

The
Project Report
on
OLXClone

By

Babariya Bhavin B. (CE-007) (19CEUEF014)

Bhalala Jenil D. (CE-010) (19CEUET035)

Domadiya Fenil A. (CE-029) (19CEUEG037)

B.Tech CE Semester-IV

Subject: Software Project

Under the guidance of,

Prof. Brijesh S. Bhatt

Prof. Pinkal C. Chauhan

Prof. Jigar M. Pandya



Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

Table of Contents

1 Introduction...	3
1.1 Brief Introduction	3
1.2 Technology and Tools Used	4
2 Software Requirement Specifications	5
2.1 System Functional Requirements	5
2.2 Other Non-Functional Requirements.....	9
3 Design... ..	10
3.1 Use Case Diagram.....	10
3.2 Class Diagram.....	11
3.3 Sequence Diagram	12
3.4 Activity Diagram	14
3.5 Data Flow Diagram.....	16
3.6 Structure Chart	19
3.7 Data Dictionary	20
4 Implementation Details	22
4.1 Modules.....	22
4.2 Functional prototypes	23
5 Testing	25
6 Screen-shots of the System.....	29
7 Conclusion.....	32
8 Limitations and Future Extensions of System	33
9 Bibliography	34

1. Introduction

1.1 Brief Introduction

“OLXClone” is useful for online exchange of the used things like if one user of the system wants to sell used products then user has to simply put all necessary informations of that product on this platform and if some another user of the system want to buy that product then he has to simply contact to the seller of that product.

So, In this way every seller can able to sell their products very easily. And every buyer gets good products of his choice. In short, this system provides medium where user can sell their used products and buyer can easily buy used products in a very short time period.

1.2 Tools/Technologies Used

Technologies:

- Django
- Python
- MySQL
- Bootstrap
- HTML
- CSS
- JavaScript

Tools:

- Git
- Visual Studio Code

Platform:

- Local development server

2. Software Requirement Specifications

2.1 System Functional Requirements

R1: Manage User's Account:

Description: If a person is using this app first time, then he will have to register himself at first using Sign-Up functionality. And then he will be able to Sign-in this app.

R.1.1: Click on the “Sign-Up” Option:

Input: User details including name of the user, password, address, mobile number, email-id etc.

Output: Confirmation of registration status.

R.1.2: Click on the “Sign-in” Option:

Input: User credentials (“Username” and “Password”).

Output: User will be redirected to the appropriate page of the application, if credentials are valid.

R.1.3: Click on the “logout” Option:

Input: Click on Option.

Output: Redirect to Login Page.

R2: Manage Selling:

R.2.1: Set category:

Input: Choose appropriate category of the Item

Output: Confirmation message.

R.2.2: Add Item details:

Description: User must have to provide an image and price of the item, how old is that item etc.

Input: An item details.

Output: Item added successfully.

R.2.3: Update Item Details:

Description: User have to give proper update item details which details he wants to update.

Input: Item details which user wants to update.

Output: Item updated successfully.

R.2.4: Delete Item:

Input: Click on the delete button.

Output: Item deleted successfully.

R3: Manage Buying:

R.3.1: Set location:

Input: User's location.

Output: Confirmation message.

R.3.2: Select Category:

Input: Select appropriate category of an item which user wants to buy.

Output: Items of selected category will be displayed.

R.3.3: Search an item:

Input: Name of the Item.

Output: Items related to entered name will be displayed.

R.3.4: Filter and Sort items:

Description: User will be able to sort the items in the order of high to low price or a vice versa. Also he will be able to filter the items of specific price range.

Input: Choose the appropriate option.

Output: Display items in appropriate order.

R4: Manage chat between seller and buyer:

Description: If a buyer is not satisfy with the price of an item which he wants to buy, then using this chat functionality they can communicate with each other.

Input: Text message

Output: Message successfully sent.

R5: Profile Modification:

Input: Change needed details.

Output: Updated details with confirmation of the changes.

R6: Give Feedback:

Input: Ratings and Feedback by user.

Output: Confirmation message.

2.2 Other Nonfunctional Requirements

1. Usability:

- The system must be easy to use by users such that they do not need to read an extensive amount of manuals.
- The options of the system must be easily navigable by the users with buttons that are easy to understand.

2. Reliability:

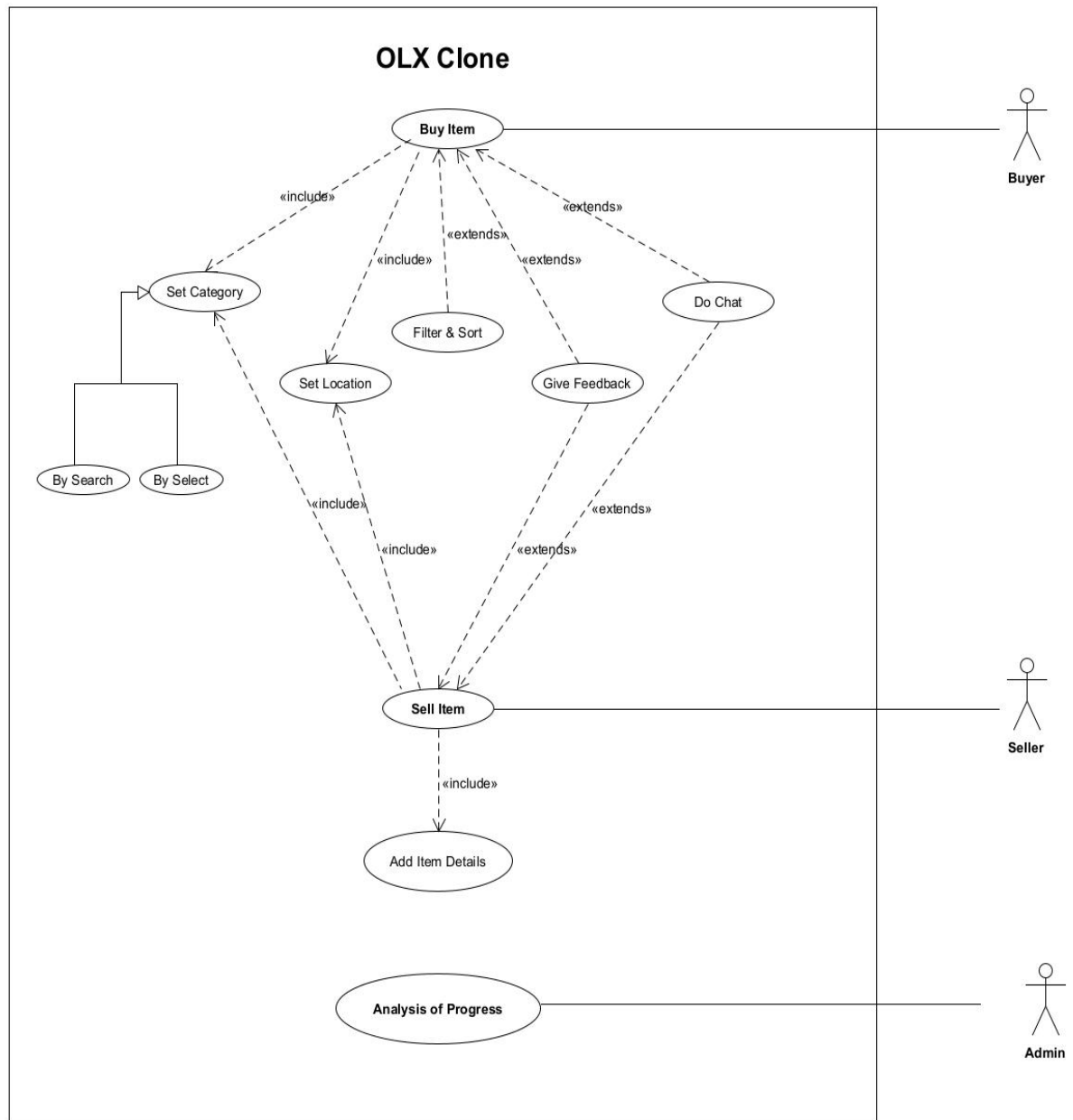
- The system should not update the data in any database for any failed processes.
- The system is able to update and delete information which is provided by system user very easily.

3. Performance:

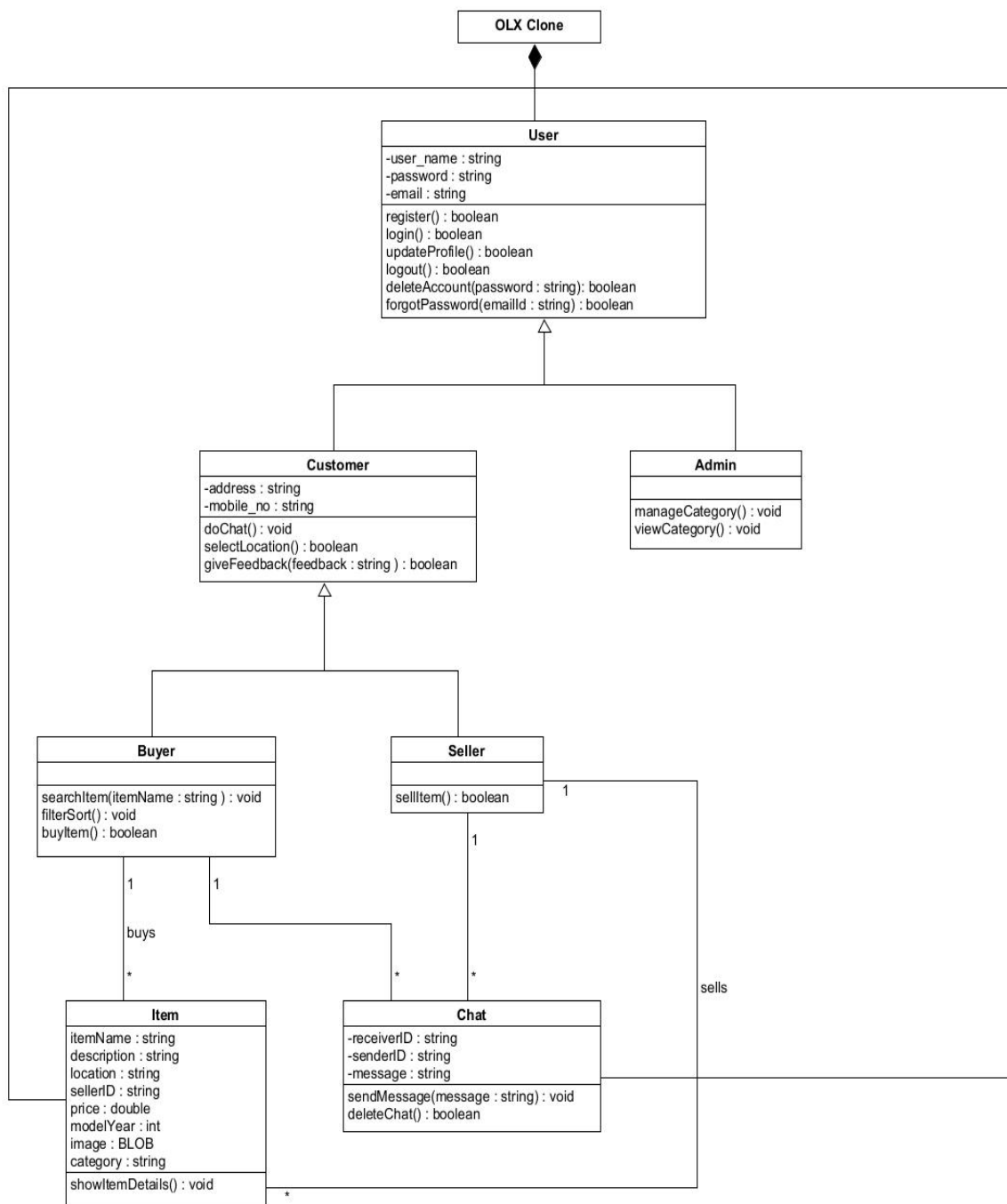
- The system must complete updating the databases options successfully every time the user requests such a process.
- All the functions of the system must be available to the user every time the system is turned on.

3. Design

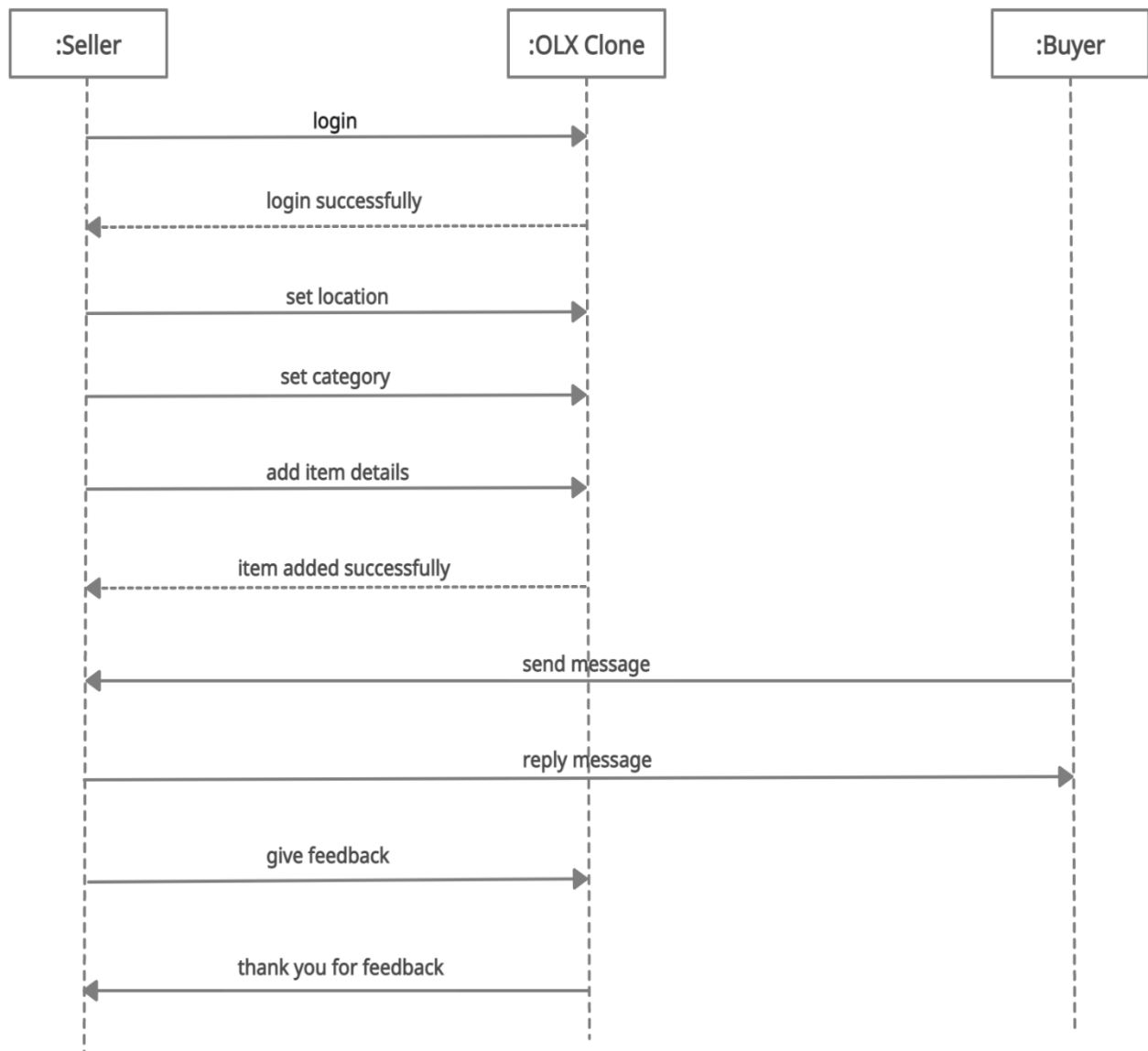
3.1 Use Case Diagram



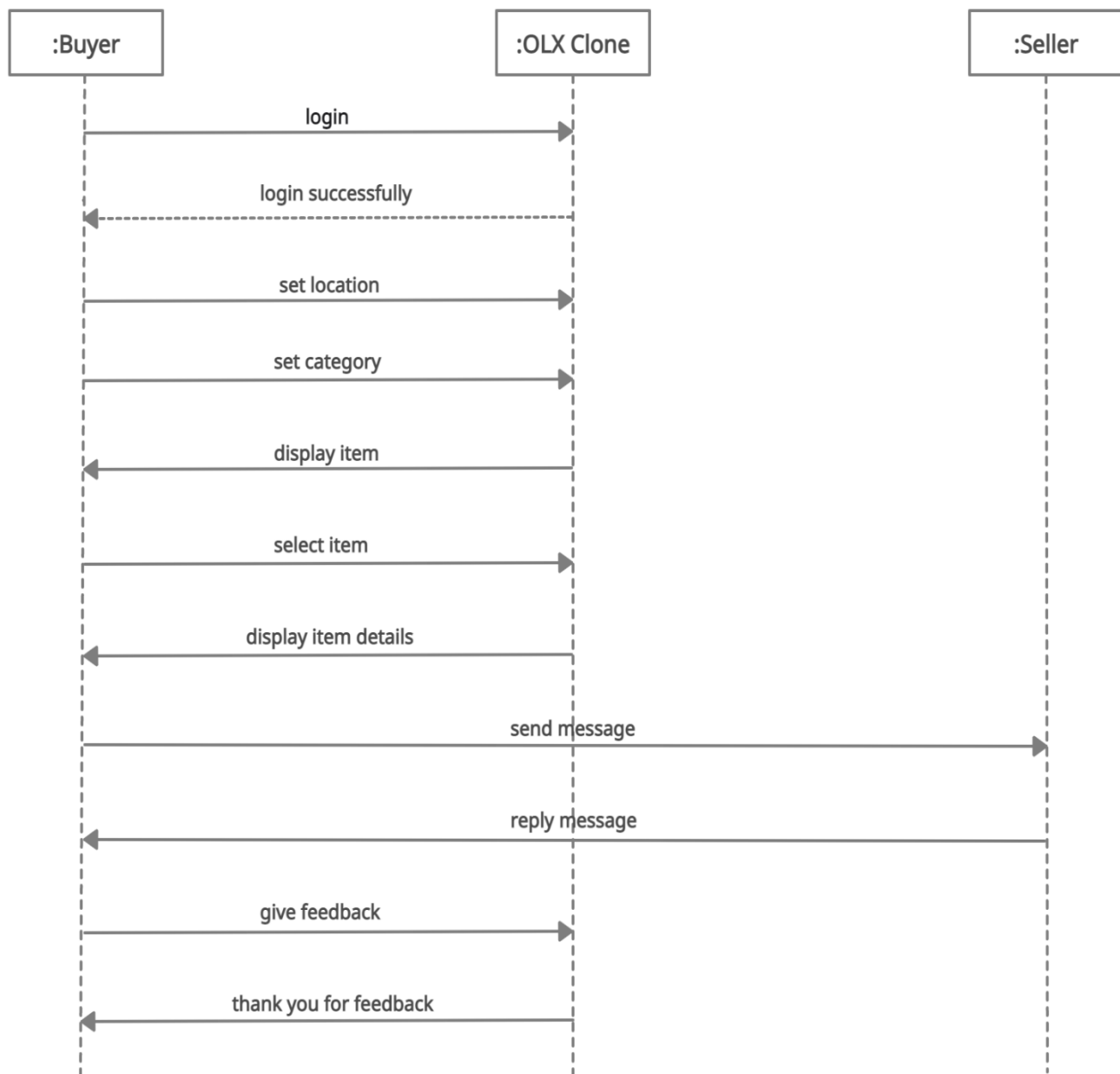
3.2 Class Diagram



3.3 Sequence Diagram



Sequence Diagram for “Sell Item”

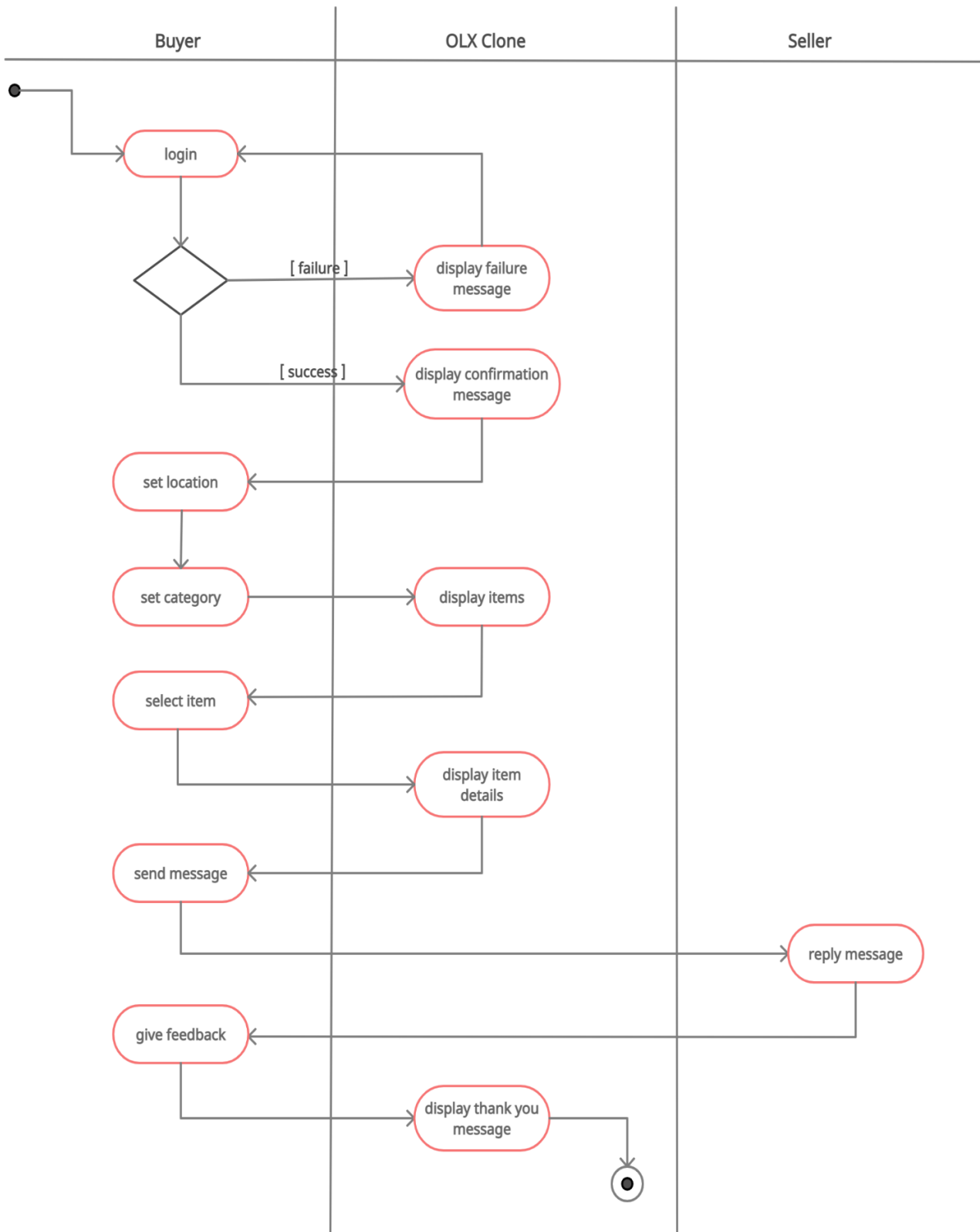


Sequence Diagram for “Buy Item”

3.4 Activity Diagram

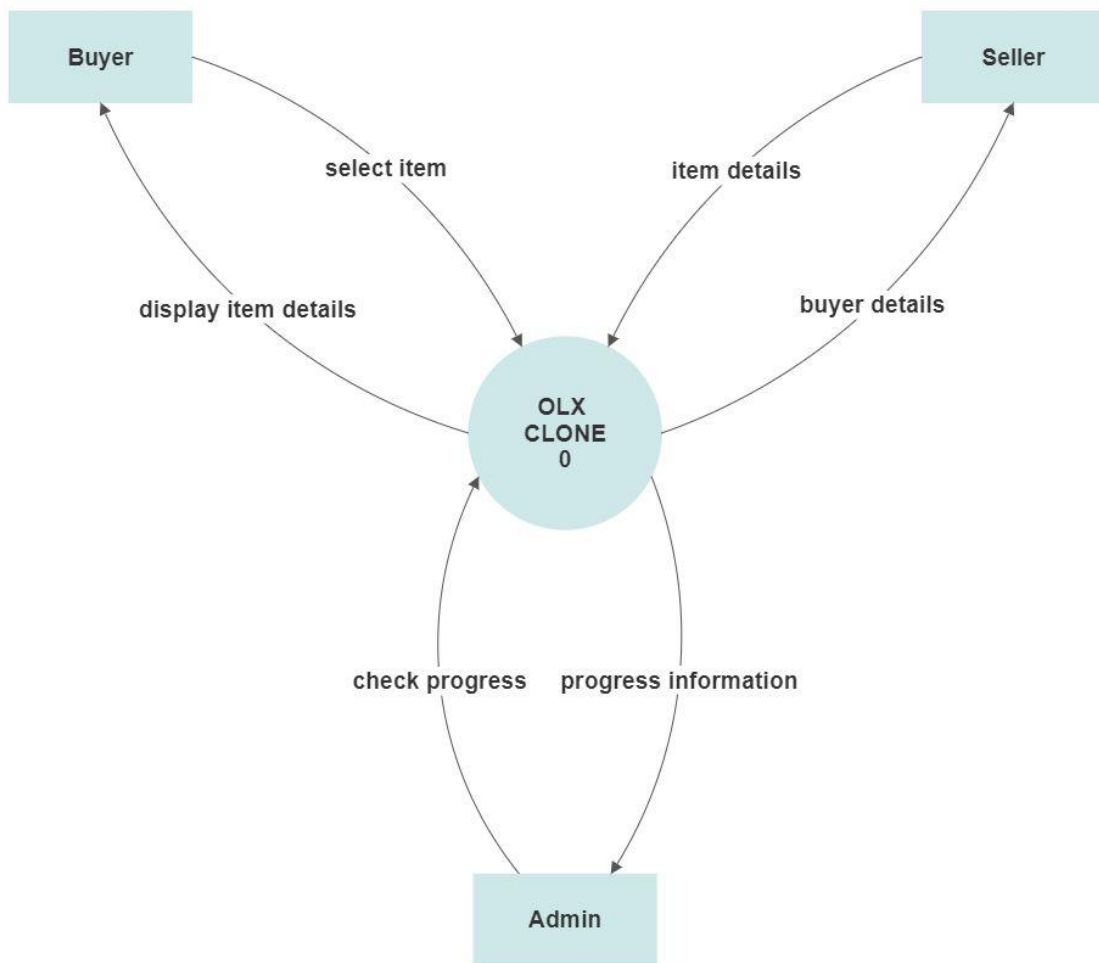


Activity Diagram for “Sell Item”

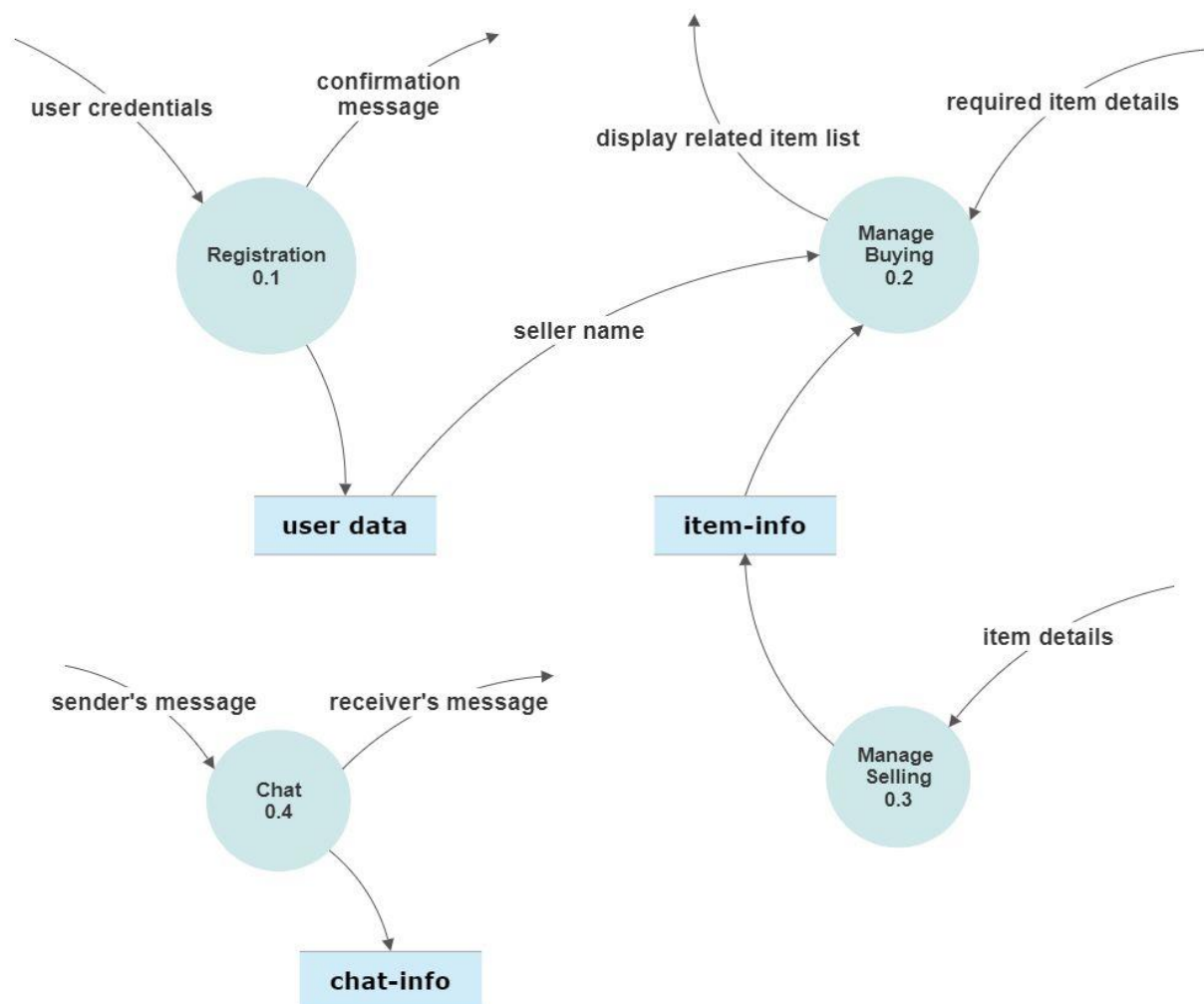


Activity Diagram for “Buy Item”

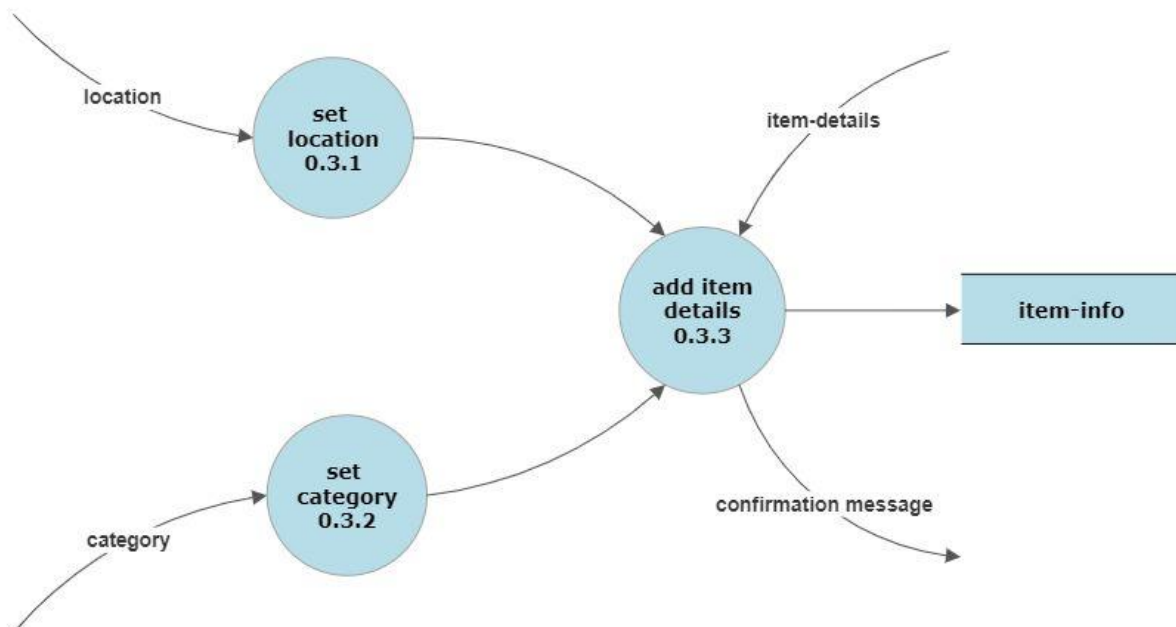
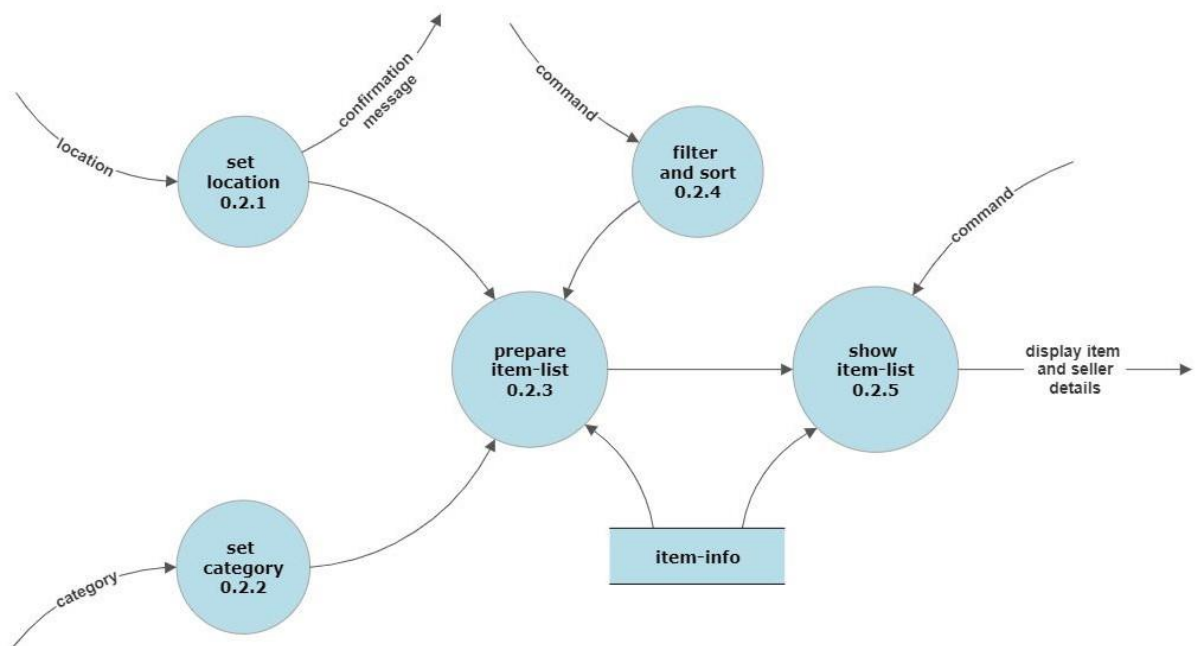
3.5 Data Flow Diagram



Level 0 : Context Diagram

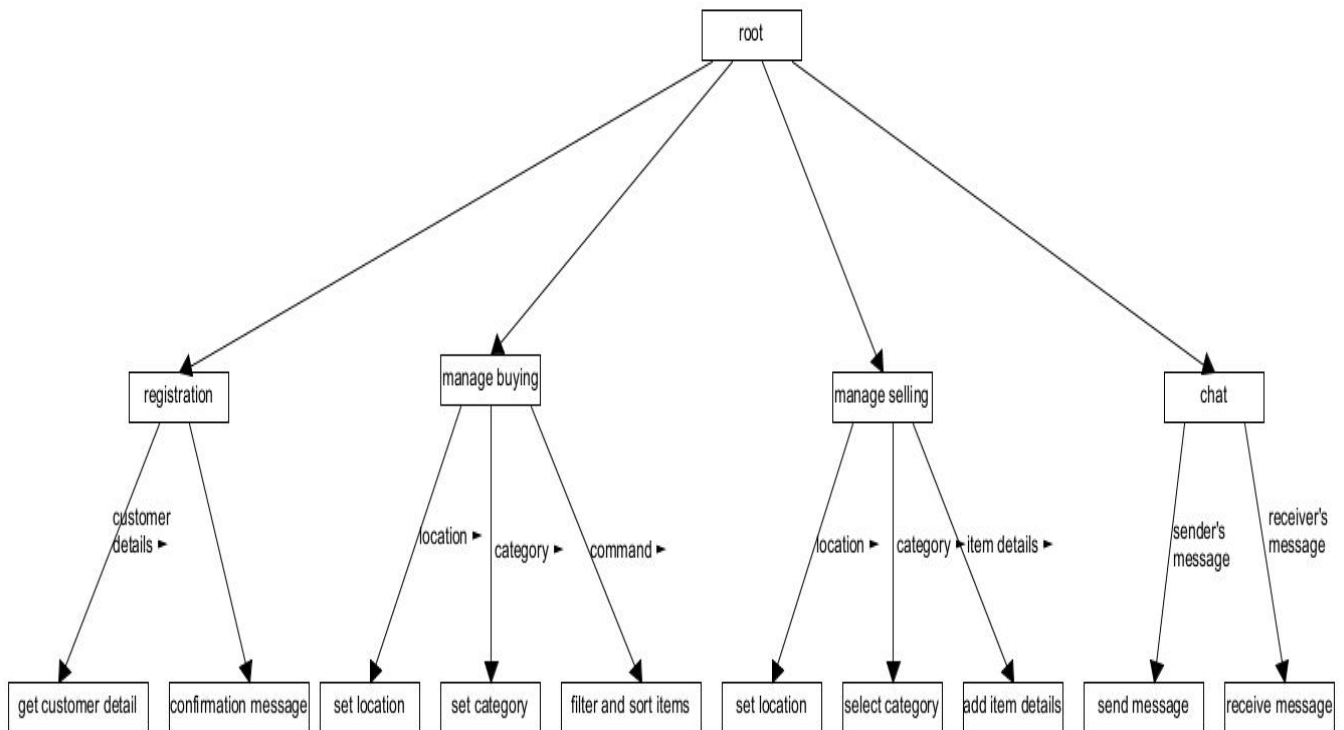


Level 1 : DFD



Level 2 : DFD

3.6 Structure Chart



3.7 Data Dictionary

User								
Sr. no.	Field Name	Data Type	Width	Required	Unique	PK/FK	Referred Table	Description
1	id	int	-	yes	yes	yes	-	Auto Increment
2	username	varchar	150	yes	yes	no	-	
3	email	varchar	254	yes	yes	no	-	
4	password	varchar	128	yes	no	no	-	
5	last_login	datetime	6	yes	no	no	-	
6	is_superuser	tinyint	1	yes	-	-	-	
7	first_name	varchar	150	yes	no	no	-	
8	last_name	varchar	150	yes	no	no	-	
9	date_joined	datetime	6	yes	no	no	-	
10	is_active	tinyint	1	yes	no	no	-	

Contact								
Sr. no.	Field Name	Data Type	Width	Required	Unique	PK/FK	Referred Table	Description
1	sno	int	-	yes	yes	yes	-	Auto Increment
2	name	varchar	255	yes	yes	no	-	
3	phone	varchar	13	yes	yes	no	-	
4	email	varchar	100	yes	no	no	-	
5	context	longtext	-	yes	no	no	-	
6	timeStamp	datetime	6	yes	-	-	-	

Item								
Sr. no.	Field Name	Data Type	Width	Required	Unique	PK/FK	Referred Table	Description
1	id	int	-	yes	yes	yes	-	Auto Increment
2	item_name	varchar	30	yes	yes	no	-	
3	price	int	-	yes	yes	no	-	
4	model_year	varchar	4	yes	no	no	-	
5	photo	varchar	100	yes	no	no	-	
6	description	longtext	-	yes	no	no	-	
7	seller_id	int	-	yes	no	no	-	
8	mo_no	varchar	10	yes	no	no	-	
9	date	datetime	6	yes	no	no	-	
10	city	varchar	15	yes	no	no	-	
11	area	varchar	20	yes	no	no	-	
12	category	varchar	20	yes	no	no	-	

4. Implementation Details

4.1 Modules

The system consists of 4 basic modules namely

1. Navigation Module
2. User Module
3. Buyer Module
4. Seller Module

Each module consists of several methods to implement the required functionality. Implementation is done using Django. Database used in these modules is MySQL.

1. Navigation Module:

This module handles functionalities like login and sign-up in the system, logout from the system and other some basic functionalities like contact to admin, search required products etc.

2. User Module:

This module handles total product sold by user and profile-modification functionalities.

3. Seller Module:

This module handles functionalities like add item to the system, update item and delete item from the system.

4. Buyer Module:

This module handles functionalities like buy item, display filtered products by using filter and display required product details.

4.2 Function Prototypes

```
def search(request):
    query=str(request.GET['query'])
    if query is not "":
        if len(query)>75 and len(query)<1:
            Items=set()
        else:
            split_query=query.split()
            Items=set(Item.objects.all())
            for q in split_query:
                item1=Item.objects.filter(item_name__icontains=q)
                item2 = Item.objects.filter(city__icontains=q)
                item3 = Item.objects.filter(description__icontains=q)
                item=set(item1.union(item2,item3))
                Items=Items & item
            if len(Items)==0:
                messages.warning(request, "No search results found. Please refine your query.")
            params={'items': Items, 'query': query}
            return render(request, 'home/search.html', params)
    else:
        return redirect("/")
```

Search functionality

```
def updateProfile(request):
    if request.user.is_authenticated:
        if request.method=="POST":
            fm=UpdateProfileForm(request.POST,instance=request.user)
            if fm.is_valid():
                fm.save()
                messages.success(request,"Profile Modification Done")
                return redirect('/')
            else:
                messages.success(request,"Profile Modification Done")
                return redirect('/user/profile/update/')
        else:
            fm=UpdateProfileForm(instance=request.user)
            return render(request,'user/updateProfile.html',{'form':fm})
    else:
        return render(request,'error404.html')
```

Profile Modification

```

def addItem(request):
    if request.user.is_authenticated:
        if request.method=='POST':
            fm=AddItemForm(request.POST,request.FILES)
            if fm.is_valid():
                fm.save()
                item1=Item.objects.latest('id')
                return render(request,'seller/show_item.html',{'item':item1})
            else:
                messages.error(request,"form is invalid - Please Fill up correctly")
                return redirect("/seller/addItem/")
        else:
            usr_id=request.user.id
            fm=AddItemForm()
            fm.initial['seller_id']=usr_id
            fm.fields['seller_id'].widget=forms.HiddenInput()
            return render(request,'seller/addItem.html',{'form':fm})
    else:
        messages.error(request,"To sell Item Please Login!!!")
        return redirect('/')

```

Add Item to the system

```

def buyItem(request):
    if request.user.is_authenticated:
        if request.method=="POST":
            fm=FilterForm(request.POST)
            if fm.is_valid():
                form=fm.cleaned_data
                query=form.get('query')
                location=form.get('location')
                min_price=int(form.get('min_price'))
                max_price=int(form.get('max_price'))

                if len(query)>75:
                    Items=Item.objects.none()
                else:
                    itemByName = Item.objects.filter(item_name__icontains=query)
                    itemByCategory = Item.objects.filter(category__icontains=query)
                    itemByDescr = Item.objects.filter(description__icontains=query)
                    itemByUnion = set(itemByName.union(itemByCategory,itemByDescr))
                    itemByFilter=set(Item.objects.filter(city=location,price__gte=min_price,price__lte=max_price))
                    Items=itemByFilter.intersection(itemByUnion)

                    if len(itemByFilter)==0:
                        messages.warning(request, "No search results found for this filter.Please enter other filter credentials")
                        return redirect("/buyer/buyItem/")
                    if len(Items)==0:
                        messages.warning(request, "No search results found. Please refine your query.")
                        return render(request, 'buyer/result.html',{'items': Items, 'query': query})
                else:
                    messages.warning(request,"please enter valid filter credentials!")
                    return redirect("/buyer/buyItem/")
            else:
                fm=FilterForm()
                return render(request,'buyer/buyItem.html',{'form':fm})
    else:
        messages.error(request,"To buy Item Please Login!!!")
        return redirect('/')

```

Buy Item

5. Testing

Manual Testing and Automated Unit Testing with selenium was performed in order to find and fix the bugs in development process.

Manual Testing :

Sr No.	Test Scenario	Expected Result	Actual Result	Status
1	Login with Correct Credentials	User should able to Log in	Success message : “Successfully Logged in ”	pass
2	Login with incorrect Credentials	User should not able to Log in	Error message : “Invalid credentials! Please try again!”	pass
3	Search Item with matching name	Shows Available Items	Shows Available possible Items related to query	pass
4	Search Item with different not matching name	No search result found	Redirect to Error page with message “No Search result found. Please refine your query”	pass
5	Buy item details with select city	Show available items in selected city	Shows totally available items in selected city with different area	pass
6	Profile modification with change credentials	All new changes are covered	Success message : “Profile Modification Done”	pass
7	Add item with it’s details	Item should be added to database	Item added Successfully and shows added item and it’s details	pass
8	Edit item details	Item details should be updated	Success message : “Item details updated successfully” and shows item with updated details	pass
9	Delete item	Item should be deleted from database	Success message : “Item deleted successfully” and deleted item is not shown in my item list	pass
10	Direct access from changing the url	Error-404 Page	Redirect 404-Not Found Page	pass

11	Logout	User should be logged out and restricted from the system until next login.	User is successfully logged out and not able to access the system without signing again.	pass
----	--------	--	--	------

Automated Testing with Selenium :

Sr No.	Test Scenario	Expected Result	Actual Result	Status
1	Home Page : click Buy Now Button	Redirect to buyItem form	Shows form for buy item	pass
2	Buy Item : fill up form automatically and click submit button	Output related to entered values	Shows items in selected specific city	pass
3	Home Page : Search item	Shows Available Items as per the query	Shows Available possible Items related to query	pass
4	Home Page : click Sell Now Button	Redirect to addItem form	Shows form for add item	pass
5	Sell Item : fill up add item form automatically and click submit button	Item should be added to database	Item added Successfully and shows added item and it's details	pass
6	Edit Item : make change in details through automatically and click update button	Item details should be updated	Success message : "Item details updated successfully" and shows item with updated details	pass

```

def add_item_test(self):
    self.browser.get('http://127.0.0.1:8000/seller/addItem/')
    self.browser.maximize_window()
    sel = Select(self.browser.find_element_by_xpath("//select[@name='category']"))
    sel.select_by_visible_text("Car")
    time.sleep(1)
    self.browser.find_element_by_name("item_name").send_keys("Car Hyundai")
    time.sleep(1)
    self.browser.find_element_by_name("price").send_keys("500000")
    time.sleep(1)
    self.browser.find_element_by_name("model_year").send_keys("2015")
    time.sleep(1)
    self.browser.find_element_by_name("model_year").send_keys("2015")
    time.sleep(1)
    sel = Select(self.browser.find_element_by_xpath("//select[@name='city']"))
    time.sleep(1)
    sel.select_by_visible_text("Vadodara")
    time.sleep(1)
    self.browser.find_element_by_name("area").send_keys("atladara")
    time.sleep(1)
    self.browser.find_element_by_xpath("//input[@type='file']").send_keys(r"C:\Users\Dell\Desktop\car.jpeg")
    time.sleep(1)
    self.browser.find_element_by_name("description").send_keys("Car is really good")
    time.sleep(1)
    self.browser.find_element_by_name("mo_no").send_keys("9563256463")
    time.sleep(1)
    self.browser.find_element_by_class_name("abc").click()
    time.sleep(10)

```

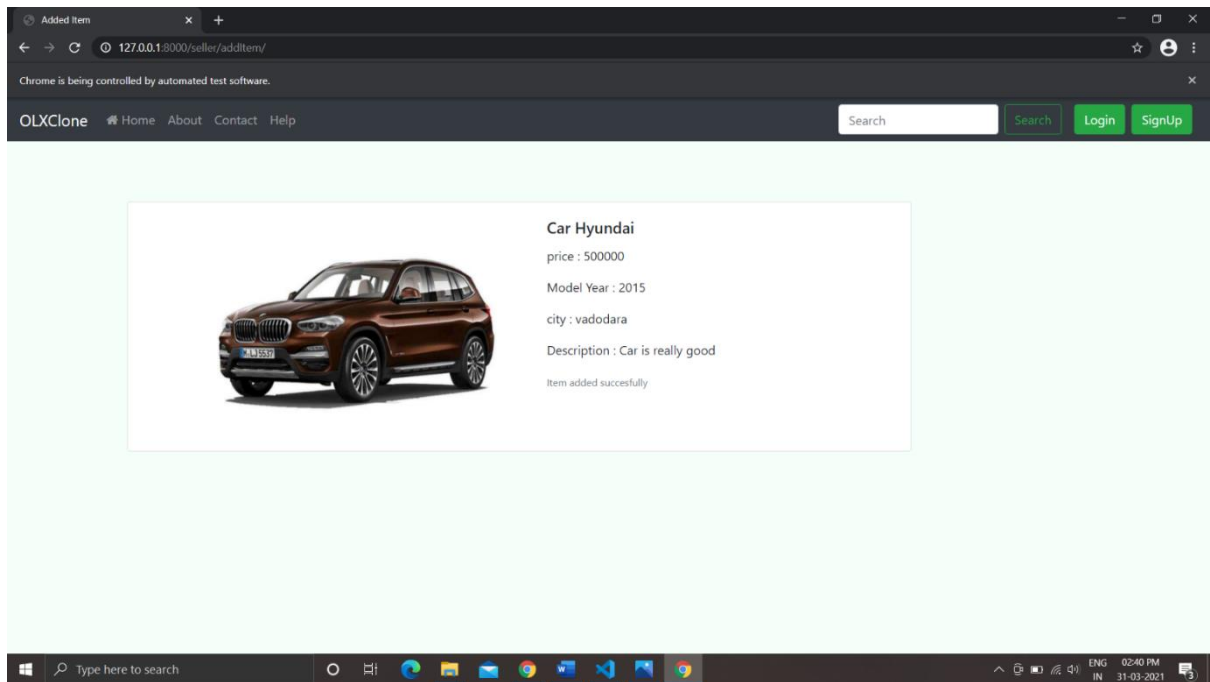
Testing function for Add Item

The screenshot displays a web browser window with the URL `http://127.0.0.1:8000/seller/addItem/`. The page title is "add item". The browser is controlled by automated test software. The page content includes a navigation bar with "OLXClone", "Home", "About", "Contact", and "Help" links, along with a search bar and "Search", "Login", and "SignUp" buttons. The main section is titled "Add Item Details" and contains the following form fields:

- Category: A dropdown menu with "Car" selected.
- Item name: A text input field containing "Car Hyundai".
- Price: A text input field containing "500000".
- Model year: An empty text input field.
- City: A dropdown menu with "Surat" selected.
- Area: An empty text input field.
- Image: A "Choose File" button with the text "No file chosen".

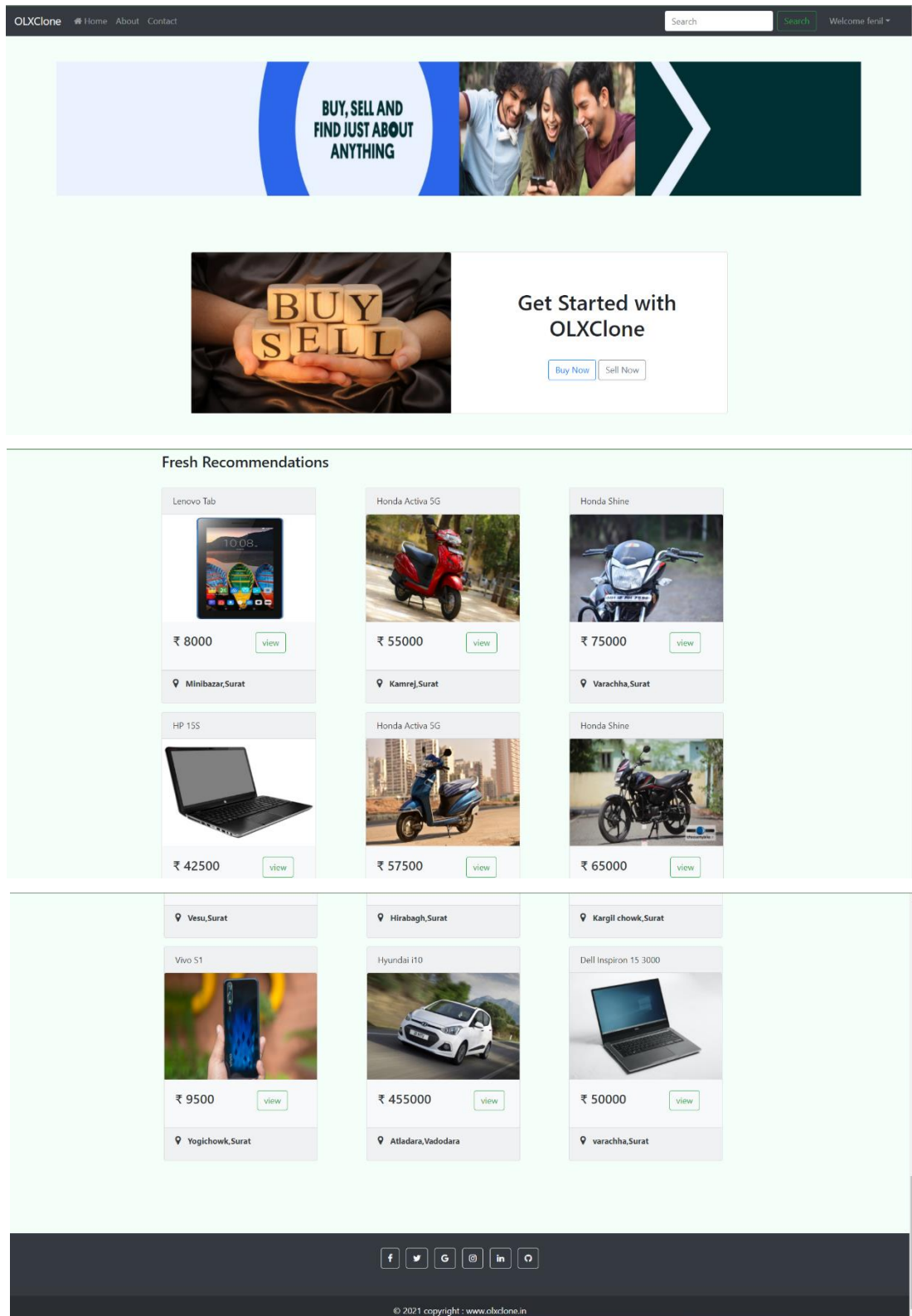
The Windows taskbar at the bottom shows the system clock as 02:39 PM on 31-03-2021.

Input given by automated testing tool



Output given by automated testing tool

6. Screenshots



OLXClone
Home
About
Contact
Search
Welcome fasil

Add Item Details

Category:

Tablet

Item name:

Lenovo Tab

Price:

8000

Model year:

2019

City:

Surat

Area:

Minibazar

Image:

Choose file
lenovo-Tab.jpg

Description:


Lenovo Tab black 4/16 Ram/Storage

Mobile Number:

9856456989

OLXClone
Home
About
Contact
Search
Welcome fasil

Items Details



Info

Name : Lenovo Tab
Price :: ₹ 8000
Model year: 2019

Minibazar, Surat
9856456989

Edit Item Details

Delete Item

Description

Lenovo Tab black 4/16 Ram/Storage

Items Details



Info

Name : Honda Shine

Price :: ₹ 75000

Model year: 2020

Varachha, Surat

9856214561

Contact Owner

Description

125 CC Engine , New Model ,Average : 52 km/ltr

Your Items

#	Date & Time	Product Name	Price	Model Year	Address	Edit	Delete	View
1	March 31, 2021, 1:54 p.m.	Vivo S1	9500	2018	Yogichowk, Surat	Edit	Delete	view
2	March 31, 2021, 2:46 p.m.	Honda Shine	75000	2020	Varachha, Surat	Edit	Delete	view
3	March 31, 2021, 3:46 p.m.	Honda Activa 5G	55000	2019	Kamrej, Surat	Edit	Delete	view
4	March 31, 2021, 3:58 p.m.	Lenovo Tab	8000	2019	Minibazar, Surat	Edit	Delete	view



7. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements. Functionalities that are successfully implemented in the system are:

- Sign-up, Sign-in with necessary validation on required fields
- Logout
- Add item with necessary details
- Update particular details of the item or whole item
- Delete item
- Filter items from all the items pricewise
- Search items by city or item-name or both
- Show required items directly or using filter or search
- Total sold products by particular user
- Contact to admin for any runtime issues
- Profile Modification

After the implementation and coding of system, **manual** and **automated unit testing** was performed on the system to determine the errors and possible flaws in the system.

8. Limitations and Future Extensions

Currently, we have not implemented the chat functionality because lack of time. But we want to implement this chat functionality in upcoming time, so that if any buyer has some query related to price of the item or some other query related to the item then he will be able to interact with the seller of that item directly in the system by chat functionality very easily.

Also we want to increase our system capability, so that more user can able to use our system. We will also improve some non-functional requirements of the system like safety, performance, supportability etc.

9. Reference / Bibliography

Book:

Fundamental of Software Engineering by Rajib Mall

Websites:

docs.djangoproject.com

getbootstrap.com

stackoverflow.com

tutorialspoint.com

youtube.com

Search Engine:

google.com