  from .forms import UserRegisterForm
11  from website.views import check, reverse
12  from django.contrib import auth
13
14  def register(request):
15      if request.method == 'POST':
16          form = UserRegisterForm(request.POST)
17          if form.is_valid():
18              user = form.save()
19              username = form.cleaned_data.get('username')
20              first_name = form.cleaned_data.get('first_name')
21              last_name = form.cleaned_data.get('last_name')
22              print(request.POST.get('role'))
23              if request.POST.get('role') == "Admin":
24                  user.is_staff = True
25                  user.is_active=False
26                  user.save()
27              elif request.POST.get('role') == "SuperAdmin":
28                  user.is_staff = True
29                  user.is_superuser = True
30
31      showNotify()
```

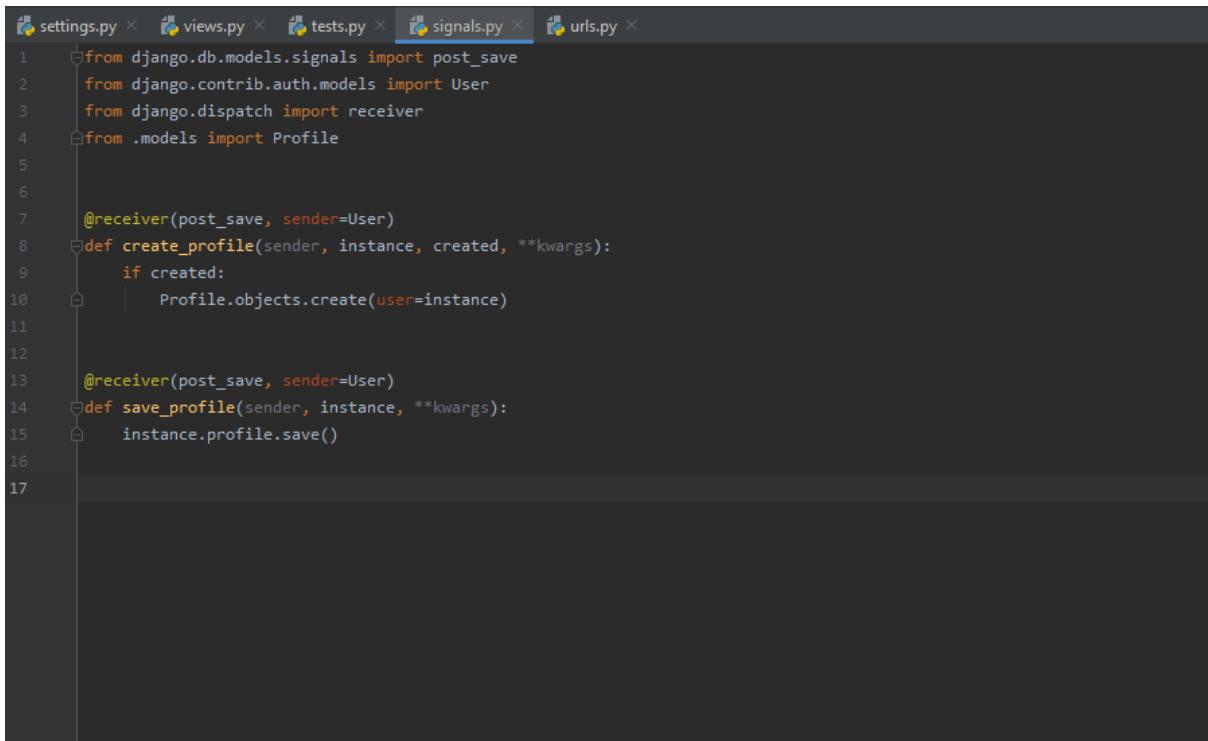
Figure 142:User Views



```
settings.py x views.py x urls.py x
1   from django.conf.urls import url
2   from django.contrib.admin import AdminSite
3   from django.contrib.auth import views as auth_views
4   from django.contrib.auth.decorators import login_required
5   from django.urls import path
6   from django.conf import settings
7   from django.conf.urls.static import static
8   from users import views as user_views
9   urlpatterns = [
10
11     url(r'^updateProfile/(?P<pk>[0-9]+)/$', user_views.ProfileUpdateView.as_view(), name='updateAcc'),
12
13     url(r'^admin-dashboard$', user_views.adminAnalytics, name="admin-home"),
14     path('password-reset/',
15         auth_views.PasswordResetView.as_view(
16             template_name='users/commonFile/password_reset.html'
17         ),
18         name='password_reset'),
19     path('password-reset/done/',
20         auth_views.PasswordResetDoneView.as_view(
21             template_name='users/commonFile/password_reset_done.html'
22         ),
23         name='password_reset_done'),
24     path('password-reset-confirm/<uidb64>/<token>/',
25         auth_views.PasswordResetConfirmView.as_view(
26             template_name='users/commonFile/password_reset_confirm.html'
27         ),
28         name='password_reset_confirm'),
29     path('password-reset-complete/','
```

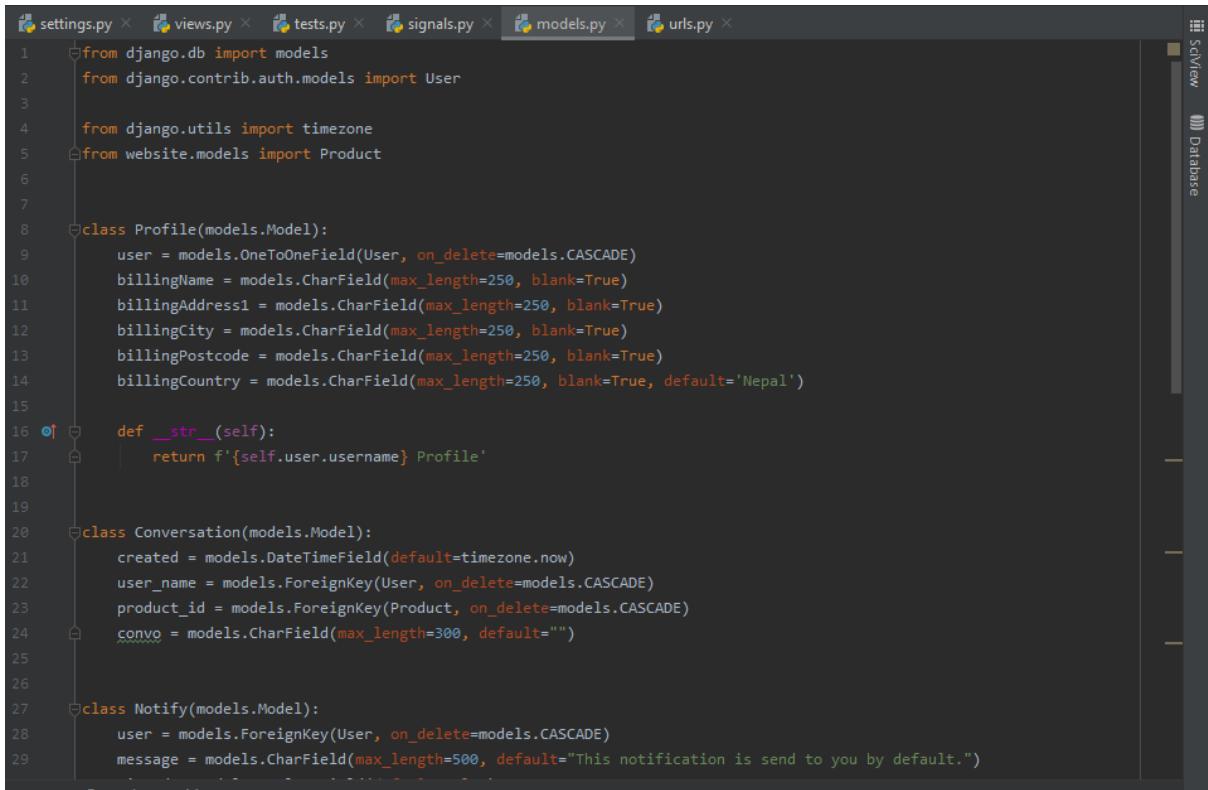
Figure 143:User urls

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



```
settings.py × views.py × tests.py × signals.py × urls.py ×
1  from django.db.models.signals import post_save
2  from django.contrib.auth.models import User
3  from django.dispatch import receiver
4  from .models import Profile
5
6
7  @receiver(post_save, sender=User)
8  def create_profile(sender, instance, created, **kwargs):
9      if created:
10         Profile.objects.create(user=instance)
11
12
13  @receiver(post_save, sender=User)
14  def save_profile(sender, instance, **kwargs):
15      instance.profile.save()
16
17
```

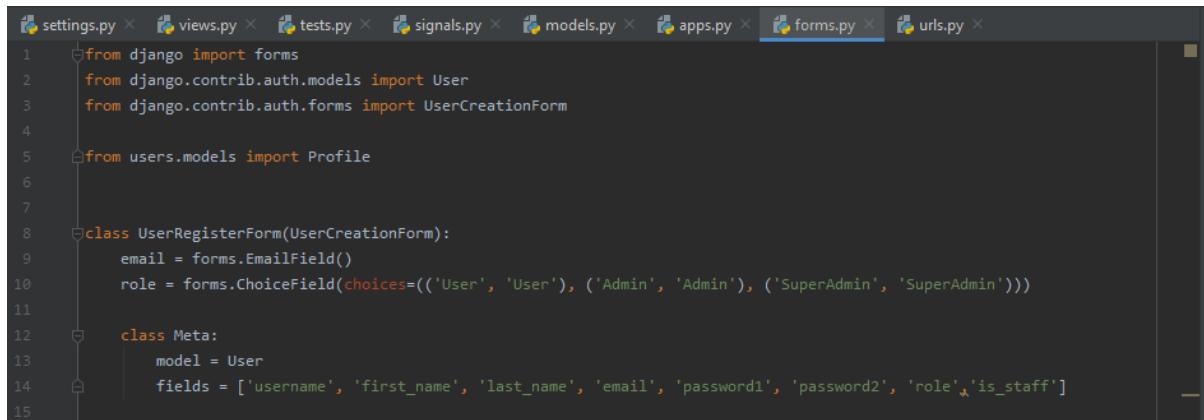
Figure 144: Signals



```
settings.py × views.py × tests.py × signals.py × models.py × urls.py ×
1  from django.db import models
2  from django.contrib.auth.models import User
3
4  from django.utils import timezone
5  from website.models import Product
6
7
8  class Profile(models.Model):
9      user = models.OneToOneField(User, on_delete=models.CASCADE)
10     billingName = models.CharField(max_length=250, blank=True)
11     billingAddress1 = models.CharField(max_length=250, blank=True)
12     billingCity = models.CharField(max_length=250, blank=True)
13     billingPostcode = models.CharField(max_length=250, blank=True)
14     billingCountry = models.CharField(max_length=250, blank=True, default='Nepal')
15
16     def __str__(self):
17         return f'{self.user.username} Profile'
18
19
20  class Conversation(models.Model):
21      created = models.DateTimeField(default=timezone.now)
22      user_name = models.ForeignKey(User, on_delete=models.CASCADE)
23      product_id = models.ForeignKey(Product, on_delete=models.CASCADE)
24      convo = models.CharField(max_length=300, default="")
25
26
27  class Notify(models.Model):
28      user = models.ForeignKey(User, on_delete=models.CASCADE)
29      message = models.CharField(max_length=500, default="This notification is send to you by default.")
```

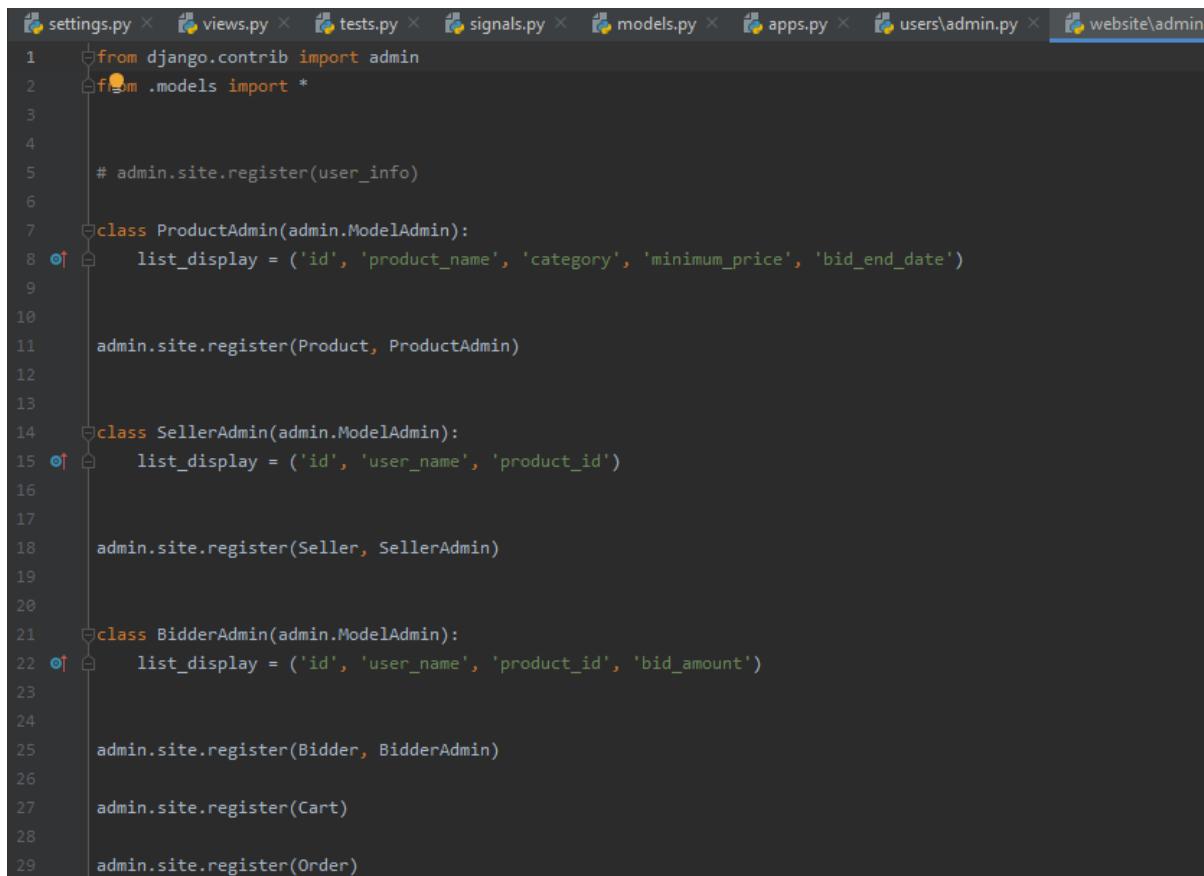
Figure 145: Models

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



```
settings.py × views.py × tests.py × signals.py × models.py × apps.py × forms.py × urls.py ×
1 from django import forms
2 from django.contrib.auth.models import User
3 from django.contrib.auth.forms import UserCreationForm
4
5 from users.models import Profile
6
7
8 class UserRegisterForm(UserCreationForm):
9     email = forms.EmailField()
10    role = forms.ChoiceField(choices=((('User', 'User'), ('Admin', 'Admin'), ('SuperAdmin', 'SuperAdmin'))))
11
12    class Meta:
13        model = User
14        fields = ['username', 'first_name', 'last_name', 'email', 'password1', 'password2', 'role', 'is_staff']
15
```

Figure 146:Forms



```
settings.py × views.py × tests.py × signals.py × models.py × apps.py × users\admin.py × website\admin.py ×
1 from django.contrib import admin
2 from .models import *
3
4
5 # admin.site.register(user_info)
6
7 class ProductAdmin(admin.ModelAdmin):
8     list_display = ('id', 'product_name', 'category', 'minimum_price', 'bid_end_date')
9
10 admin.site.register(Product, ProductAdmin)
11
12
13
14 class SellerAdmin(admin.ModelAdmin):
15     list_display = ('id', 'user_name', 'product_id')
16
17
18 admin.site.register(Seller, SellerAdmin)
19
20
21 class BidderAdmin(admin.ModelAdmin):
22     list_display = ('id', 'user_name', 'product_id', 'bid_amount')
23
24
25 admin.site.register(Bidder, BidderAdmin)
26
27 admin.site.register(Cart)
28
29 admin.site.register(Order)
```

Figure 147:Admin

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

A screenshot of a code editor showing Python code for class-based views. The file is named `classviews.py`. The code defines a class `AddProductView` that inherits from `CreateView`. It imports various Django modules and models. The `fields` attribute is set to a list of product-related fields. The `template_name` attribute specifies the template to use. The `get_context_data` method adds distinct categories to the context. The code is annotated with several TODO and note markers.

```
1  from datetime import datetime
2
3  from django.contrib import messages
4  from django.db.models import Max
5  from django.db.models import Q
6  from django.http import HttpResponseRedirect
7  from django.urls import reverse
8  from django.utils import timezone
9  from django.views.generic import ListView
10 from django.views.generic.detail import DetailView
11 from django.views.generic.edit import CreateView
12 from django.views.generic.edit import DeleteView
13 from django.views.generic.edit import UpdateView
14
15 from users.models import Conversation, History, Favourites, Notify
16
17 from .models import Product, Seller, Bidder
18
19
20 class AddProductView(CreateView):
21     model = Product
22     fields = ["product_name", "category", "minimum_price", "bid_end_date", "status", "image", "image2", "image3",
23               "description"]
24
25     template_name = 'website/product_form.html'
26
27     def get_context_data(self, **kwargs):
28         context = super(AddProductView, self).get_context_data(**kwargs)
29         context['categories'] = Product.objects.values('category').distinct()
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1167
1168
1169
1170
1171
1172
1173
1174
1175
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1290
1291
1292
1293
1294
1295
1295
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1390
1391
1392
1393
1394
1395
1395
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1490
1491
1492
1493
1494
1495
1495
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1590
1591
1592
1593
1594
1595
1595
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1690
1691
1692
1693
1694
1695
1695
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1790
1791
1792
1793
1794
1795
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1889
1890
1891
1892
1893
1894
1895
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1987
1988
1989
1990
1991
1992
1993
1994
1995
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2087
2088
2089
2090
2091
2092
2093
2094
2095
2095
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2177

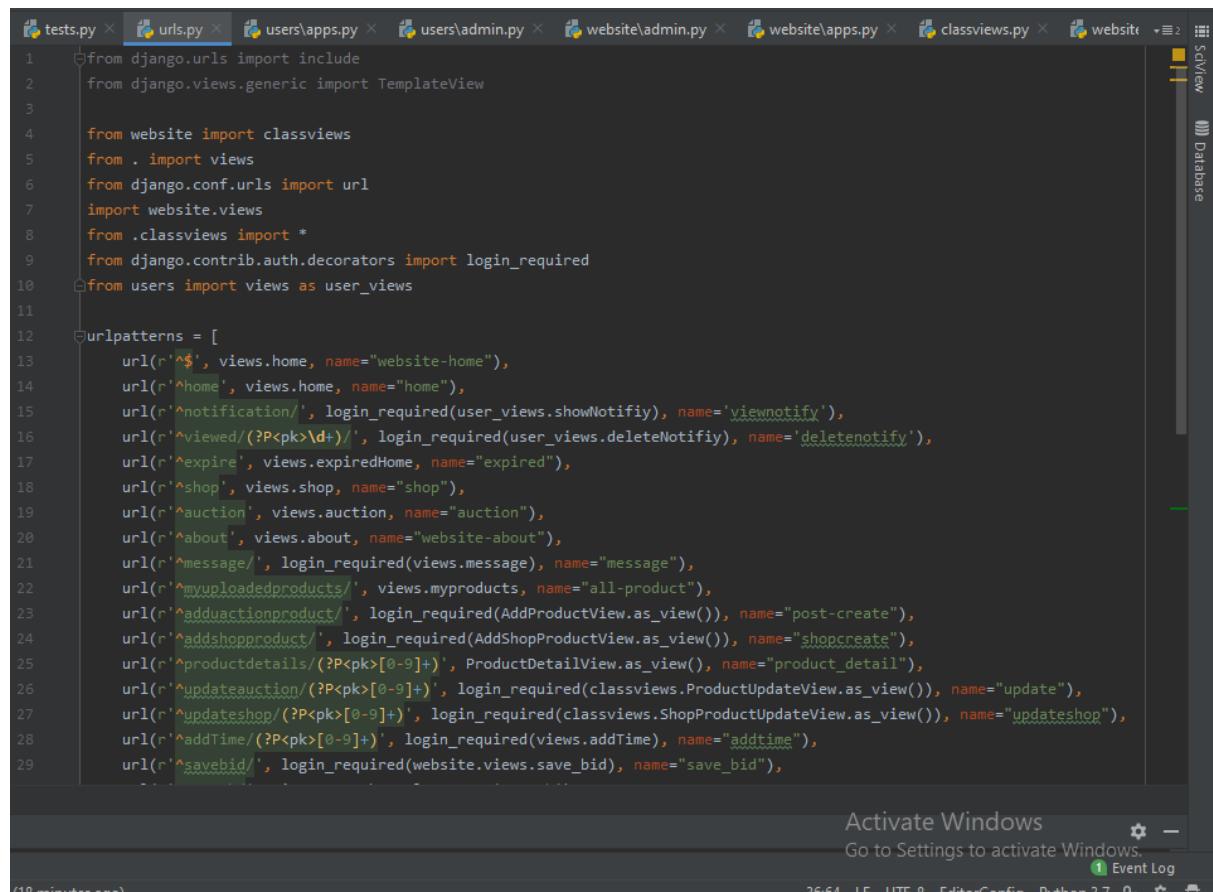
```

The screenshot shows a code editor with a dark theme. The file being edited is `tests.py`, which contains Django test code. The code includes imports for `TestCase` and `reverse` from `django.test` and `django.urls` respectively. It also imports `UserRegisterForm` and `Profile` from `users.forms` and `users.models`. The code defines two test classes: `RegisterProfile2Test` and `RegisterProfileTest`. Both classes have a `create_user` method that returns a user object with the username 'kushal'. The `test_create_user` method in `RegisterProfile2Test` asserts that the created user has a corresponding profile object. The `test_login()` method is shown at the bottom of the stack.

```
1  from django.test import TestCase
2
3  # Create your tests here.
4  from django.urls import reverse
5
6  from users.forms import UserRegisterForm
7  from .models import *
8
9  from users.models import Profile, Notify
10
11
12 class RegisterProfile2Test(TestCase):
13
14     def create_user(self):
15         return User.objects.create(username='kushal')
16
17     def test_create_user(self):
18         a = self.create_user()
19         b = Profile.objects.last()
20         self.assertTrue(isinstance(a, User))
21         self.assertTrue(isinstance(b, Profile))
22         self.assertEqual(a, b.user)
23
24
25 class RegisterProfileTest(TestCase):
26
27     def create_user(self):
28         return User.objects.create(username='kushal')
```

Figure 150:Tests

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



```
tests.py × urls.py × users\apps.py × users\admin.py × website\admin.py × website\apps.py × classviews.py × website × SciView Database
1 from django.urls import include
2 from django.views.generic import TemplateView
3
4 from website import classviews
5 from . import views
6 from django.conf.urls import url
7 import website.views
8 from .classviews import *
9 from django.contrib.auth.decorators import login_required
10 from users import views as user_views
11
12 urlpatterns = [
13     url(r'^$', views.home, name="website-home"),
14     url(r'^home', views.home, name="home"),
15     url(r'^notification/$', login_required(user_views.showNotify), name='viewnotify'),
16     url(r'^viewed/(?P<pk>\d+)/$', login_required(user_views.deleteNotify), name='deletenotify'),
17     url(r'^expire', views.expiredHome, name="expired"),
18     url(r'^shop', views.shop, name="shop"),
19     url(r'^auction', views.auction, name="auction"),
20     url(r'^about', views.about, name="website-about"),
21     url(r'^message/$', login_required(views.message), name="message"),
22     url(r'^myuploadedproducts/$', views.myproducts, name="all-product"),
23     url(r'^addauctionproduct/$', login_required(AddProductView.as_view()), name="post-create"),
24     url(r'^addshopproduct/$', login_required(AddShopProductView.as_view()), name="shopcreate"),
25     url(r'^productdetails/(?P<pk>[0-9]+)$', ProductDetailView.as_view(), name="product_detail"),
26     url(r'^updateauction/(?P<pk>[0-9]+)$', login_required(classviews.ProductUpdateView.as_view()), name="update"),
27     url(r'^updateshop/(?P<pk>[0-9]+)$', login_required(classviews.ShopProductUpdateView.as_view()), name="updateshop"),
28     url(r'^addTime/(?P<pk>[0-9]+)$', login_required(views.addTime), name="addtime"),
29     url(r'^savebid/$', login_required(website.views.save_bid), name="save_bid"),
```

Figure 151: Website URLs

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows a code editor with a dark theme. The file being edited is `views.py`. The code defines a function-based view named `home` that takes a `request` parameter. Inside the function, it calls `check()`, sets the title to 'Home', and checks if the user is authenticated. If authenticated, it retrieves four notifications for the user, counts the number of unviewed notifications, and filters products by status ('shop' or 'auction') and purchase status ('purchased=False'). The code uses Django's ORM and various utility functions from the django.core module.

```
1  from datetime import datetime, timedelta
2
3  import stripe
4
5  from django.conf import settings
6  from django.contrib import messages
7  from django.contrib.auth.decorators import login_required
8  from django.contrib.auth.models import auth
9  from django.core.exceptions import ObjectDoesNotExist
10 from django.core.mail import send_mail
11 from django.core.paginator import Paginator, EmptyPage, PageNotAnInteger
12 from django.db.models import Q, Count
13 from django.http import JsonResponse
14 from django.shortcuts import *
15 from django.utils.timezone import get_current_timezone
16
17 from users.models import *
18 from .models import *
19
20
21 def home(request):
22     check()
23     title = 'Home'
24     if request.user.is_authenticated:
25
26         notify = Notify.objects.filter(user=request.user)[:4]
27         view = Notify.objects.filter(user=request.user, viewed=False).count()
28         shop = Product.objects.filter(status='shop', purchased=False).order_by('id')[:4]
29         auction = Product.objects.filter(status='auction', expired=False).order_by('id')[:4]
```

Figure 152: Function based website views

8.4 APPENDIX D: DESIGNS

8.4.1 GANTT CHART

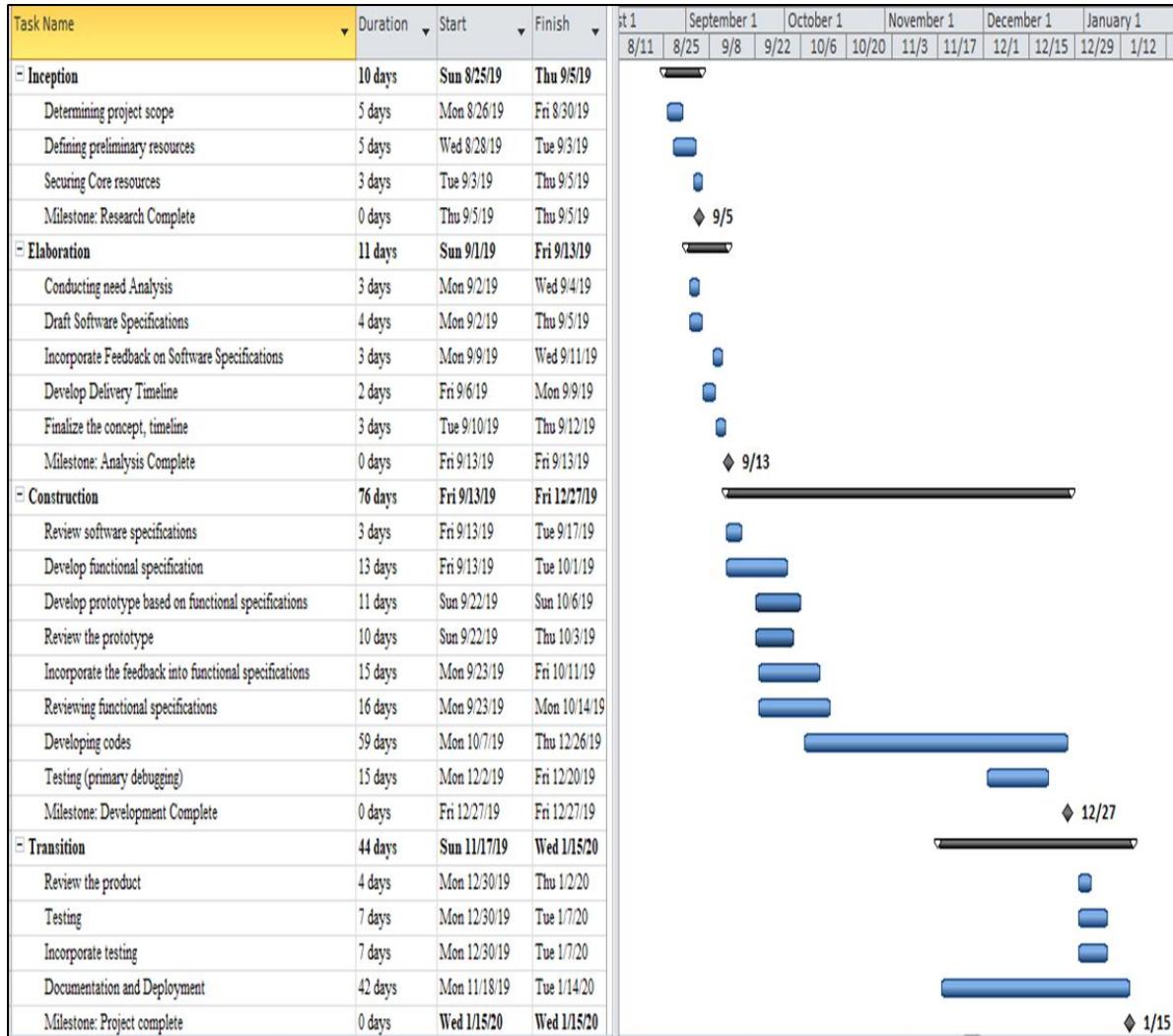


Figure 153: Gantt chart 1

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

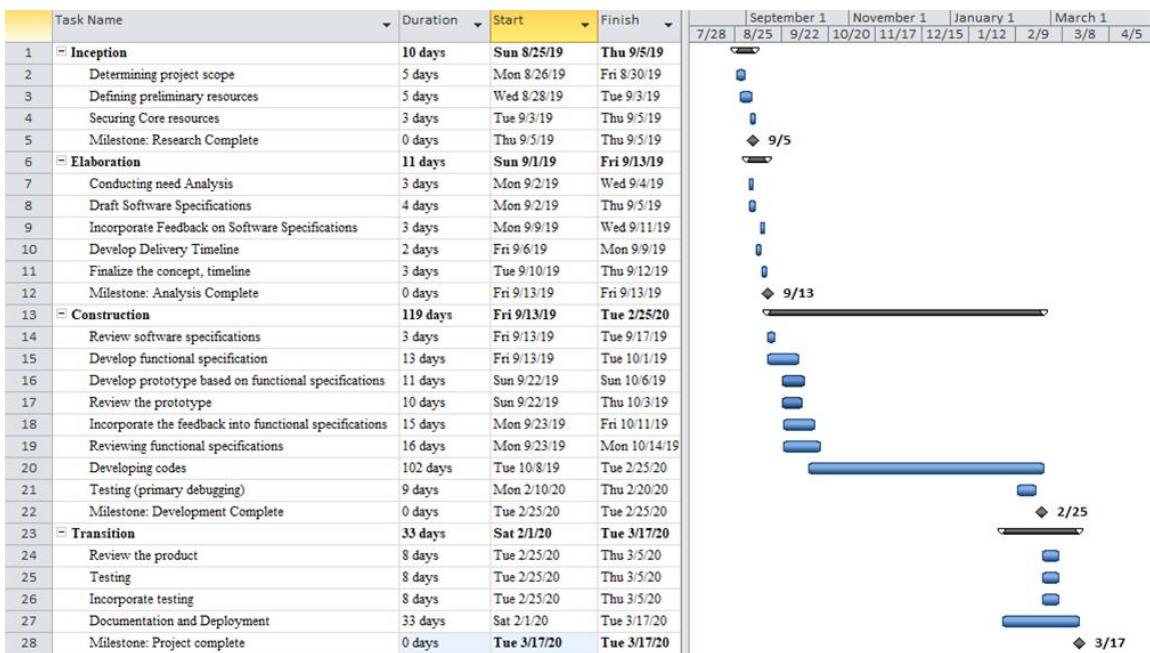


Figure 154: Gantt chart 2

8.4.2 WORK BREAKDOWN STRUCTURE

SN	Structure	Description	Time
1	Inception	Start of the project where research will be conducted and resources will be collected.	10- 15 days
	1.1	Determining project scope	
	1.2	Defining preliminary resources	
	1.3	Securing Core resources	
Research Complete			
2	Elaboration	Analysis of the project, Scope, and model creation, timeline creating use cases.	10- 15 days
	2.1	Conducting need Analysis	
	2.2	Draft Software Specifications	
	2.3	Incorporate Feedback on Software Specifications	
	2.4	Develop models, timeline, strategies	
	2.5	Finalise the concept, timeline, strategies	
Analysis Complete			
3	Construction	Designing and coding of the website, testing, completing website design, checking functionalities, testing is done in this phase.	3 – 4 months
	3.1	Review software specifications	
	3.2	Develop functional specification	
	3.3	Develop prototype based on functional Specifications	
	3.4	Review the prototype	
	3.5	Incorporate the feedback into functional Specifications	
	3.6	Reviewing functional specifications	
	3.7	Developing codes	
	3.8	Testing (primary debugging)	
	Development Complete		
4	Transition	Final Stages of the product and deployment of the website with last of testing and writing documentation.	1 and half month
	4.1	Review the product	
	4.2	Testing	
	4.3	Incorporate testing result	
	4.4	Documentation and Deployment	
Project complete			

Table 27: Work breakdown structure

8.4.3 SEQUENCE DIAGRAMS

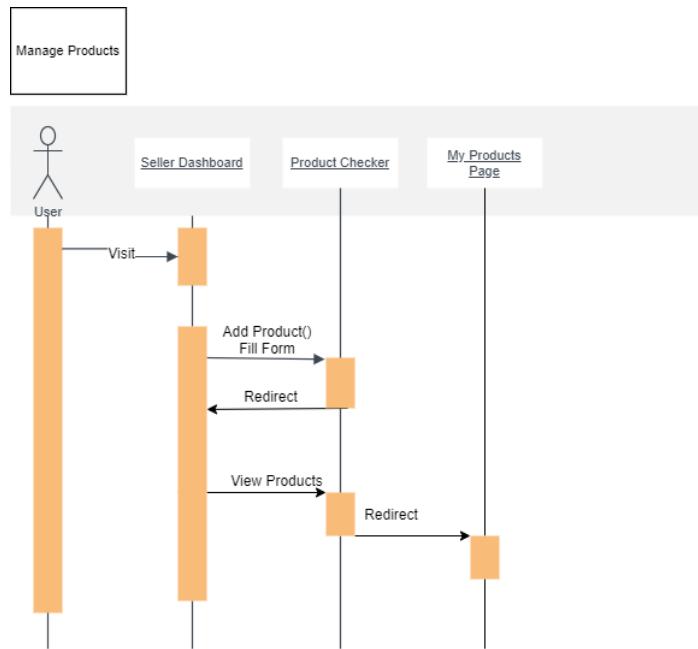


Figure 155:Manage Products

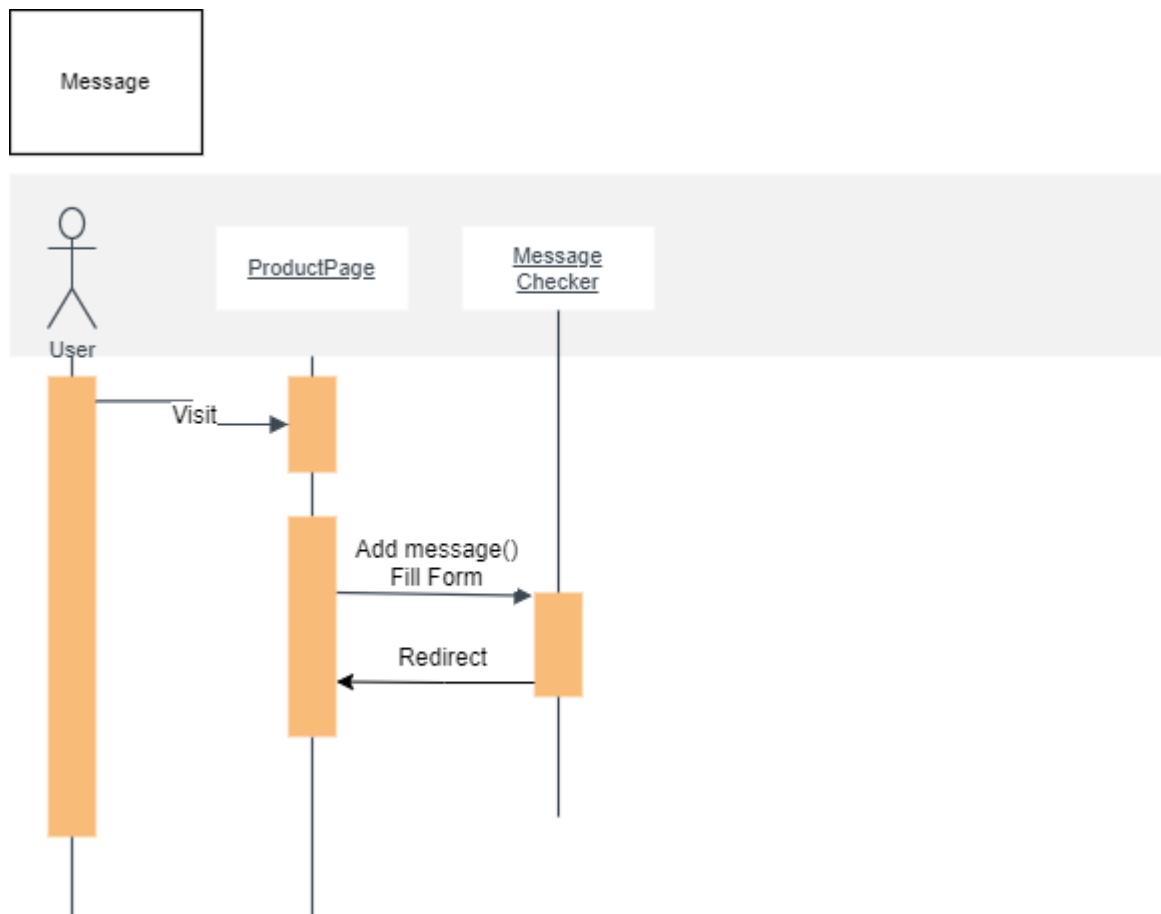


Figure 156:messages

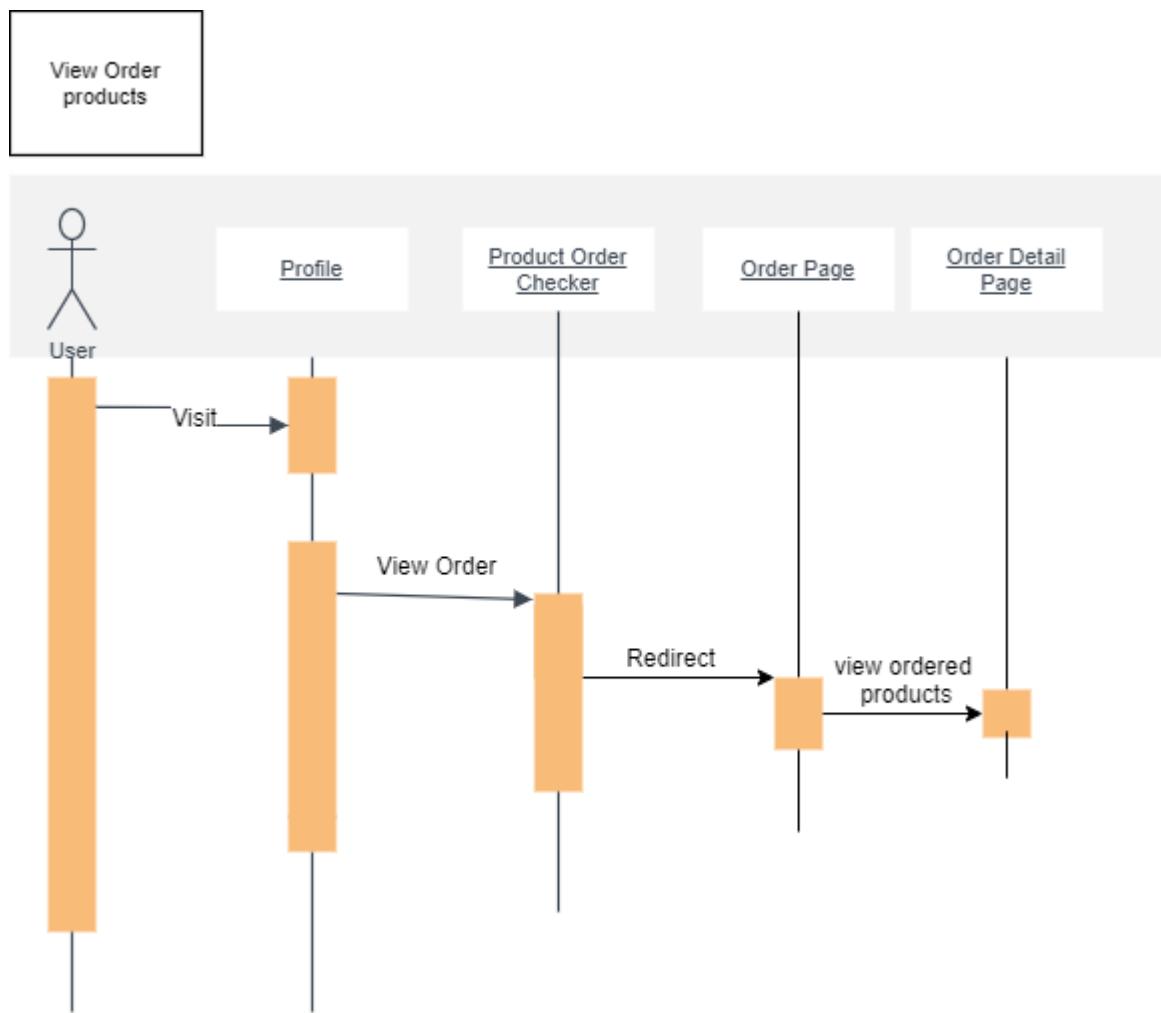


Figure 157:View Order Products

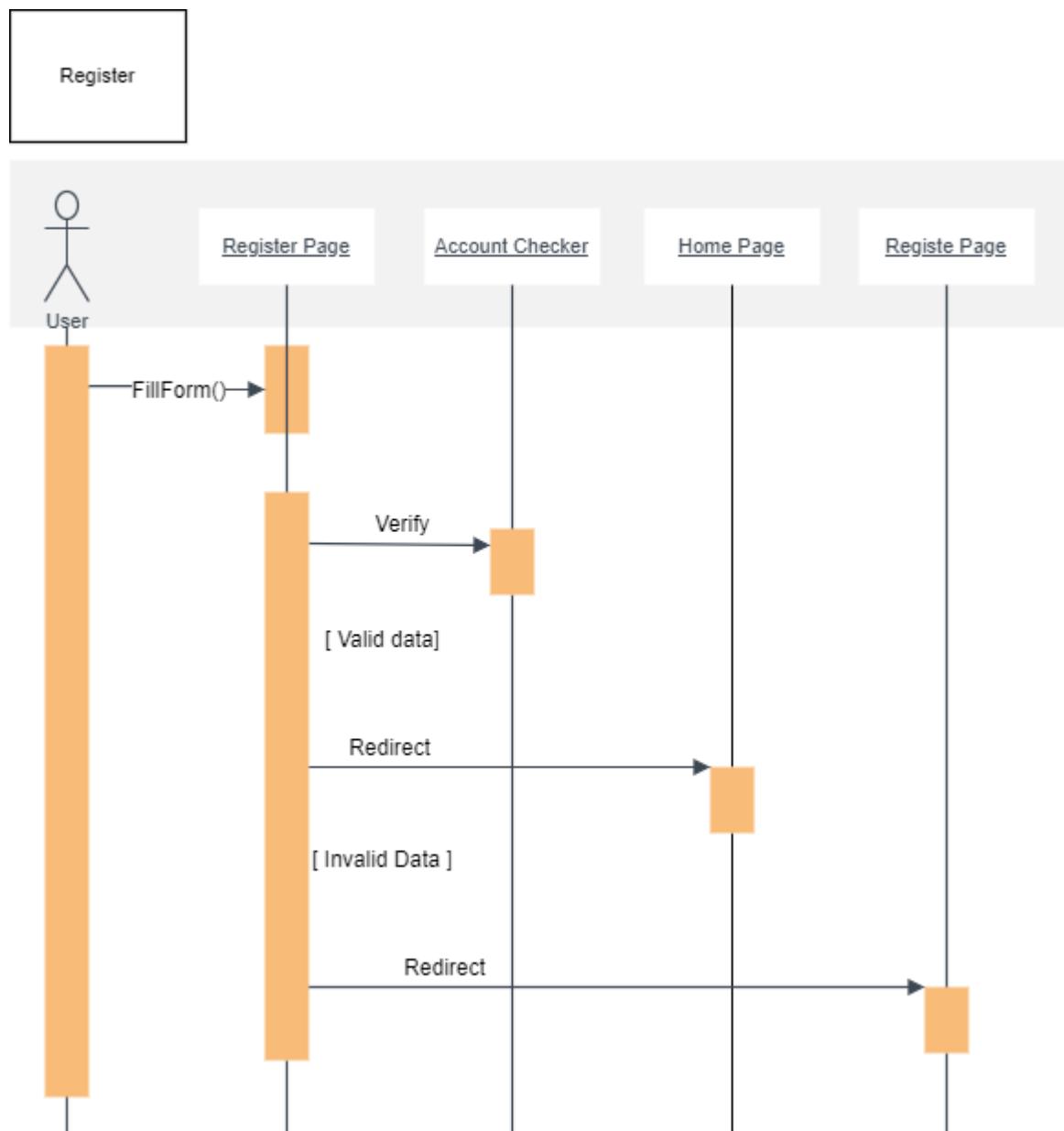


Figure 158: Register

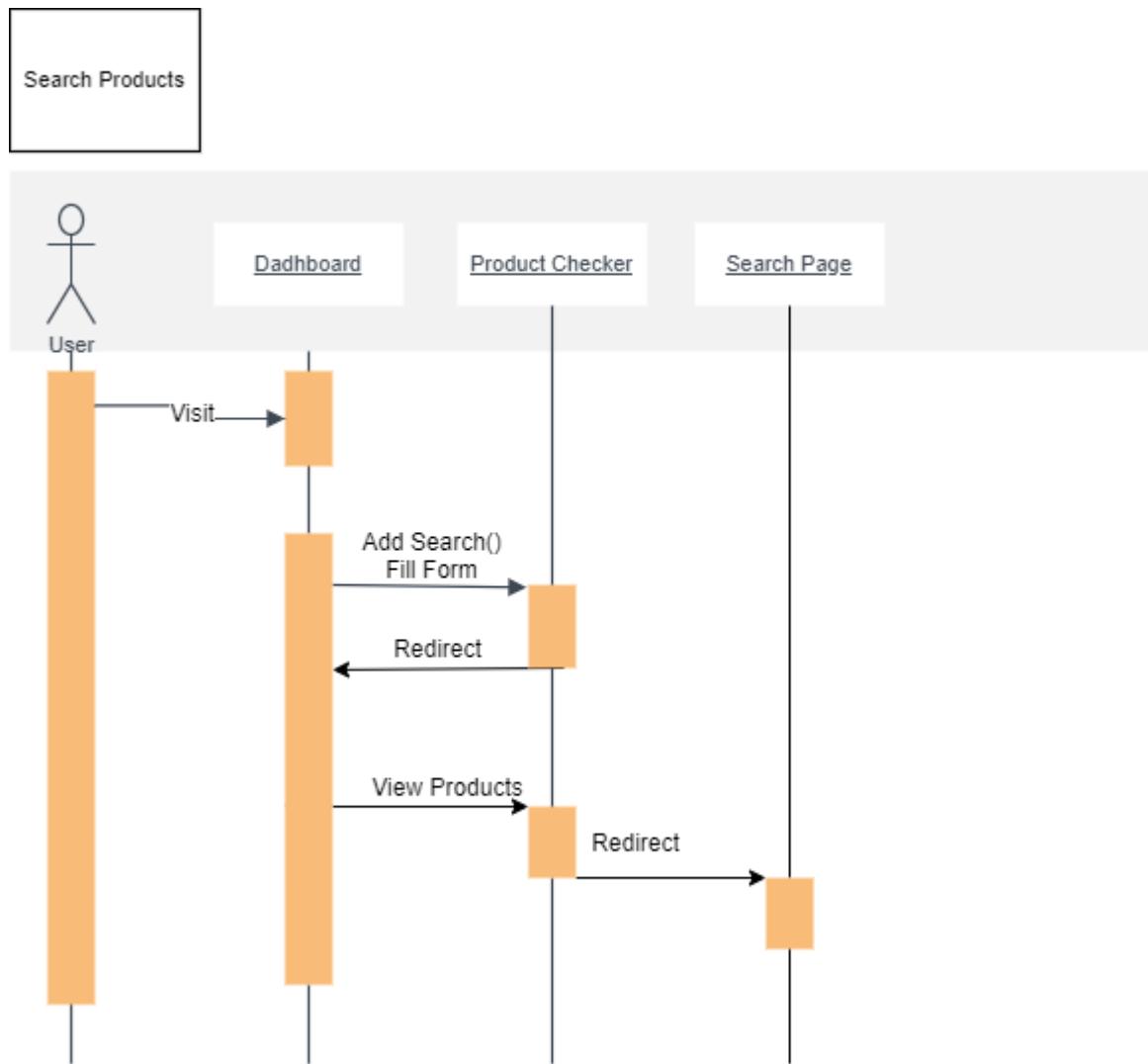


Figure 159: SearchProducts

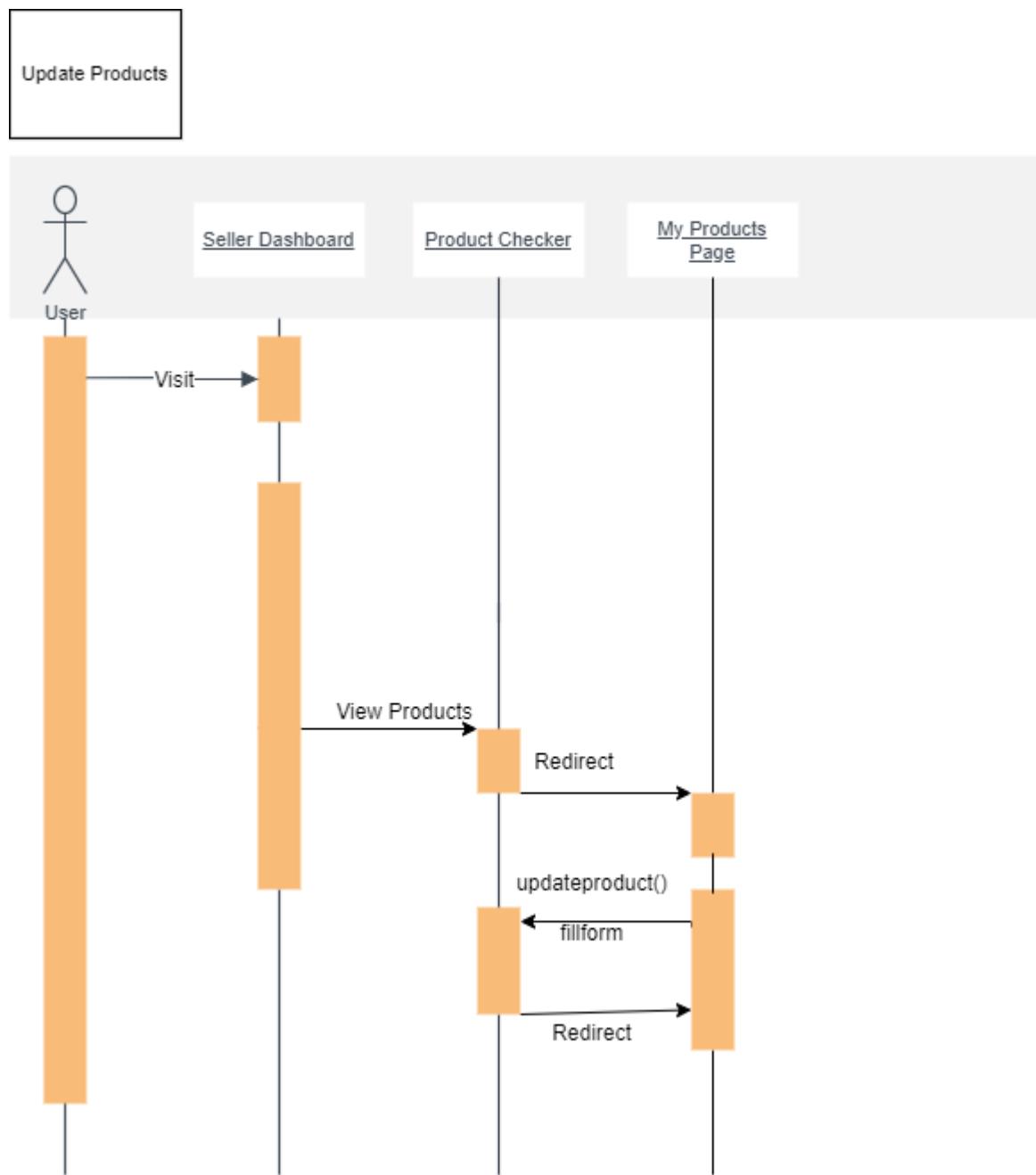


Figure 160: Update Products

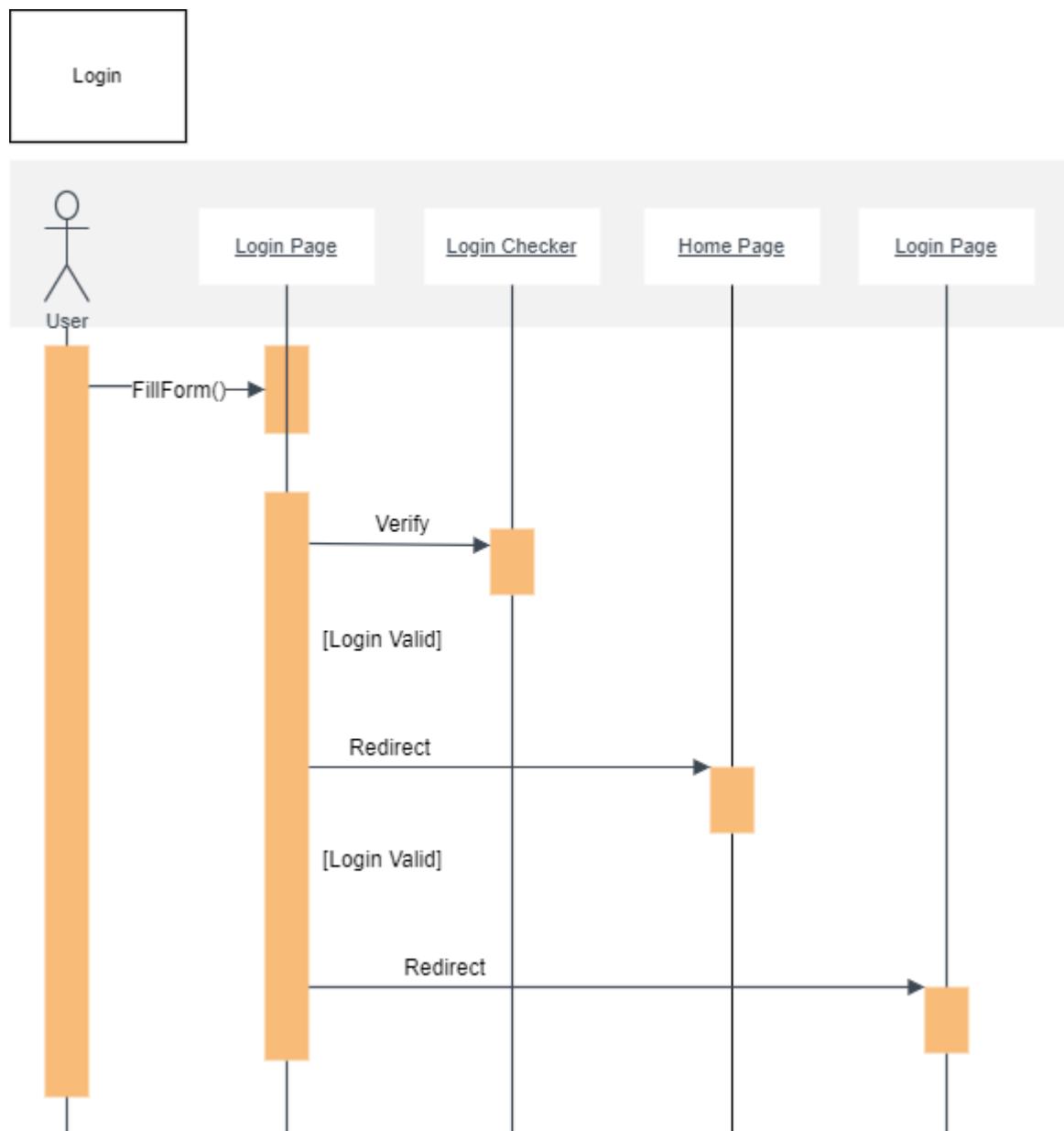


Figure 161: Login

8.4.4 ER-DIAGRAM

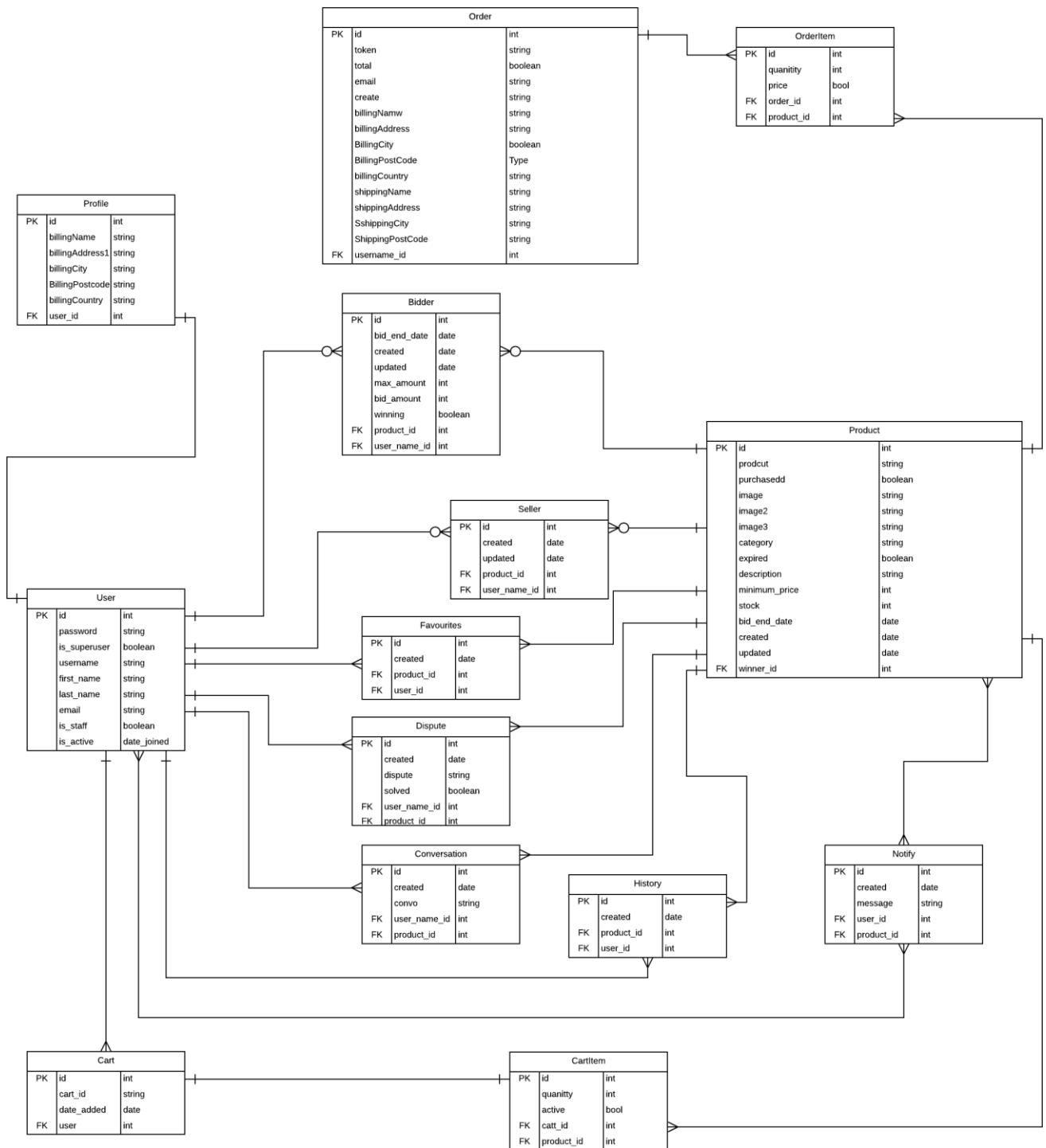


Figure 162:Final ER daigram

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

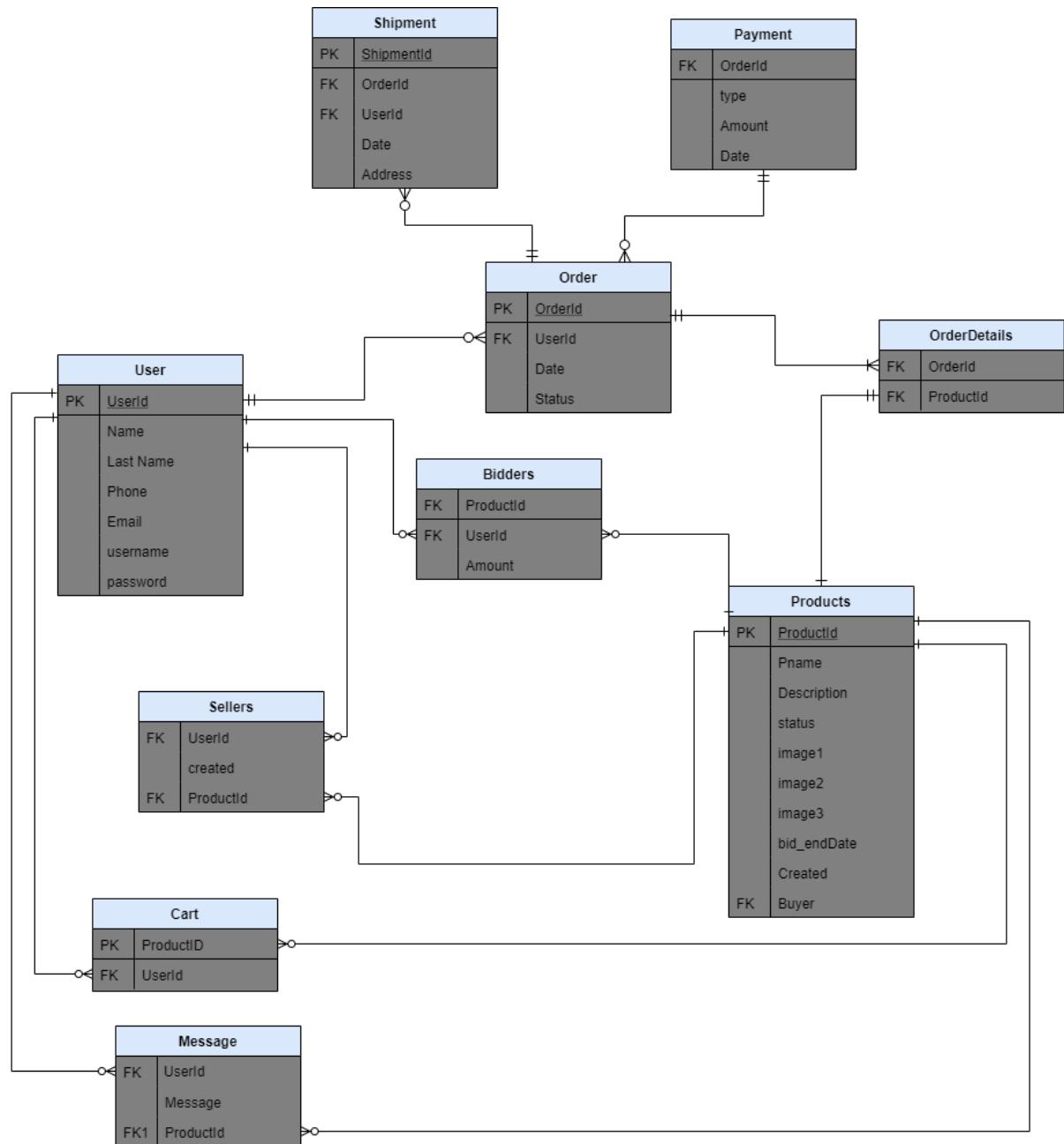
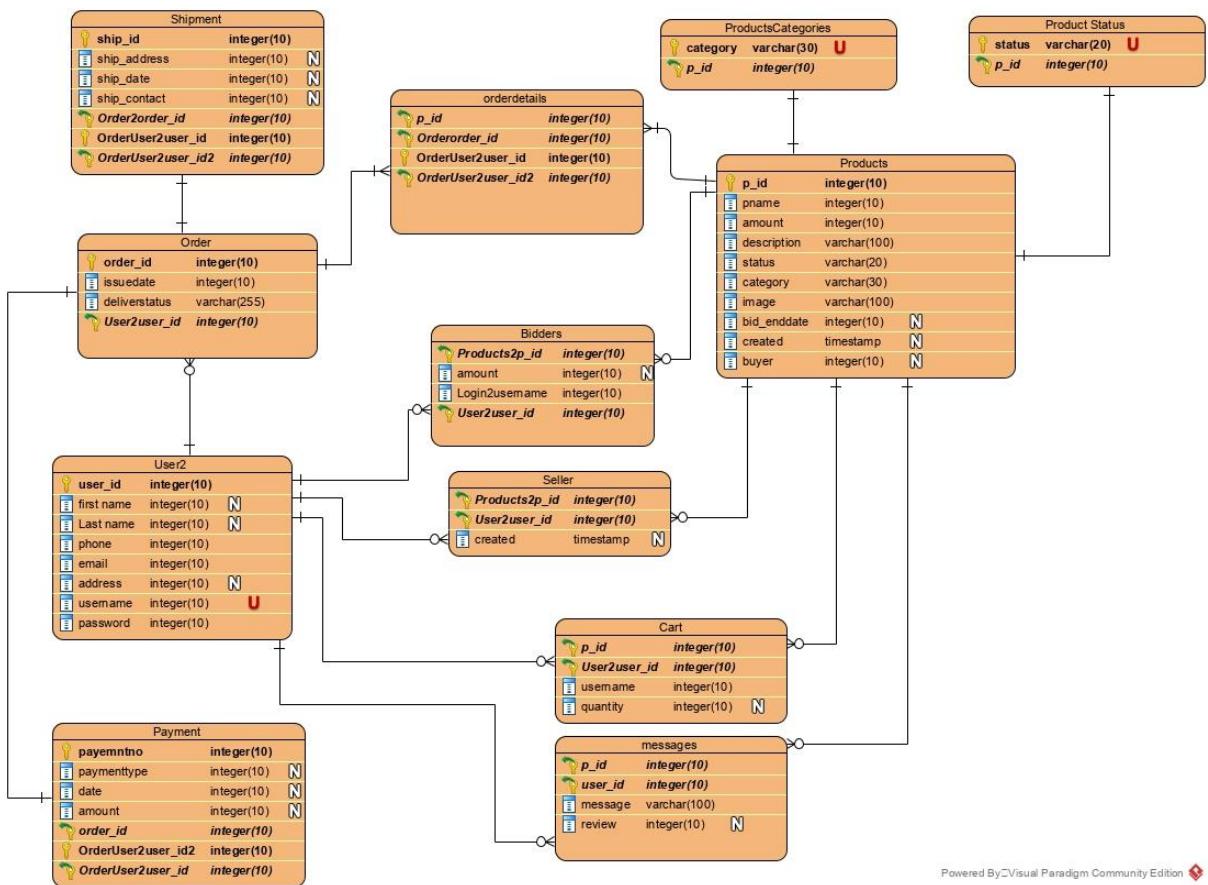


Figure 163: 1st Draft Diagram

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



Powered By Visual Paradigm Community Edition

Figure 164: 2nd Draft

8.4.5 ACTIVITY DIAGRAM

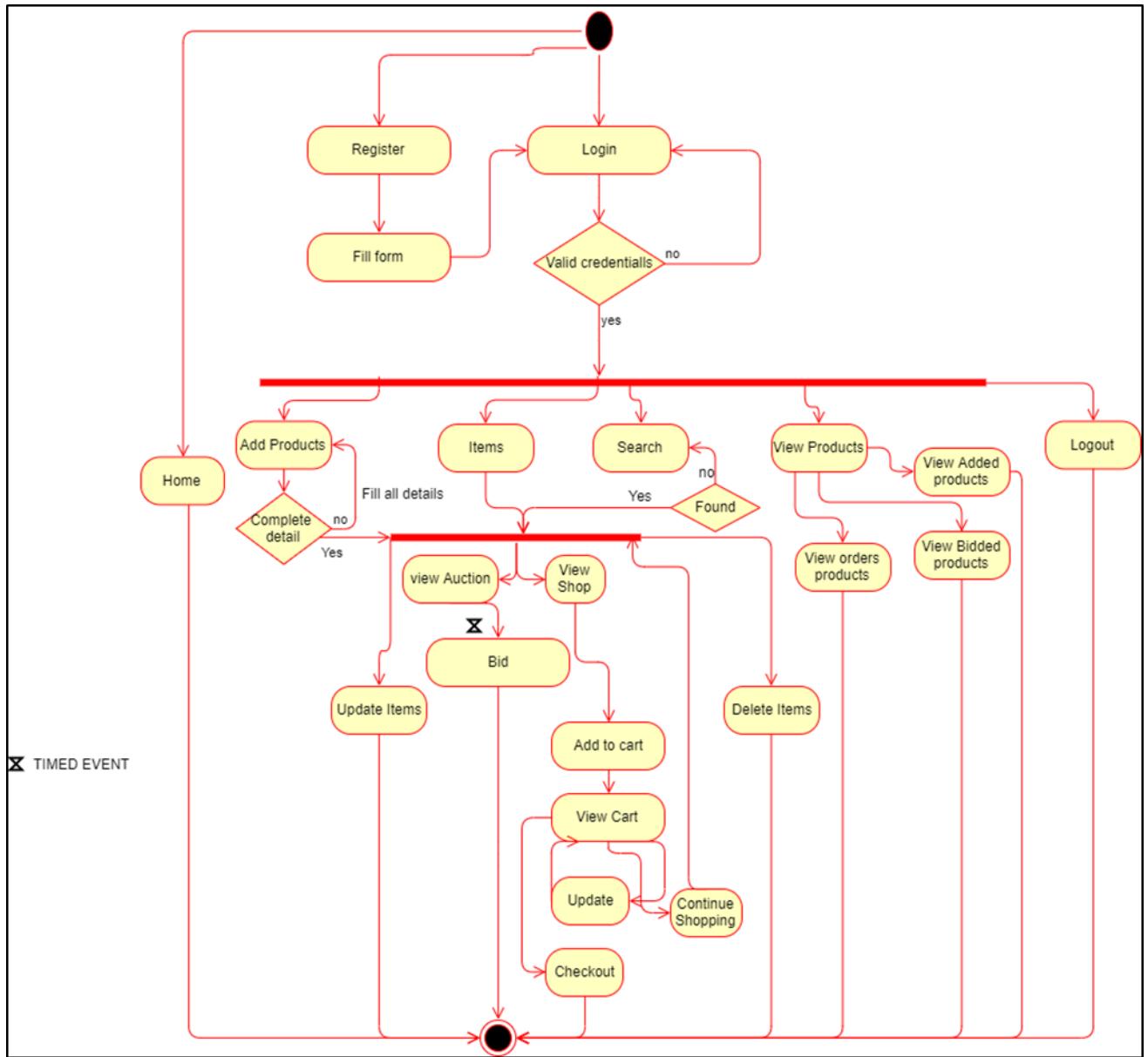


Figure 165:Activity diagram

8.4.6 DATA FLOW DIAGRAMS

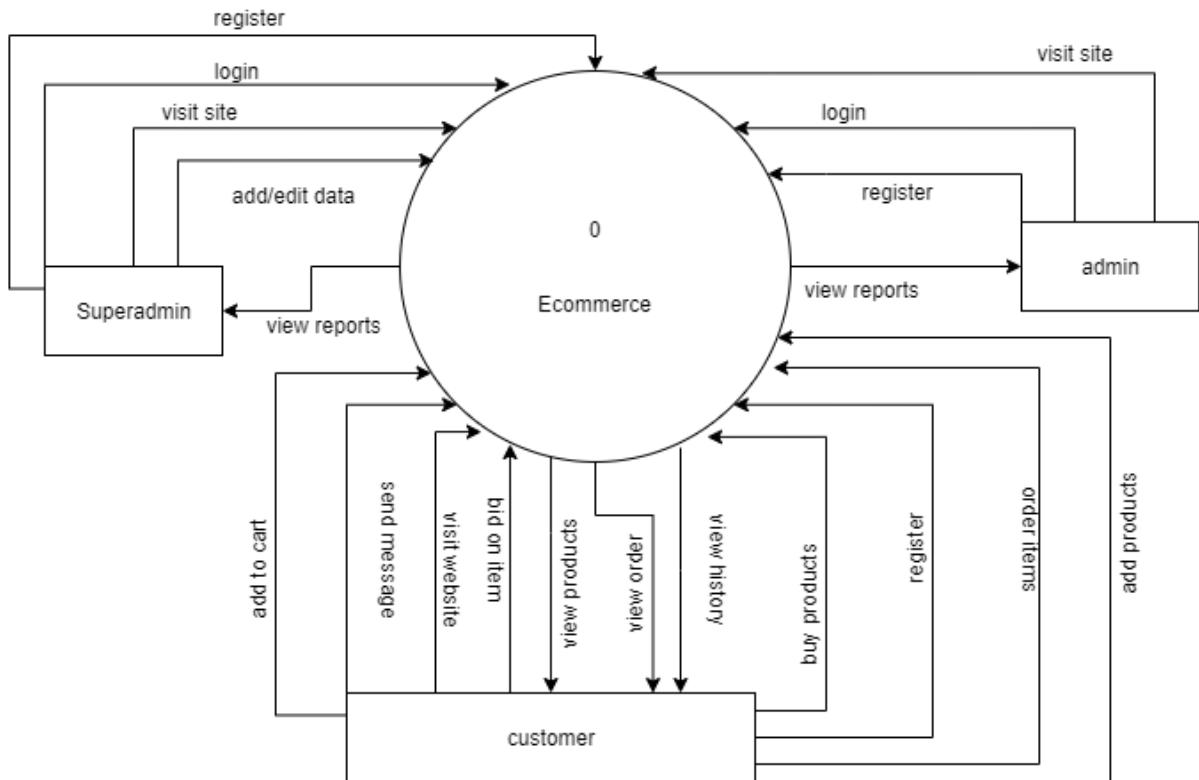


Figure 166: DFD level 0

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

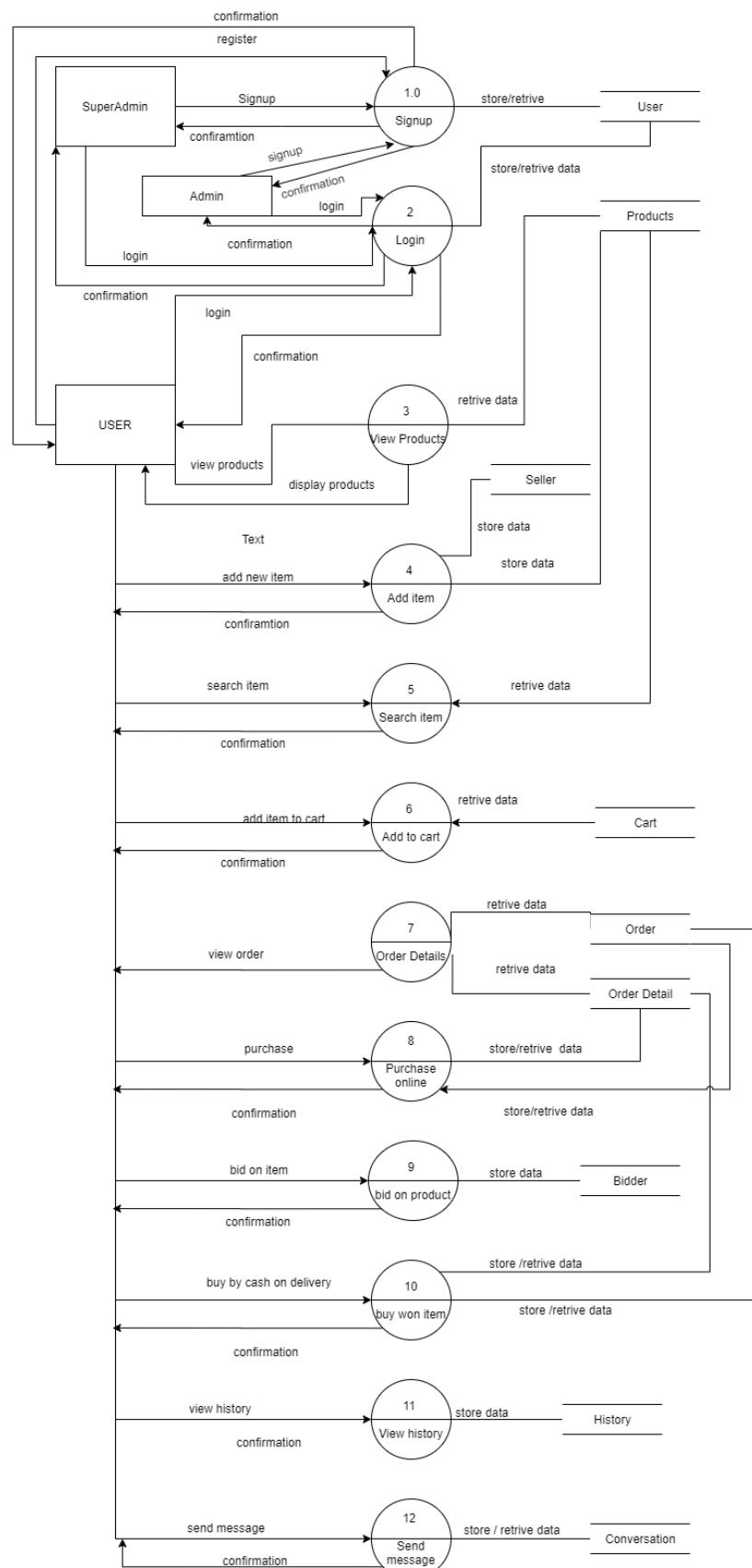


Figure 167: DFD level 1

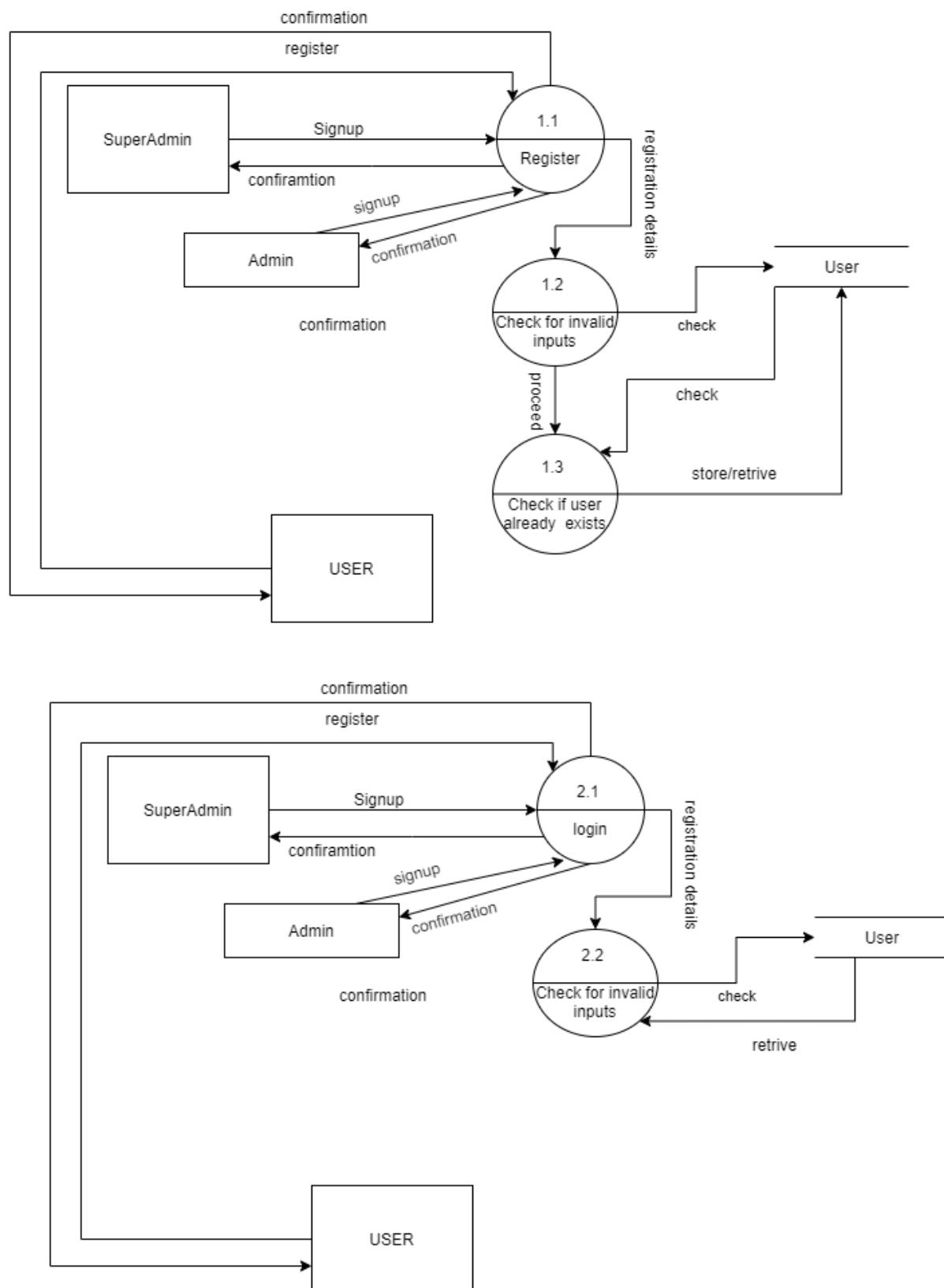


Figure 168: Dfd level1 of function 1 and 2

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

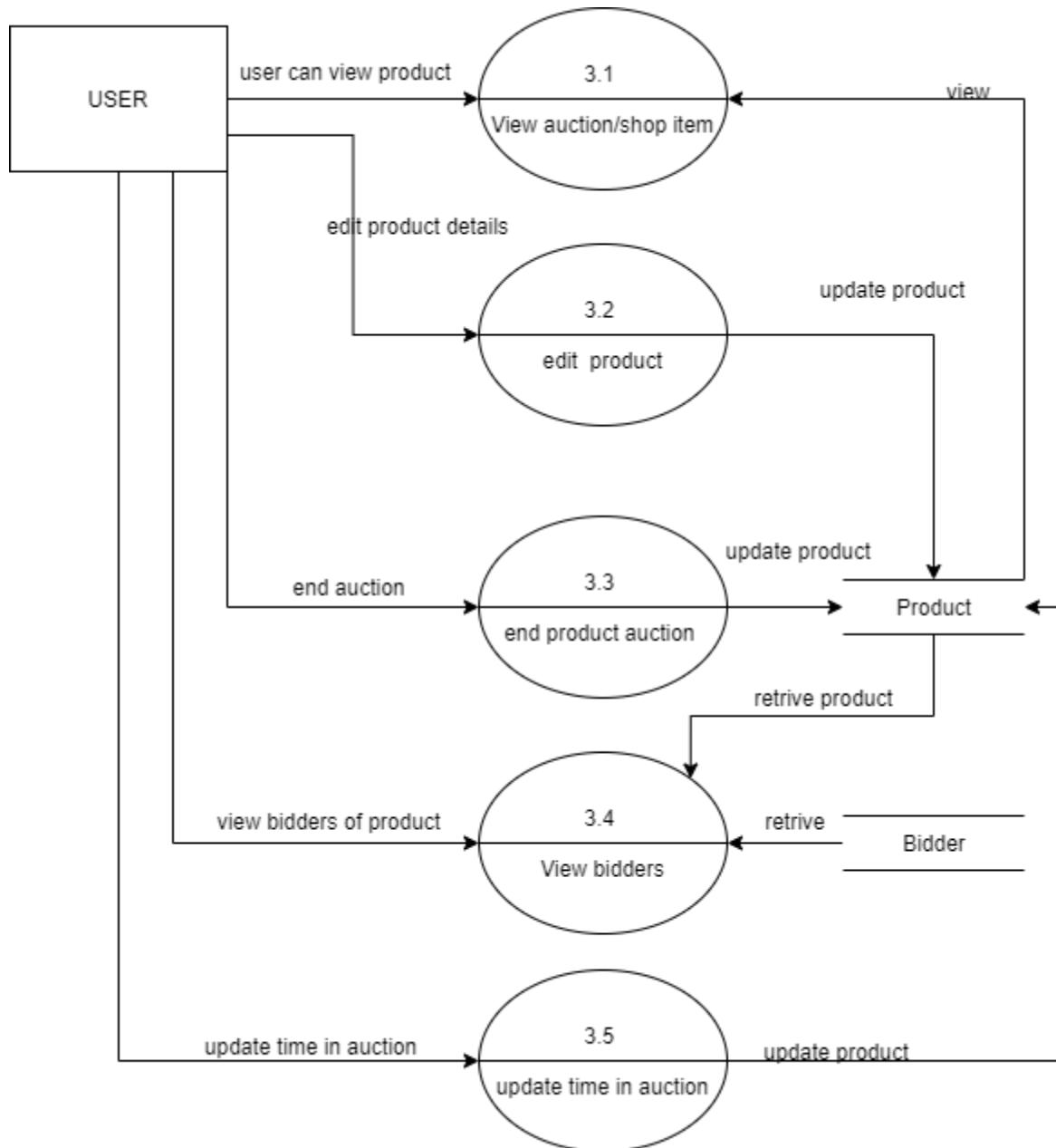


Figure 169:Dfd 3 level 2

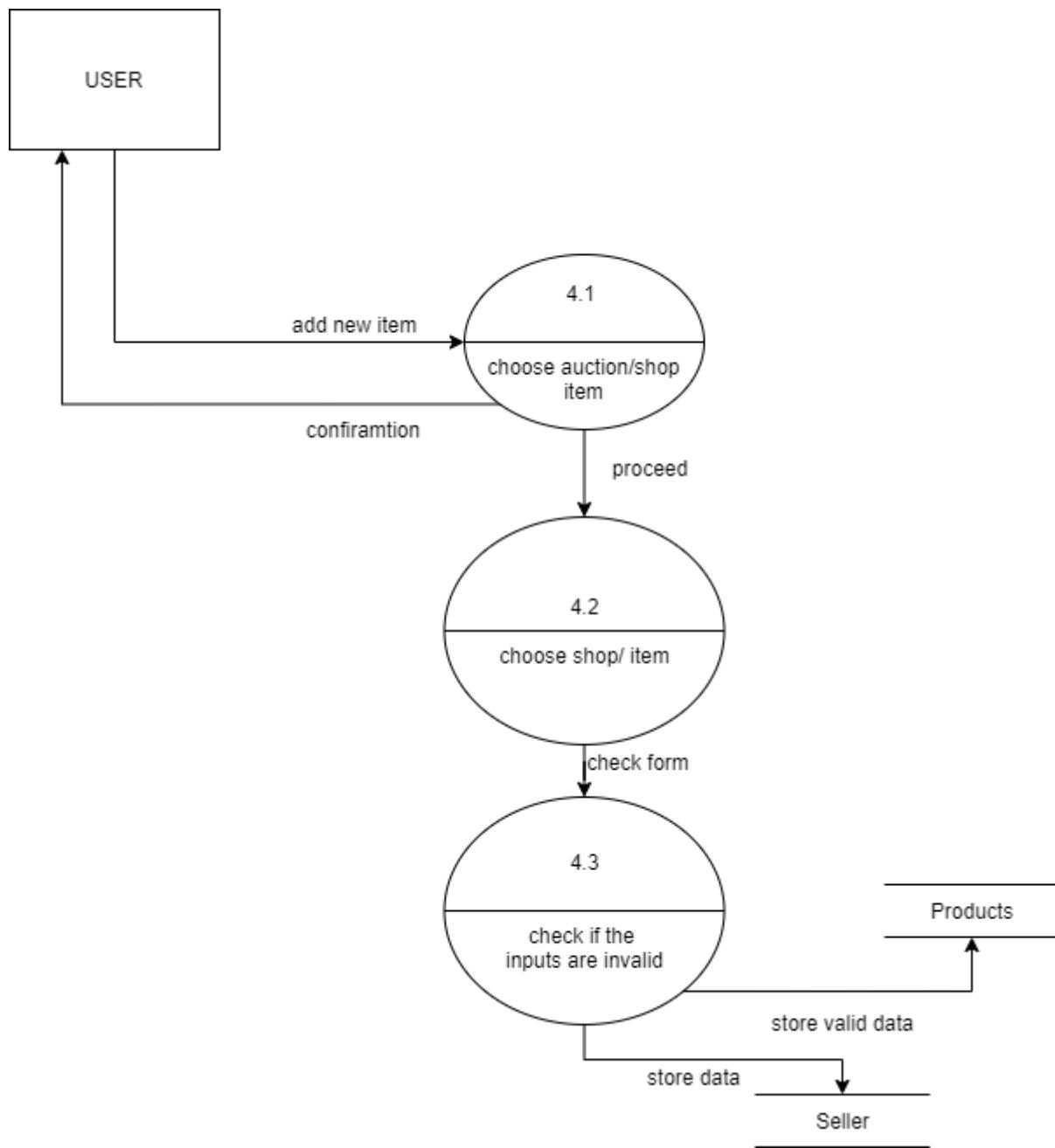


Figure 170: dfd diagram level2 of function 3

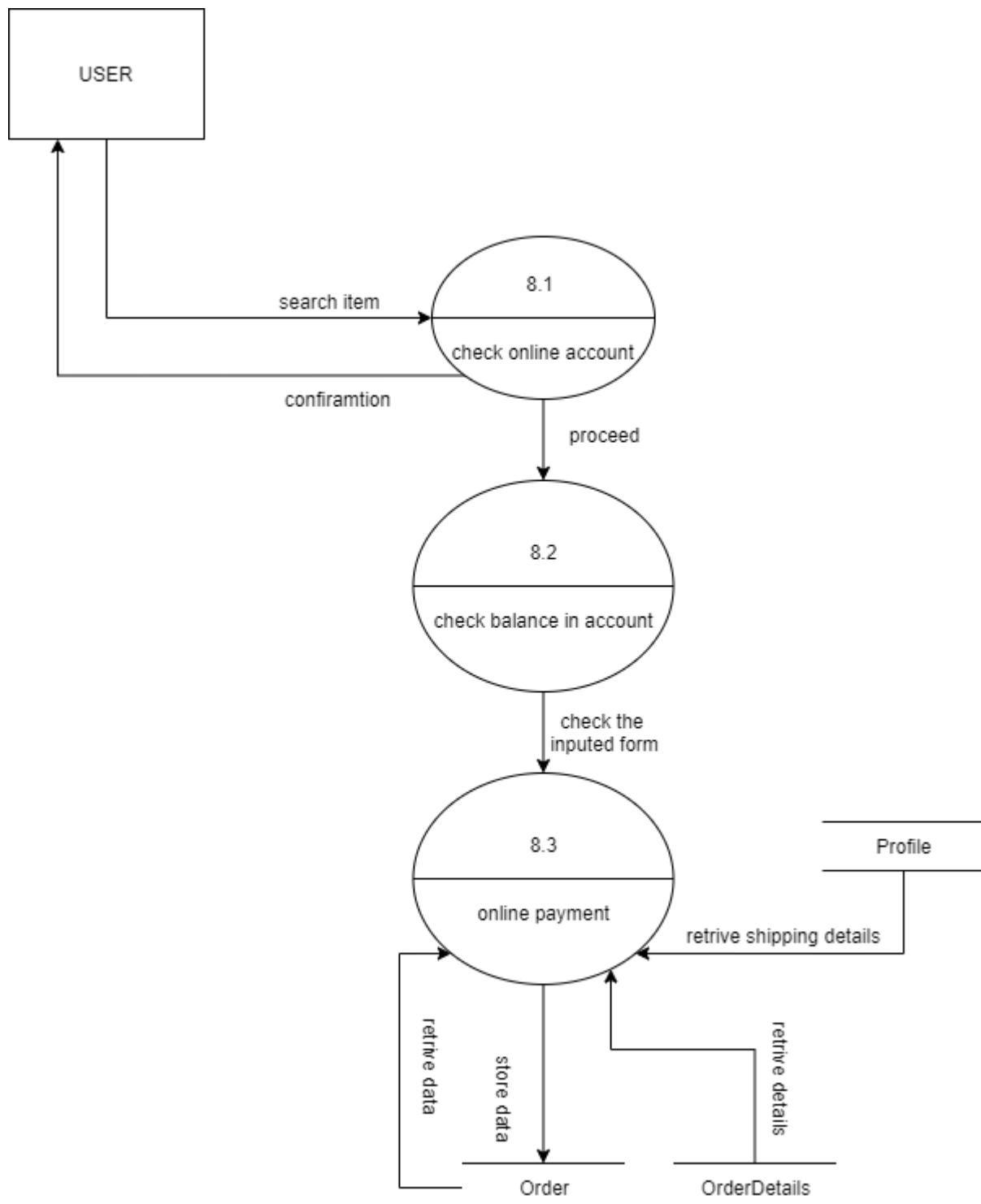


Figure 171: dfd diagram level2 of function8

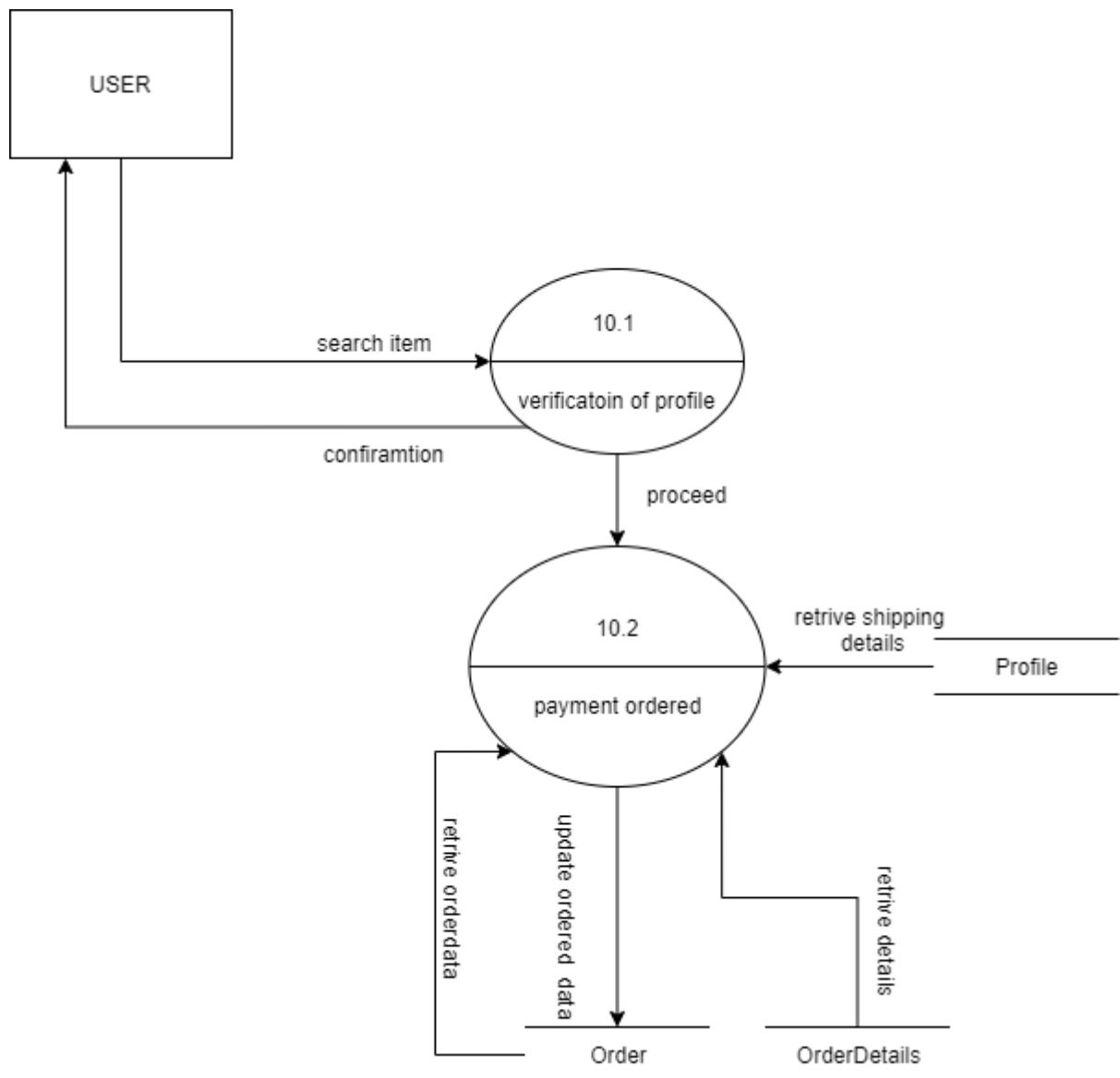


Figure 172: dfd diagram level2 of function8

8.4.7 USE CASE

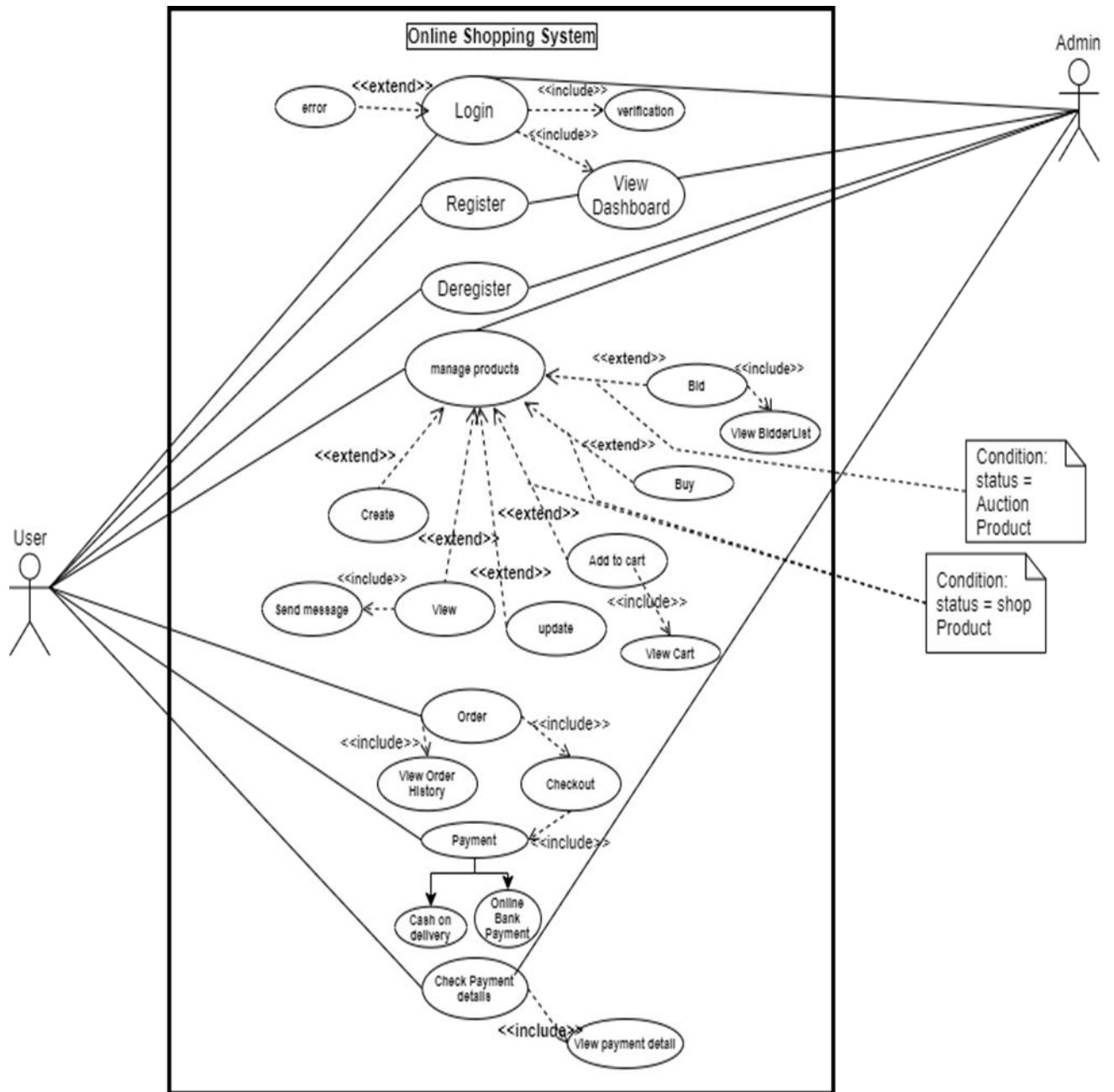


Figure 173: Use Case Diagram

8.4.8 WIREFRAME

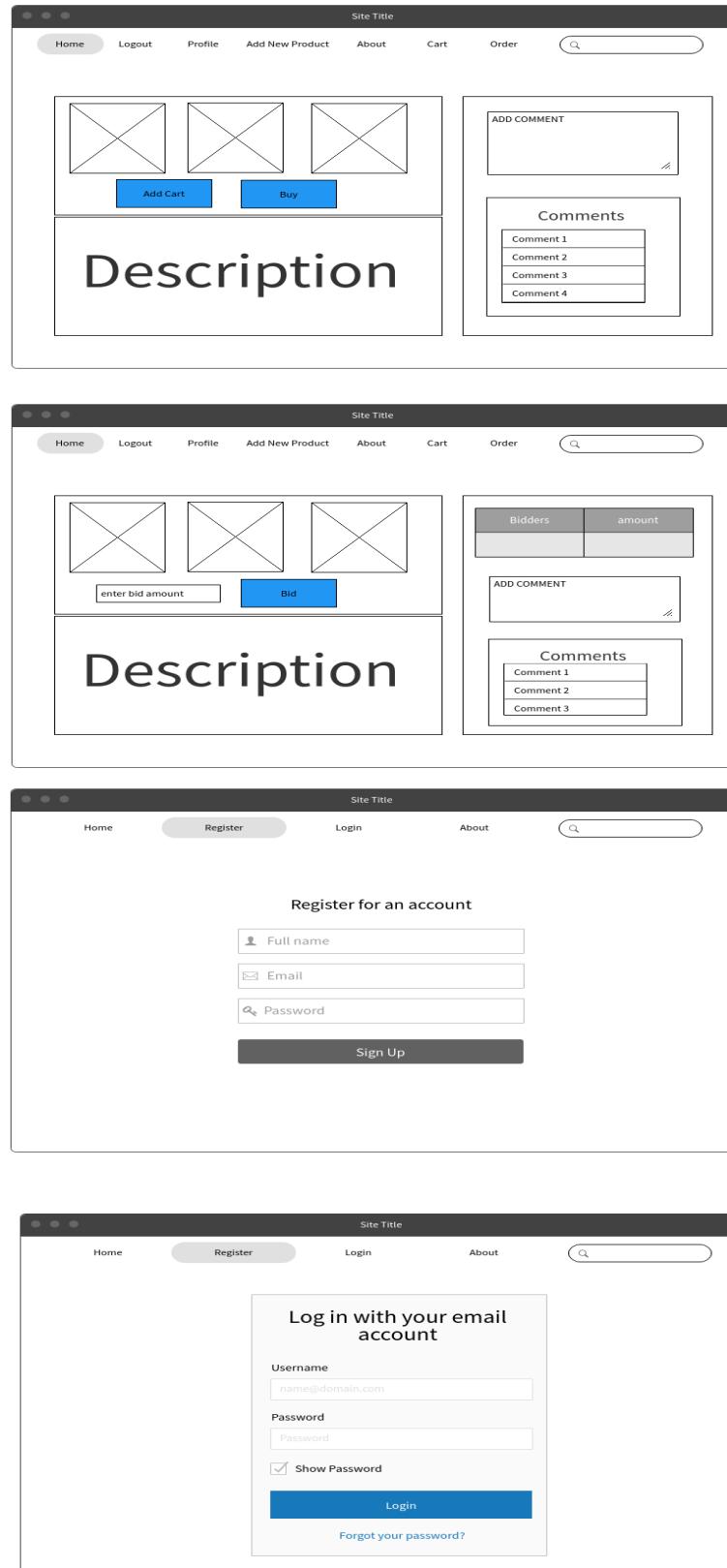


Figure 174:Wireframe part I

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

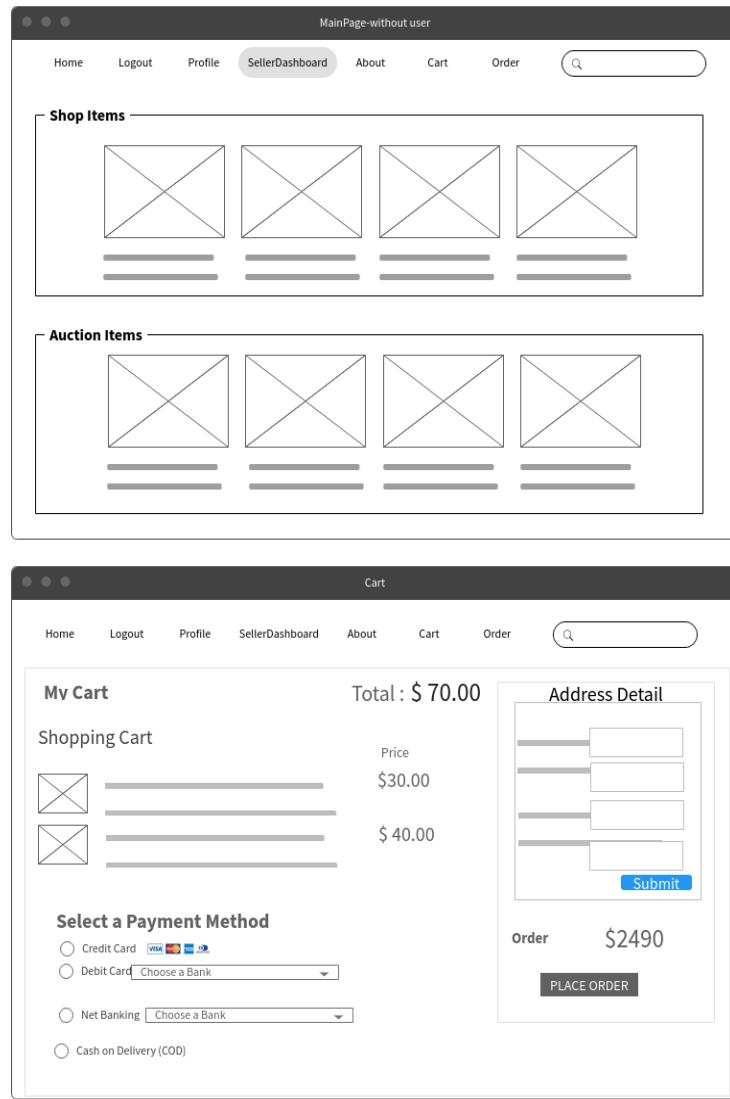


Figure 175: Wireframe part 2

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

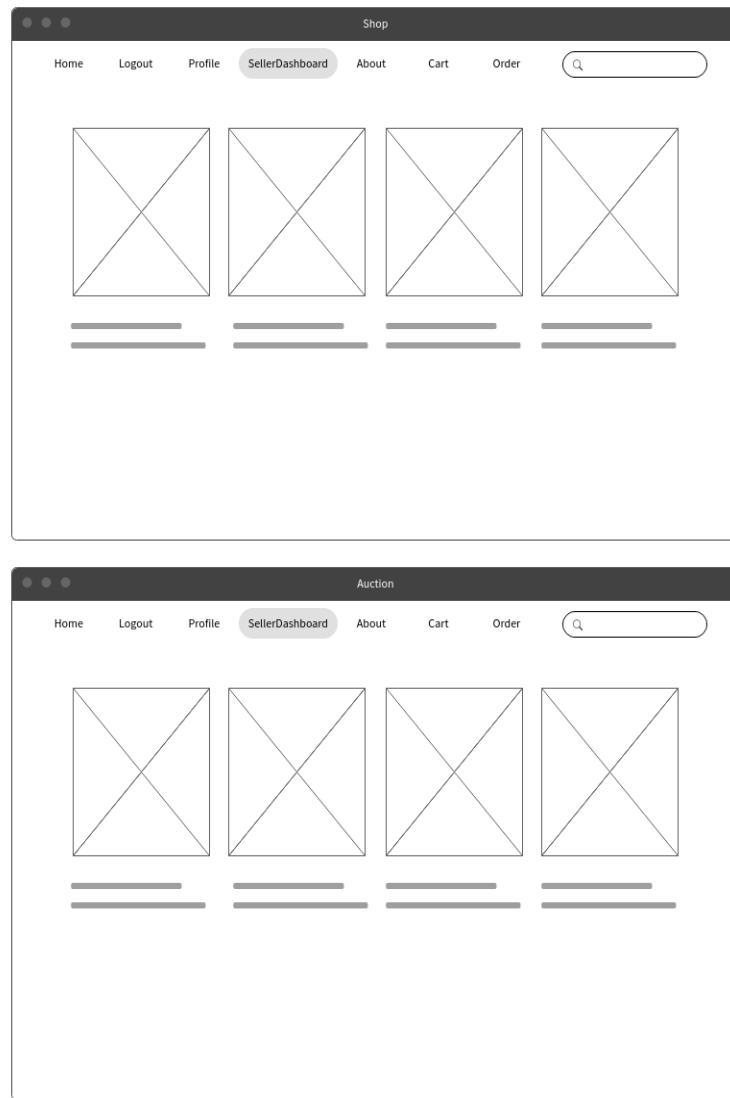
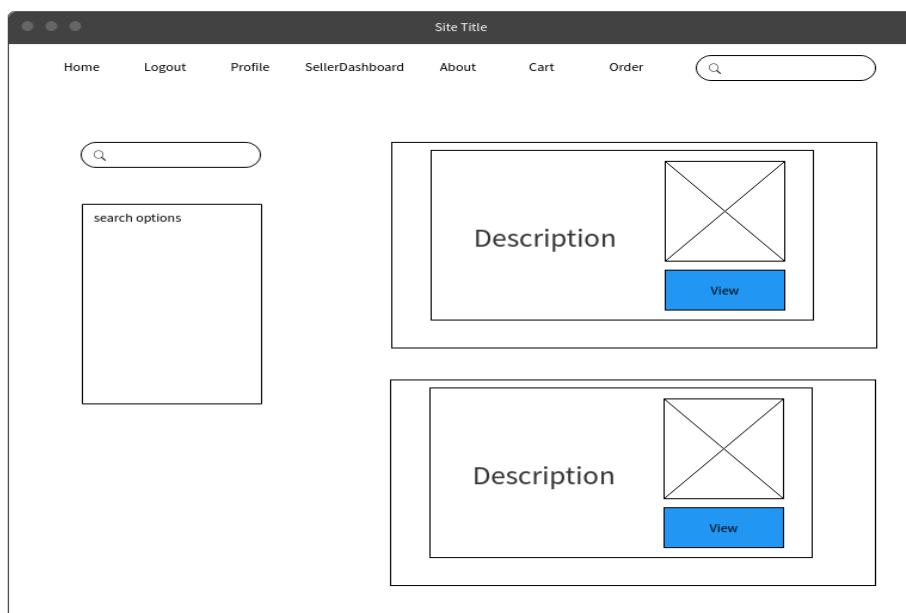
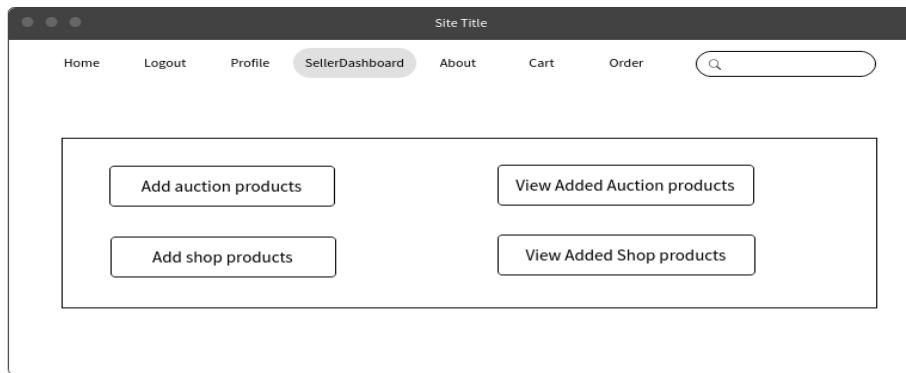


Figure 176: Wireframe part 3

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION



This wireframe shows an order summary page. At the top, there is a navigation bar with links for Home, Logout, Profile, SellerDashboard, About, Cart, Order (which is highlighted), and a search bar. Below the navigation, the title "Ordered Products" is displayed. A table summarizes the items in the order:

Product	Quantity	Price
	1	\$ 88.99
	2	\$ 50.00

Figure 177: Wireframe part 4

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The wireframes illustrate the Seller Dashboard interface for managing products across different categories.

Bidding Won Products:

Product	Quantity	Price
Men's bag \$ 88.99	1	\$ 88.99
Bed \$25.00	1	\$ 50.00

Added shop Products:

Product	Quantity	Price
Men's bag \$ 88.99	1	\$ 88.99
Sun glass \$25.00	1	\$ 50.00

Added Auction Products:

Product	Winner	Quantity	Price
Men's bag \$ 88.99	None	1	\$ 88.99
Sun glass \$25.00	User2	1	\$ 50.00

Figure 178: Wireframe part5

8.5 APPENDIX E: SCREENSHOTS OF THE SYSTEM

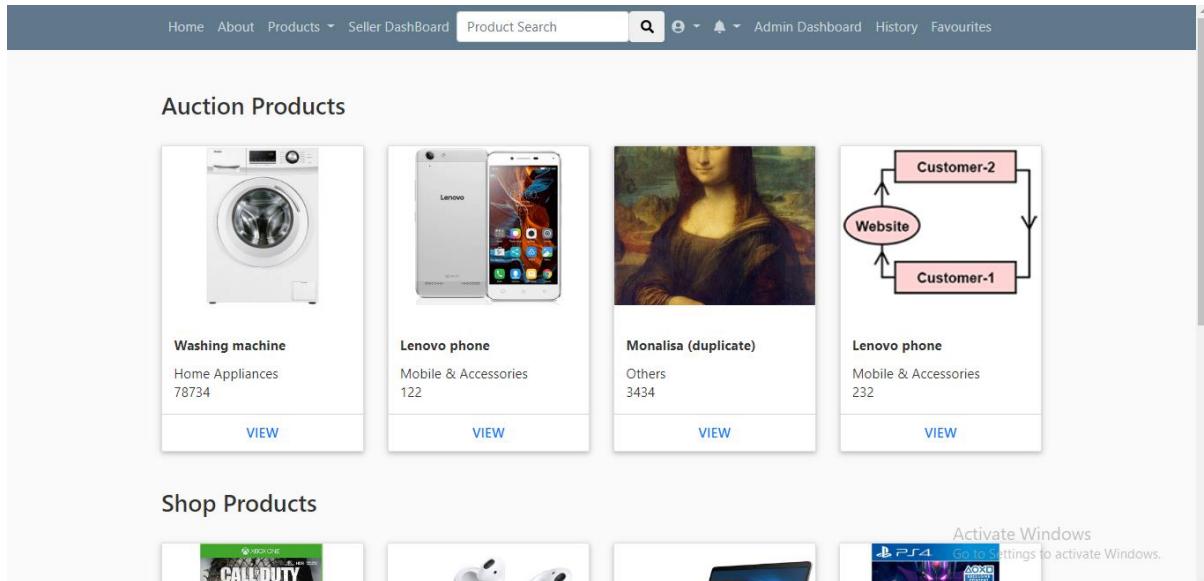


Figure 179: home page

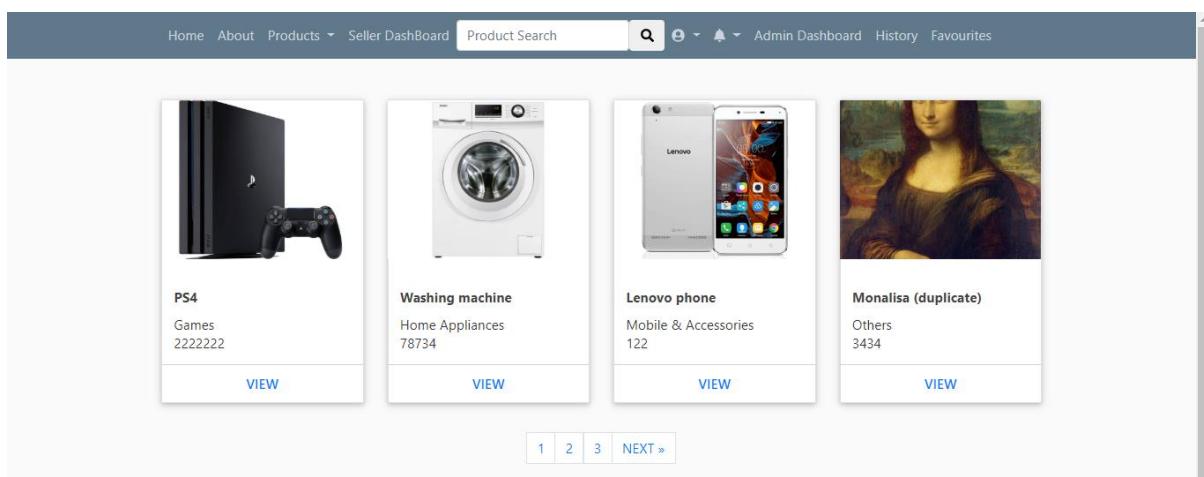


Figure 180:Auction page

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

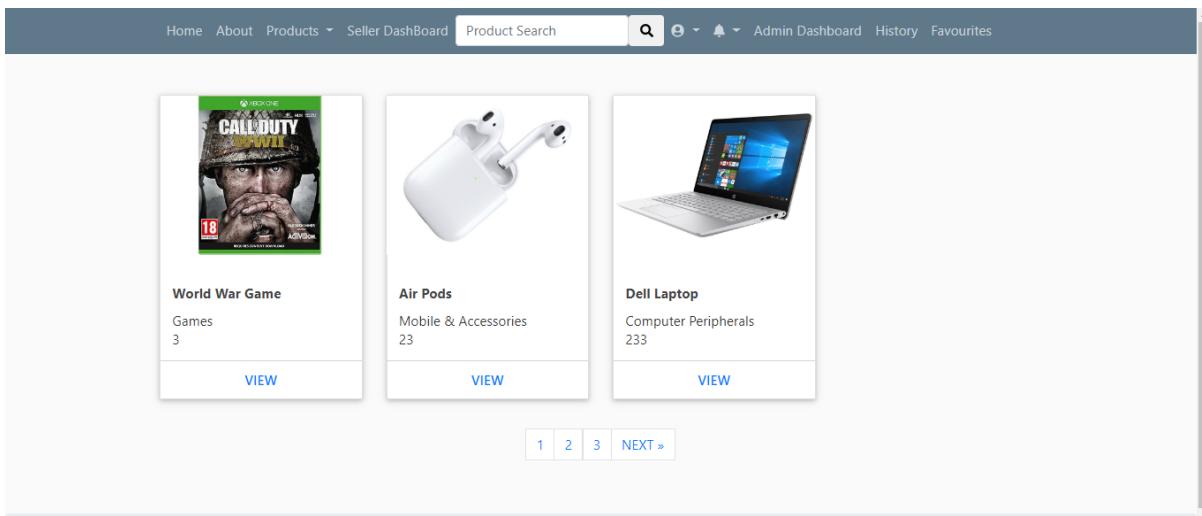


Figure 181: Shop page

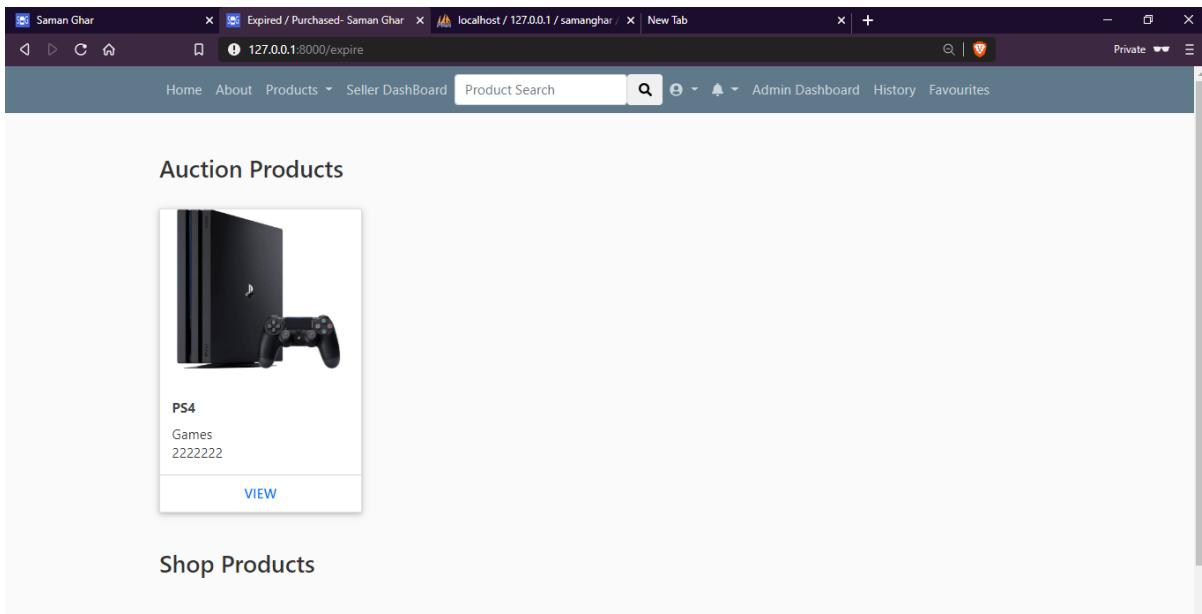


Figure 182: Expired/purchased

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

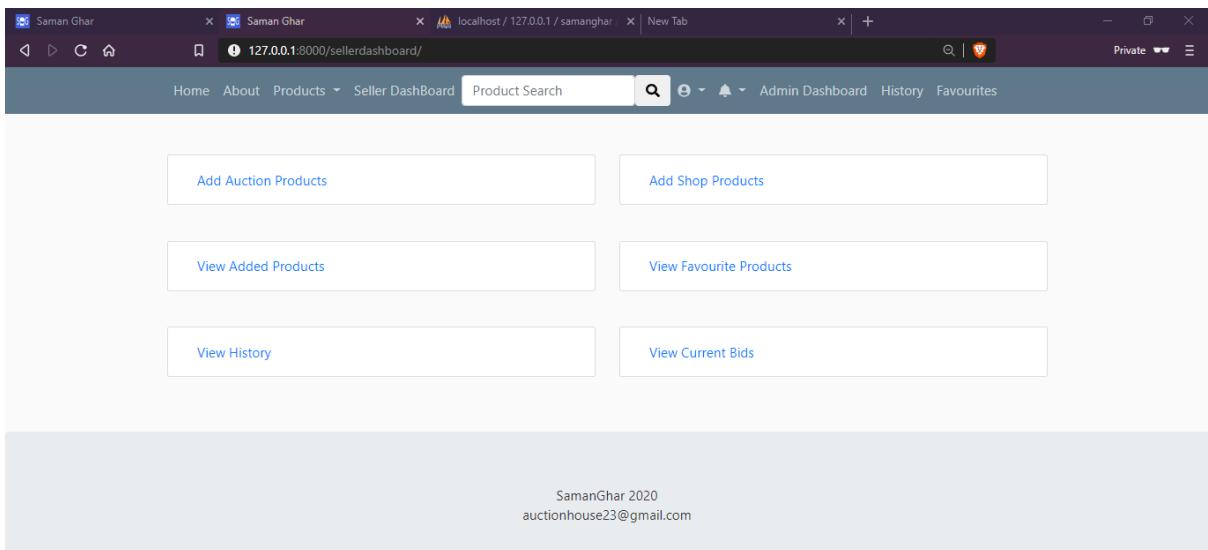


Figure 183: Seller dashboard

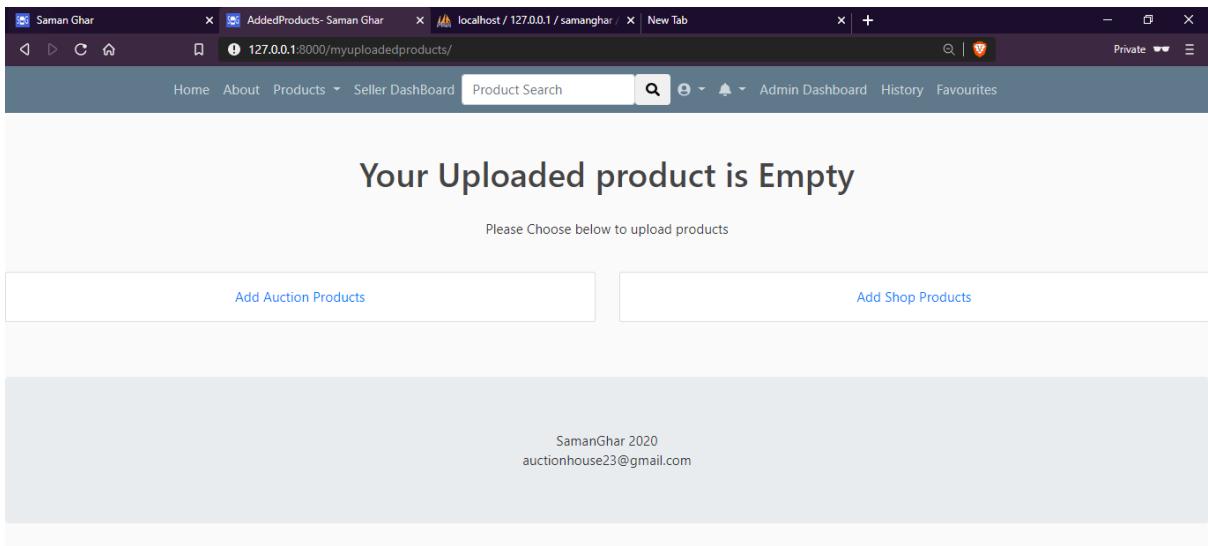


Figure 184: MY uploaded product

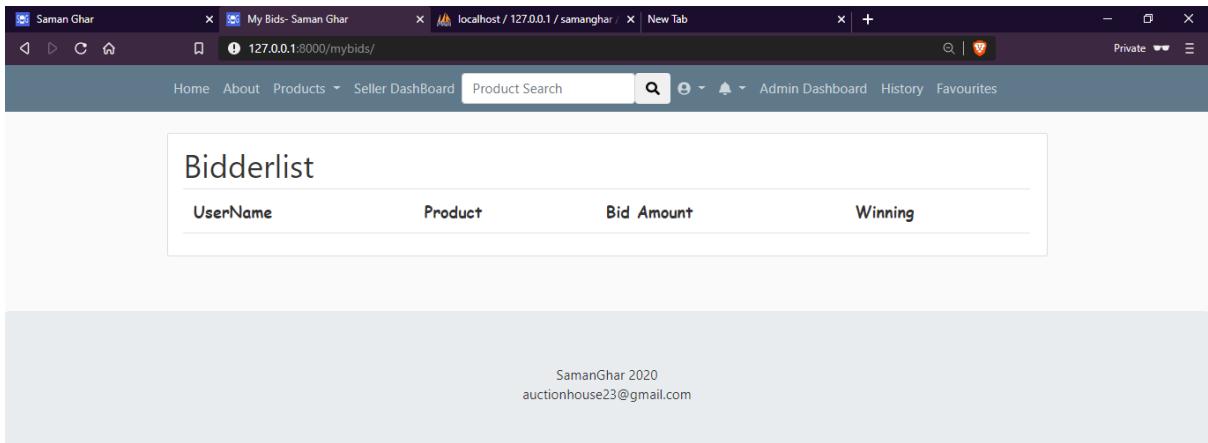


Figure 185: MY bids page

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows a web browser window with four tabs open. The active tab is titled 'History- Saman Ghar' and displays a table of viewed products. The table has columns for 'Product', 'Time', and 'View'. The products listed are Bed, PS4, Lenovo phone, and Washing machine, all viewed on June 5, 2020, at various times between 10:02 a.m. and 10:37 a.m. A footer at the bottom of the page displays the text 'SamanGhar 2020' and 'auctionhouse23@gmail.com'.

Product	Time	View
Bed	June 5, 2020, 10:02 a.m.	View
PS4	June 5, 2020, 10:21 a.m.	View
Lenovo phone	June 5, 2020, 10:15 a.m.	View
Washing machine	June 5, 2020, 10:37 a.m.	View

Figure 186:History page

The screenshot shows a web browser window with four tabs open. The active tab is titled 'Favourited Products- Saman Ghar' and displays a table of favourited products. The table has columns for 'Product', 'Time', and 'View'. The product listed is a Washing machine, favourited on June 5, 2020, at 10:39 a.m. A footer at the bottom of the page displays the text 'SamanGhar 2020' and 'auctionhouse23@gmail.com'.

Product	Time	View
Washing machine	June 5, 2020, 10:39 a.m.	View

Figure 187: favourites

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

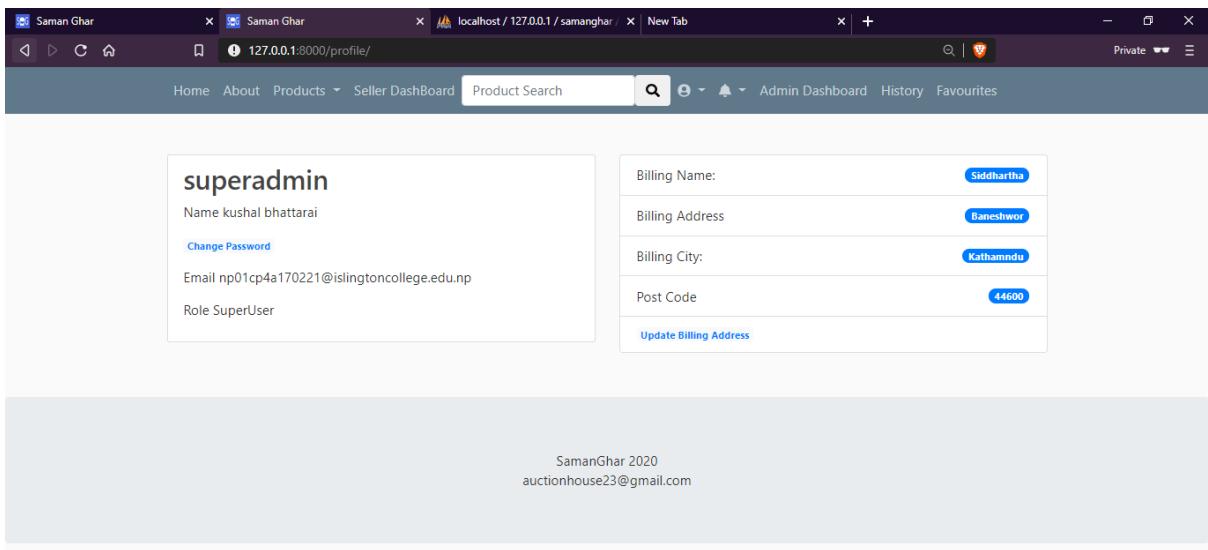


Figure 188: profile

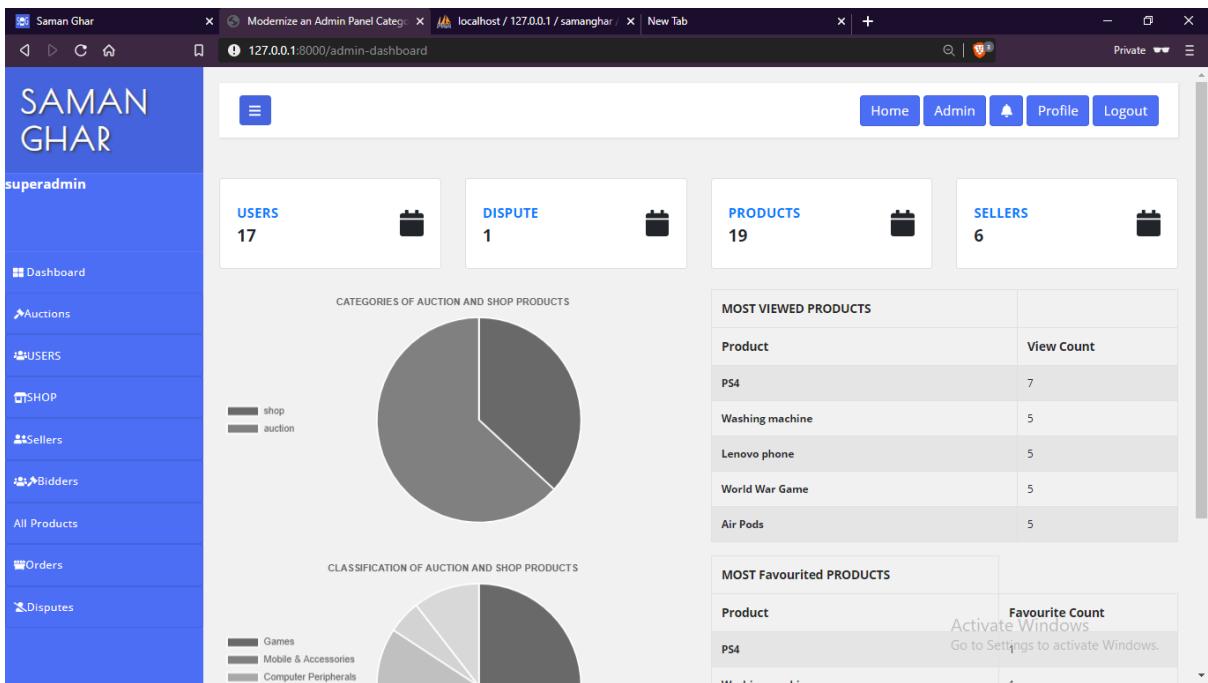


Figure 189:Admin dashboard

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

The screenshot shows the 'SG | Admin' dashboard. At the top, there's a header bar with tabs for 'WELCOME TO SAMANGHAR SUPER' and 'localhost / 127.0.0.1 / samanghar'. Below the header, a navigation bar includes links for 'Private', 'WELCOME KUSHAL', 'VIEW SITE / VIEW DASHBOARD', 'CHANGE PASSWORD', and 'LOG OUT'. The main content area is titled 'WELCOME TO SAMANGHAR SUPER ADMIN PANEL'. It features three main sections: 'AUTHENTICATION AND AUTHORIZATION' (Groups, Users), 'USERS' (Conversations, Disputes, Favourites, History, Notifications, Profiles), and 'WEBSITE' (Bidders, Carts, Order items, Orders, Products, Sellers). Each section has 'Add' and 'Change' buttons. To the right, there's a sidebar titled 'Recent actions' showing a list of recent product and order actions, and a message 'Activate Windows' with a link to Settings.

Figure 190: Super admin dash board

The screenshot shows a search results page for 'bed'. The URL is '127.0.0.1:8000/search/?q=bed&data=product'. The top navigation bar includes 'Home', 'About', 'Products', 'Seller Dashboard', 'Product Search' (with a search icon), 'Admin Dashboard', 'History', and 'Favourites'. Below the navigation, there's a search bar with dropdowns for 'Search Category' (set to 'Games') and 'Search' (set to 'bed'). The main content area displays a product card for a bed, featuring an image of a wooden bed with blue bedding, the text 'Bed' and '23333', and a 'VIEW' button.

Figure 191:Search product

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

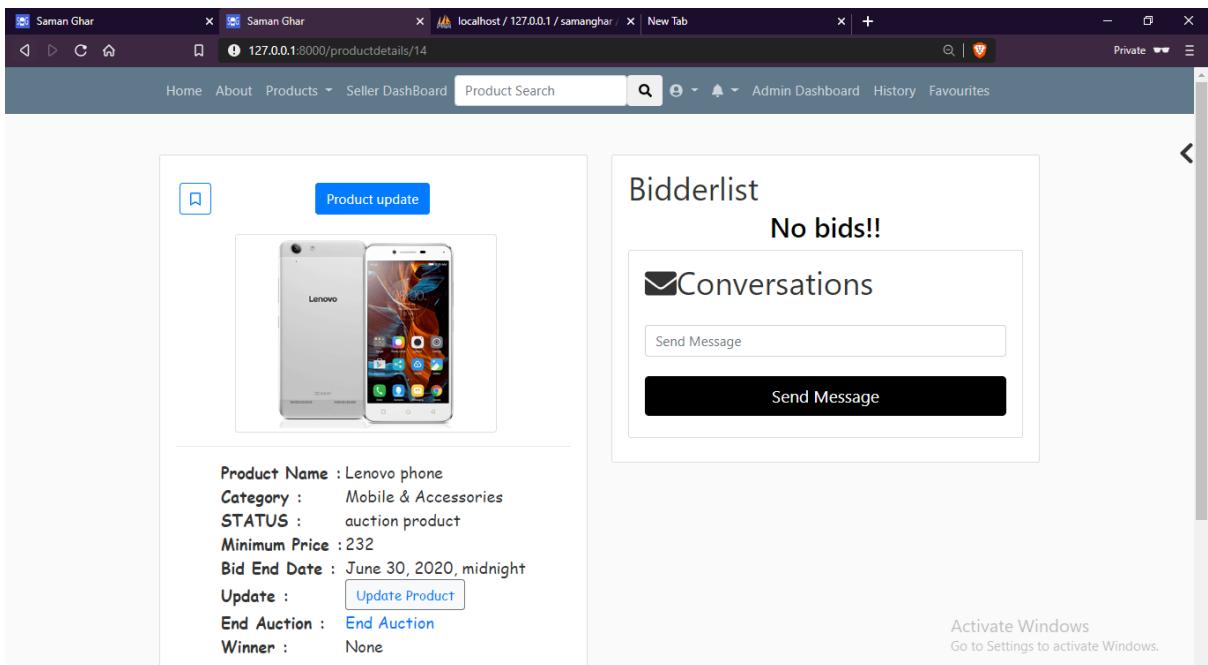


Figure 192:product deatils

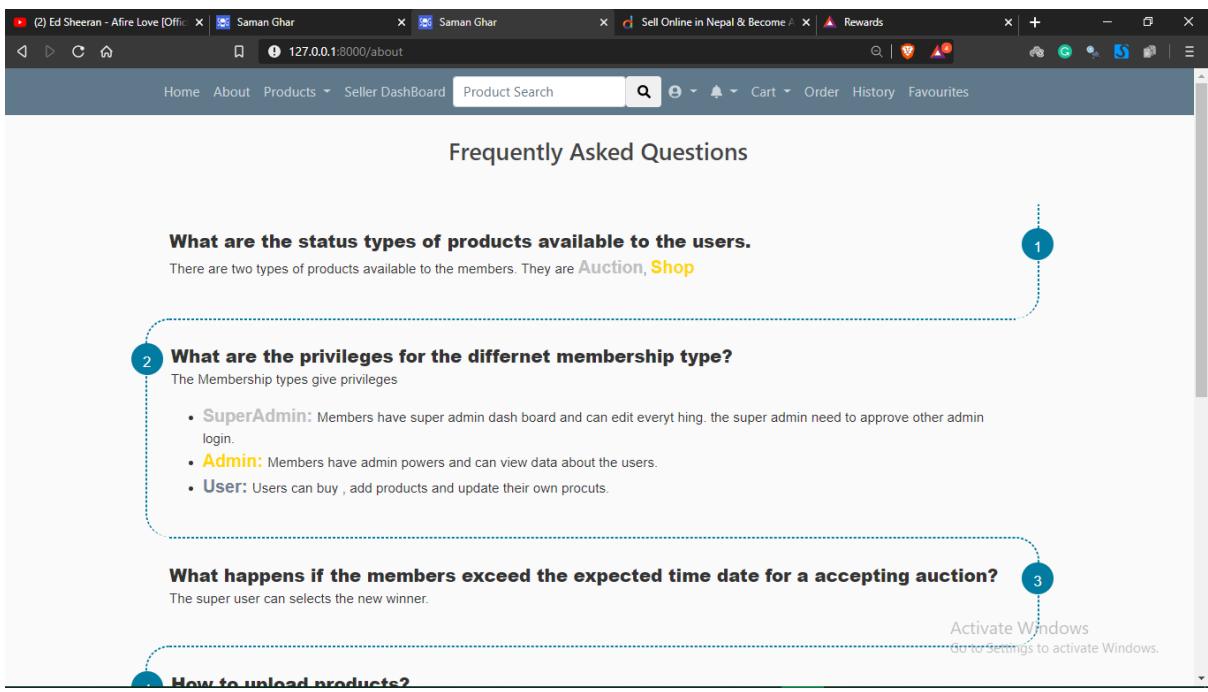


Figure 193: About page

8.6 APPENDIX F: USER FEEDBACK

8.6.1 USER FEEDBACK FORM

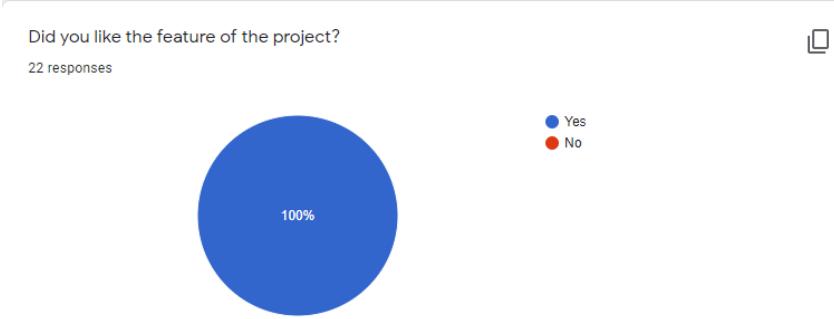


Figure 194: user feedback result -1

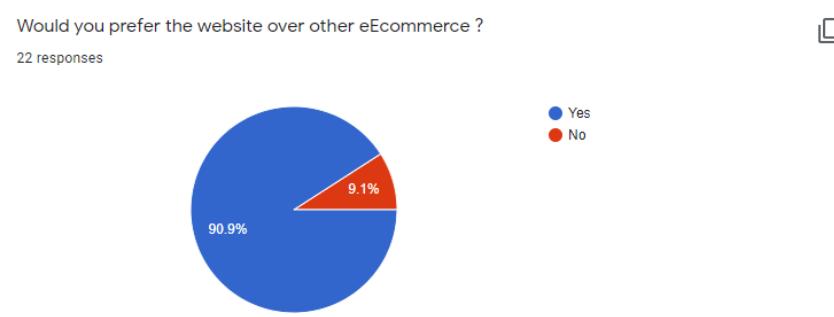


Figure 195: user feedback result -2

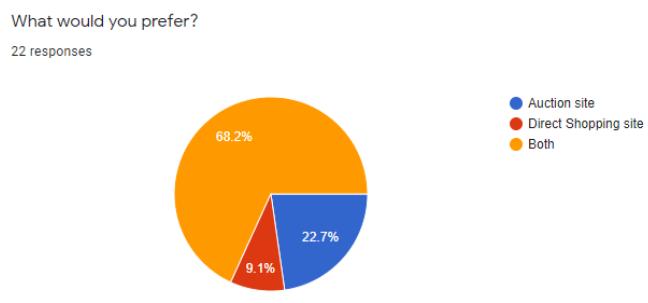


Figure 196: user feedback result -3

SAMANGHAR – AN AUCTION AND SHOP BASED ECOMMERCE SOLUTION

Did you like the bidding feature in the website ?

22 responses

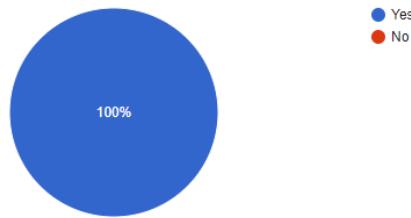


Figure 197: user feedback result - 4

How easy was it to access the content of the website?

22 responses

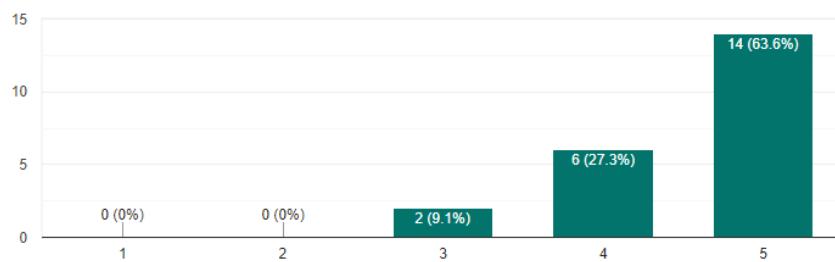


Figure 198: user feedback result -5

Could you easily upload your products in the website?

22 responses

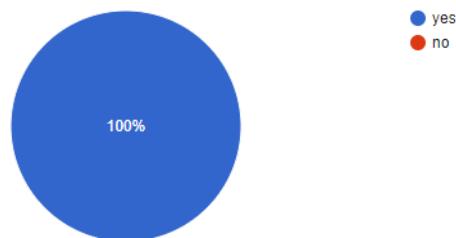


Figure 199: user feedback result 6

8.6.2 SAMPLE OF FILLED USER FEEDBACK FORMS

22 responses

Accepting responses

Summary Question Individual

joshibinayak17@gmail.com ▾ 4 of 22

Responses cannot be edited

Customer Feedback

We would love to hear your thoughts or feedback on how we can improve your experience!

*Required

This screenshot shows a Google Forms survey titled "Customer Feedback". At the top, it displays "22 responses" and a toggle switch for "Accepting responses" which is turned on. Below this are three tabs: "Summary", "Question", and "Individual", with "Individual" being the active tab. A dropdown menu shows the email "joshibinayak17@gmail.com". The main content area has a green header bar with the text "Responses cannot be edited". The first section is titled "Customer Feedback" with the sub-instruction "We would love to hear your thoughts or feedback on how we can improve your experience!". It includes a note "Required" with a red asterisk. The survey consists of two main questions, each with a list of options. The first question asks "Did you like the feature of the project? *". The second option "Yes" is selected with a green radio button. The second question asks "Would you prefer the website over other eCommerce ? *". The first option "Yes" is selected with a green radio button.

Figure 200: sample -1

Did you like the feature of the project? *

Yes
 No

Would you prefer the website over other eCommerce ? *

Yes
 No

This screenshot shows a Google Forms survey titled "Customer Feedback" with 22 responses. The survey contains two questions. The first question is "Did you like the feature of the project? *". The second option "Yes" is selected with a green radio button. The second question is "Would you prefer the website over other eCommerce ? *". The first option "Yes" is selected with a green radio button.

Figure 201: sample -2

Did you like the bidding feature in the website? *

Yes
 No

How easy was it to access the content of the website? *

1 2 3 4 5

hard easy

Figure 202: sample3

Could you easily upload your products in the website? *

yes
 no

Figure 203: sample 4

8.7: APPENDIX G: FUTURE WORK

8.7.1 READINGS FOR FUTURE WORK

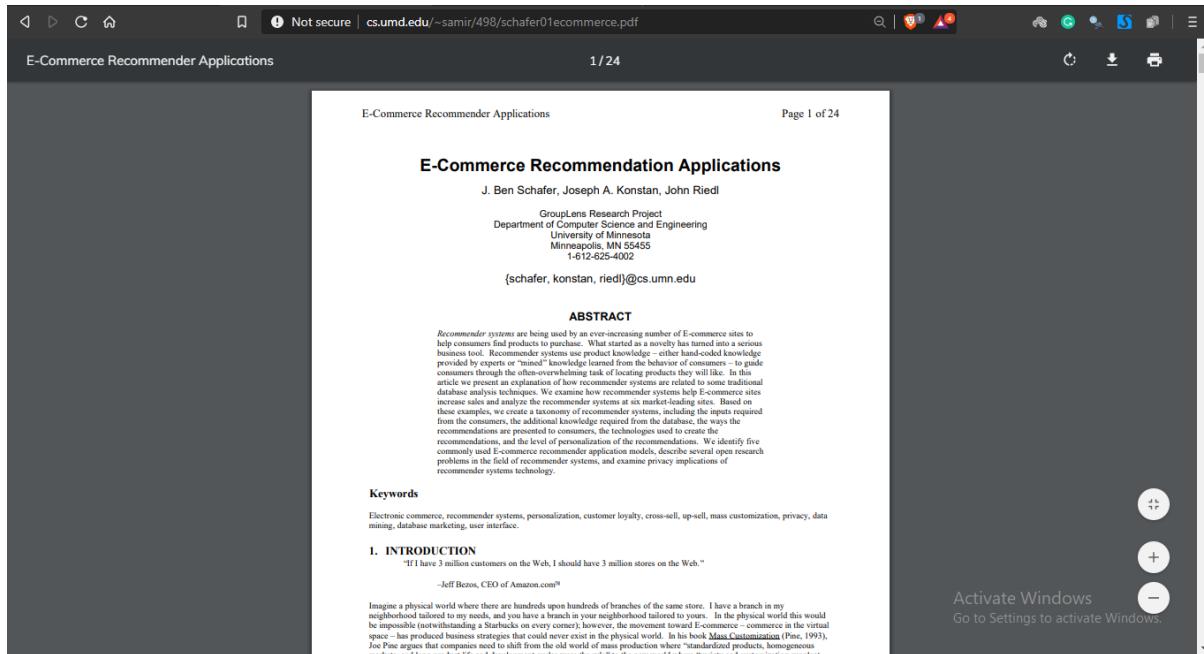


Figure 204: recommendation further work sample

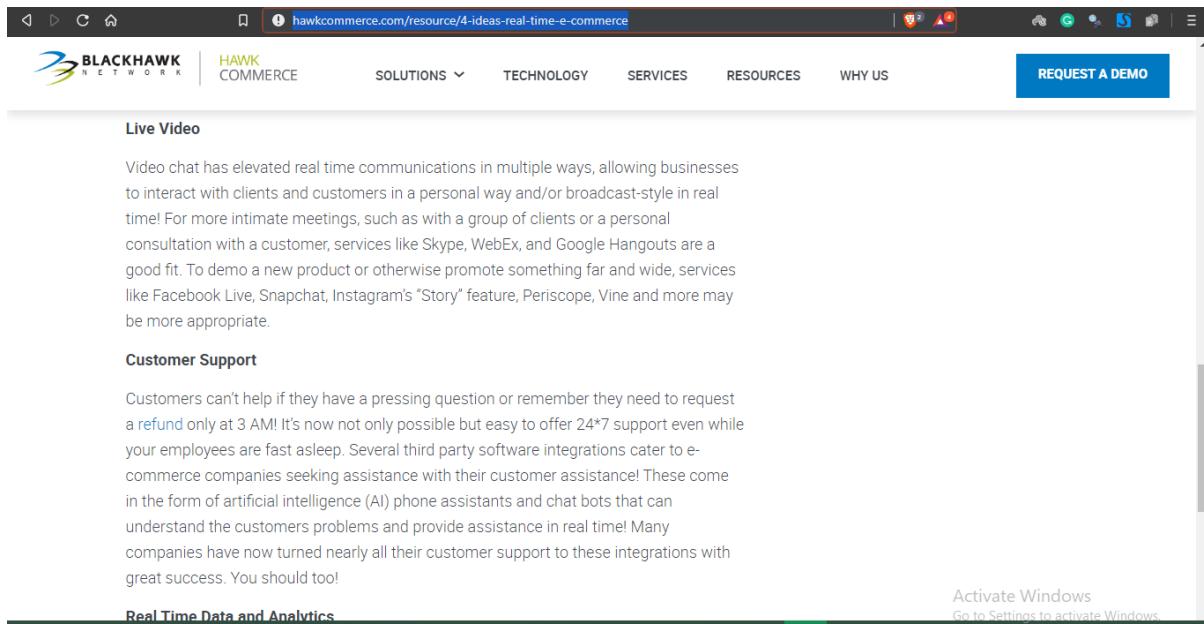


Figure 205:communication with buyer and supplier