St. Thomas' College of Engineering and Technology

Documentation of the Project for Vocational Training by Dirac Business Solutions Private Limited

(July-2016)

Dynamic Web Project to create an e-Shopping application using Python

Name: Ritam Ganguly

Stream: CSE

Roll: 33

E-mail id: ritam.ganguly@yahoo.co.in

Introduction

I present here a Dynamic Web Project using Python. An e-Shopping dynamic web project has been created that can be divided into mainly two parts, first the module for customers, who will use the application to order products and the other module has been created for sellers, who can upload the details of the product that they want to sell. The customer module enables a customer to select a product and place it in his/her wish list for easy reference later and can pick up items to his/her cart to order them. Order quantity for each product has been limited to one quantity per order. The application also helps the seller to add new products and update the price of the existing products. Both the seller and the customer can manage the orders that have been placed.

Methodology

- Software used:
  - Python 2.7
  - HTML5
  - CSS
  - MySQL
  - XAMPP
  - SQL Yog
  - Jinja2
- The Application

  A database of all the required tables has been created. The table structures are as follows:
  Database – eShop

tbl_cust	**tbl_cart**
cust_id : number	cart_id : number
cust_name : varchar	cart_cust : number
cust_add : varchar	cart_prod : number
cust_username : varchar	
cust_password : varchar	
tbl_sell	**tbl_wish**
sell_id : number	wish_id : number
sell_name : varchar	wish_cust : number
sell_add : varchar	wish_prod : number
sell_username : varchar	
sell_password : varchar	
tbl_prod	**tbl_ordr**
prod_id : number	ordr_id : number
prod_name : varchar	ordr_cust : number
prod_desp : varchar	ordr_prod : number
prod_mrp : number	ordr_sell : number
prod_sell : number	ordr_stat : varchar

Once the database is created (using the **eShop.sql** file) we are ready to run the application.
All the required classes of the respective packages are imported and the database is configured.

```
from flask import Flask, render_template, json, request,redirect
from flask.ext.mysql import MySQL
from werkzeug import generate_password_hash, check_password_hash
from flask import session

mysql = MySQL()
app = Flask(__name__)
app.secret_key = '1111'

app.config['MYSQL_DATABASE_USER'] = 'root'
app.config['MYSQL_DATABASE_PASSWORD'] = ''
app.config['MYSQL_DATABASE_DB'] = 'eShop'
app.config['MYSQL_DATABASE_HOST'] = 'localhost'
mysql.init_app(app)
```

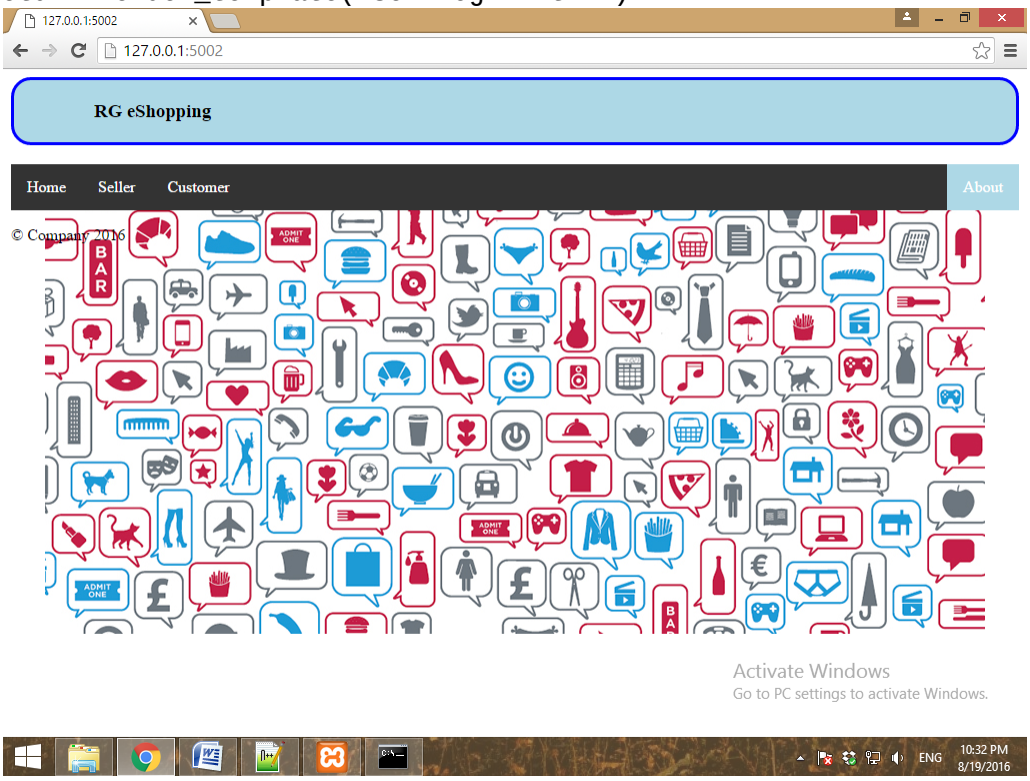
The root page is assigned and the port is configured for the application.

```
@app.route('/')
def main():
      return render_template('index.html')

if __name__ == "__main__":
      app.run(port=5002)
```

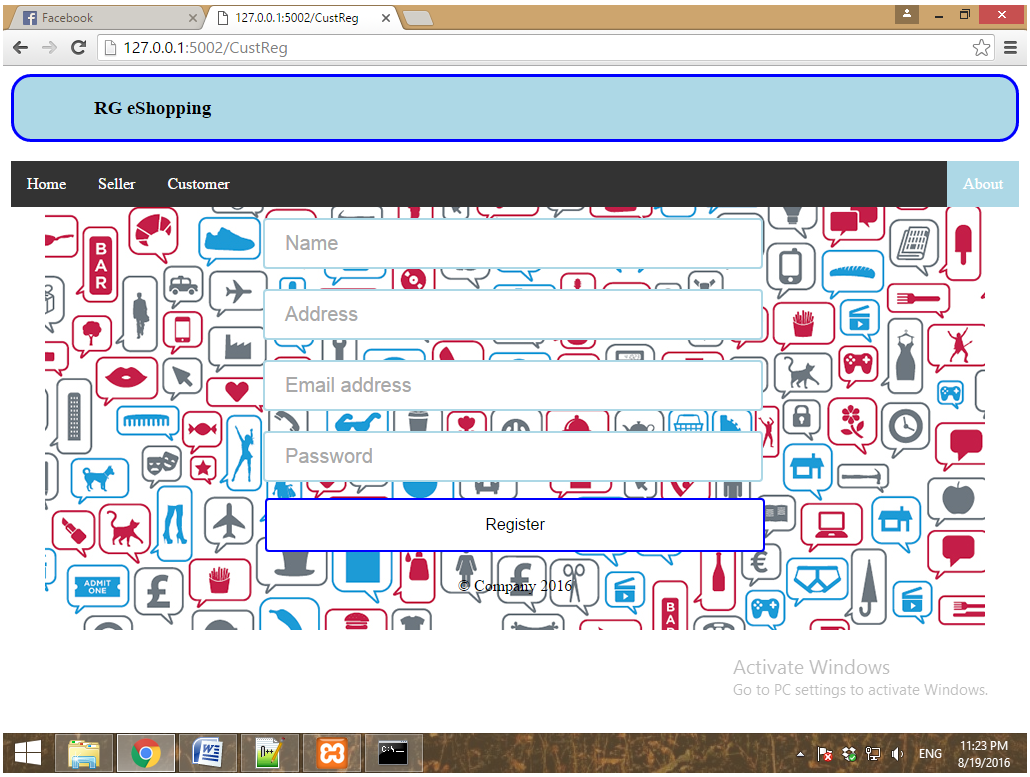
The tabs on the navigation bar that reads Seller and Customer are two drop-down menus that open on mouse-over. Each gives two options for the respective registration and the other for Sign In. The controls of the page to the other page are linked as

```
@app.route('/')
def main():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('About.html')

@app.route('/CustReg')
def CustReg():
      return render_template('CustRegl.html')
```

```
@app.route('/CustLogIn')
def CustLogIn():
      return render_template('CustLogIn.html')

@app.route('/SellReg')
def SellReg():
      return render_template('SellRegl.html')

@app.route('/SellLogIn')
def SellLogIn():
      return render_template('SellLogIn.html')
```



Index Page



Sign Up Page

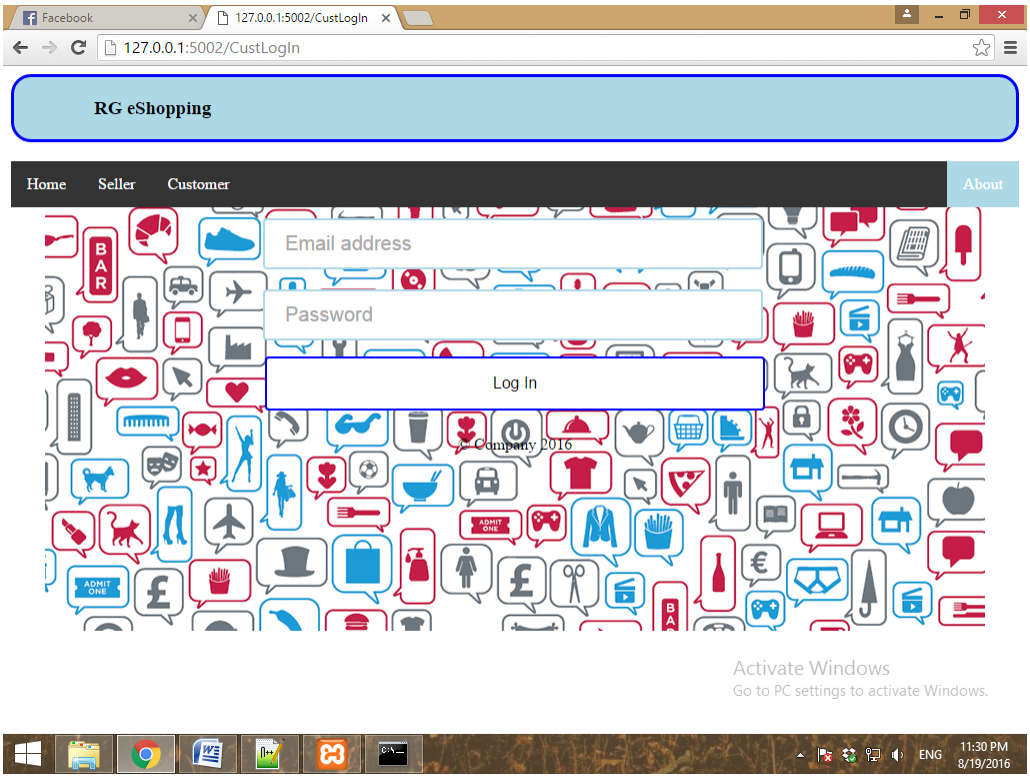
The Sign Up Page on clicking Register button first checks if all the fields have been filled or not using JavaScript if any field is found empty, an appropriate error message is shown.

```
function verify()
{
	if(document.getElementById("mail").value.length==0 ||
document.getElementById("name").value.length==0 ||
document.getElementById("add").value.length==0 ||
document.getElementById("pass").value.length==0)
		alert("All the fields are mandatory");
	return;
}
```

When all the fields are filled the following python subroutine is executed for registration of new seller. Similar is the subroutine for registration of a new customer, where a new record is added to the tbl_cust table in place of tbl_sell table here.

```
@app.route('/New_Sell', methods=['POST'])
def NewSell():
	_name=request.form['name']
	_address=request.form['address']
	_email=request.form['email']
	_password=request.form['password']
```

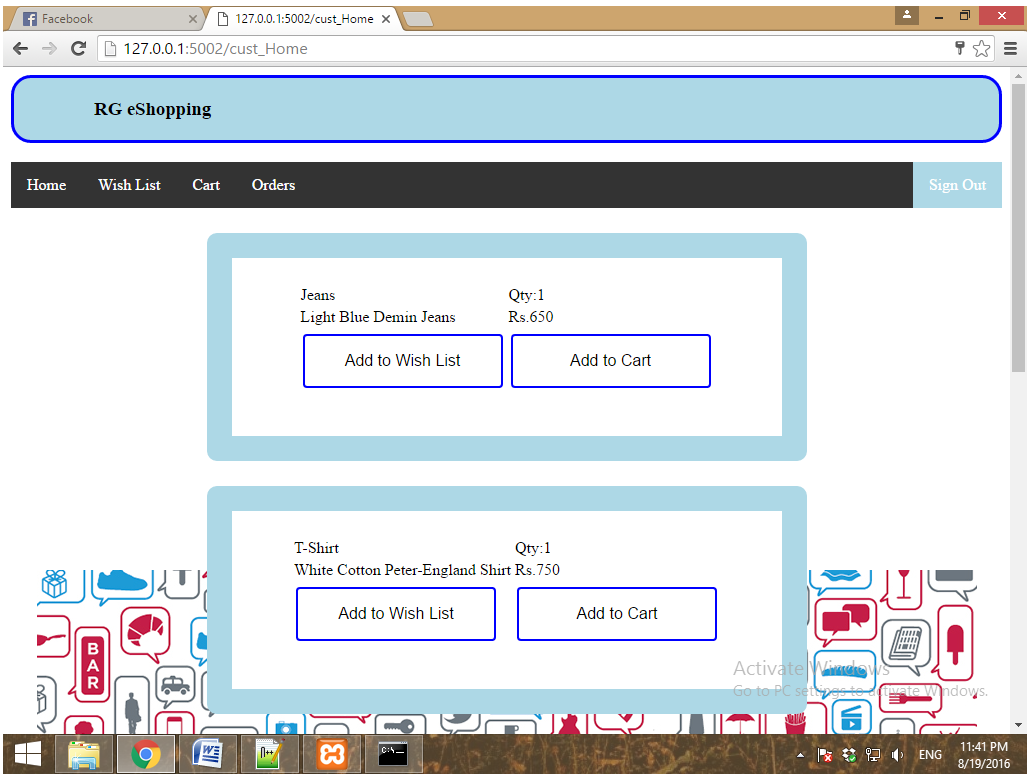
```
	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"

	cursor.execute("select max(sell_id) from tbl_sell;")
	data = cursor.fetchall()
	_id=data[0][0]+1

	cursor.close()
	cursor = conn.cursor()
	cursor.execute("insert into tbl_sell
values(%s,'%s','%s','%s','%s');"%(_id,_name,_address,_email,_password))
	data = cursor.fetchall()

	cursor.close()

	if len(data) == 0:
		conn.commit()
		conn.close()
		return render_template('SellLogIn.html')
	else:
		conn.close()
		return render_template('error.html',error = "Wrong Credentials
supplied")
```



Sign In Page

When Log In button in the Sign In page is clicked, it checks for the validation of the customer and opens the customer Home page.

```
@app.route('/cust_Home')
def custHome():
	_user=session.get('user')

	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"
	try:
		cursor.execute("select * from tbl_prod;")
		data = cursor.fetchall()
		if len(data) > 0:
			return render_template('CustHome.html',data =
data,len=len(data))
		else:
			return render_template('error1.html',error = 'Unauthorised
Customer')
	except Exception as e:
		return render_template('error1.html',error = e)
	finally:
		cursor.close()
		conn.close()
```



Customer Home Page

In the customer home page, all the products and their respective details are shown with two options with each other, one to add the product to the customer's wish list and the other to his cart. The following subroutine is executed for the respective action.

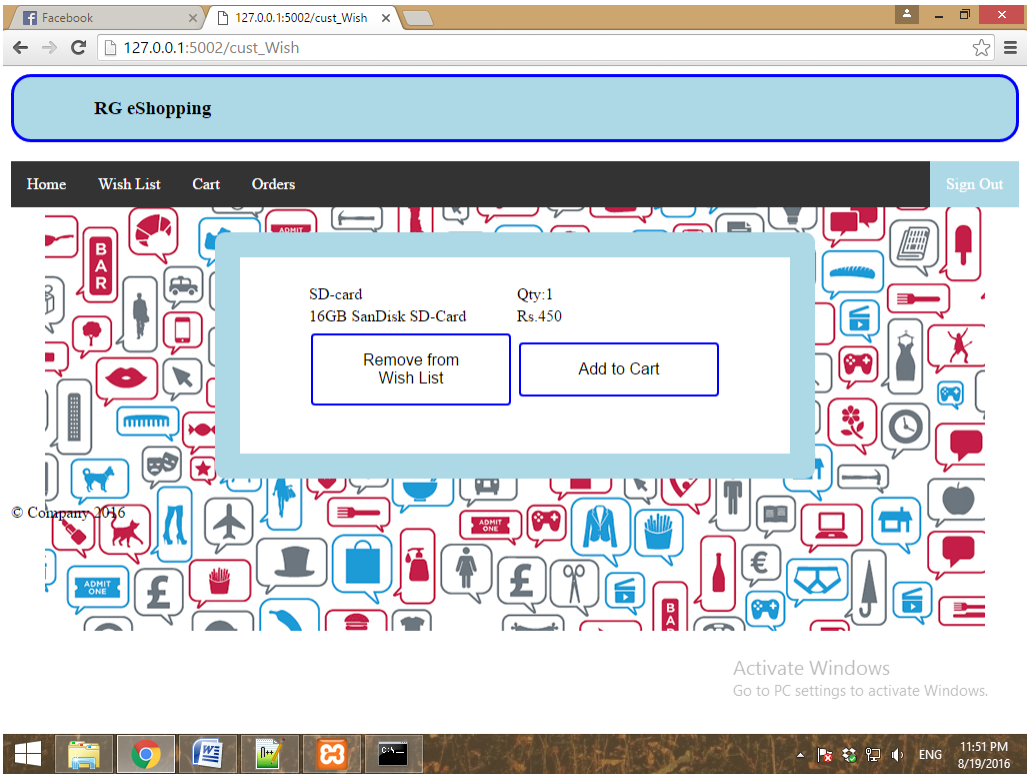
```
@app.route('/CustWishAdd', methods=['POST'])
def custWishAdd():
      _user = session.get('user')
      _prodId = request.form['id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("select max(wish_id) from tbl_wish")
            data = cursor.fetchall()
            _id = data[0][0]+1

            cursor.close()
            cursor = conn.cursor()
            cursor.execute("insert into tbl_wish values
(%s,%s,%s)"%(_id,_user,_prodId))
            data = cursor.fetchall()
            if len(data) == 0:
                  conn.commit()
                  cursor.close()
                  cursor = conn.cursor()
                  cursor.execute("select * from tbl_prod;")
                  data = cursor.fetchall()
                  cursor.close()
                  if len(data) > 0:
                        return render_template('CustHome.html',data =
data,len=len(data))
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()

@app.route('/CustCartAdd', methods=['POST'])
def custCartAdd():
      _user = session.get('user')
      _prodId = request.form['id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("select max(cart_id) from tbl_cart")
            data = cursor.fetchall()
            _id = data[0][0]+1
```

```
            cursor.close()
            cursor = conn.cursor()
            cursor.execute("insert into tbl_cart values
(%s,%s,%s)"%(_id,_user,_prodId))
            data = cursor.fetchall()
            if len(data) == 0:
                  conn.commit()
                  cursor.close()
                  cursor = conn.cursor()
                  cursor.execute("select * from tbl_prod;")
                  data = cursor.fetchall()
                  cursor.close()
                  if len(data) > 0:
                        return render_template('CustHome.html',data =
data,len=len(data))
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()
```

The navigation bar in the customer home page has three tabs, go to Wish List, go to Cart and go to Orders page. The respective subroutines are as follows

```
@app.route('/cust_Wish')
def custWish():
      _user=session.get('user')

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor = conn.cursor()
            cursor.execute("select count(*) from tbl_wish where
wish_cust=%s;"%(_user))
            data = cursor.fetchall()
            if data[0][0] == 0:
                  return render_template('error1.html',error = 'No Item(s)
Selected')
                  cursor.close()
            else:
                  cursor.execute("select * from tbl_wish where
wish_cust=%s;"%(_user))
                  data = cursor.fetchall()
                  if len(data) > 0:
                        cursor.close()
                        cursor = conn.cursor()

                        str=""
                        for i in range(len(data)):
```

```
                        if i == 0:
                              str = "%s"%(data[i][2])
                        else:
                              str = "%s,%s"%(str,data[i][2])

                  cursor.execute("select * from tbl_prod where prod_id
in (%s);"%(str))
                  data = cursor.fetchall()
                  cursor.close()
                  if len(data) > 0:
                        return render_template('CustWish.html',data =
data,len=len(data))
                  else:
                        return render_template('error1.html',error =
'Unauthorised Customer')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()
```

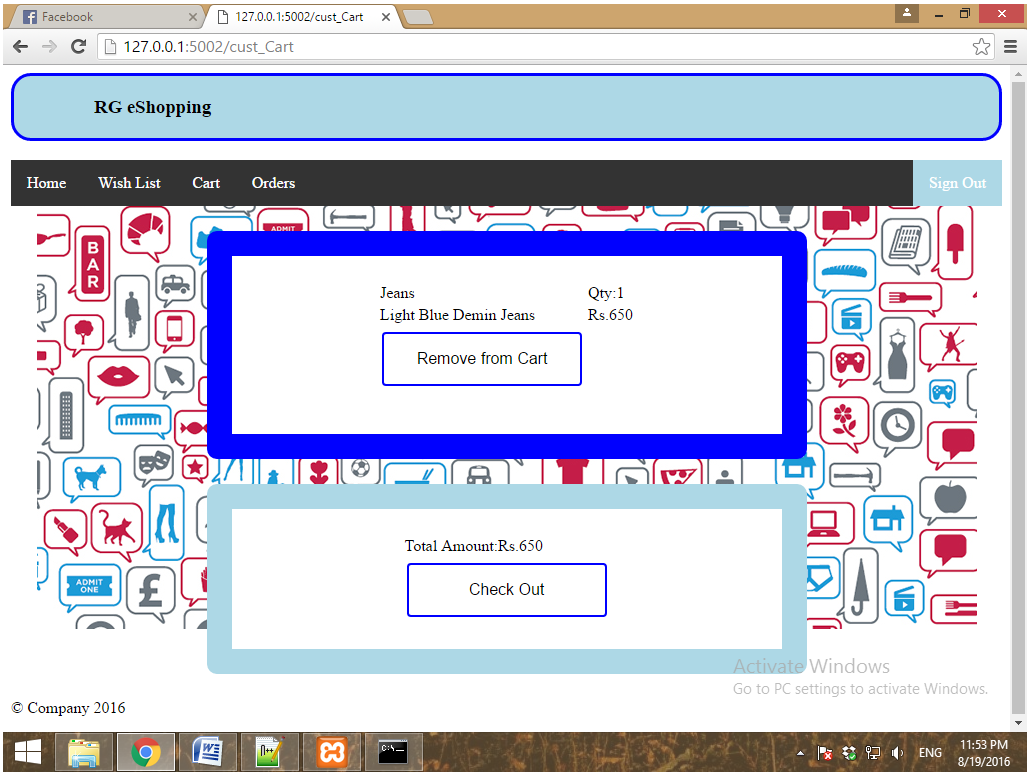


Wish List Page

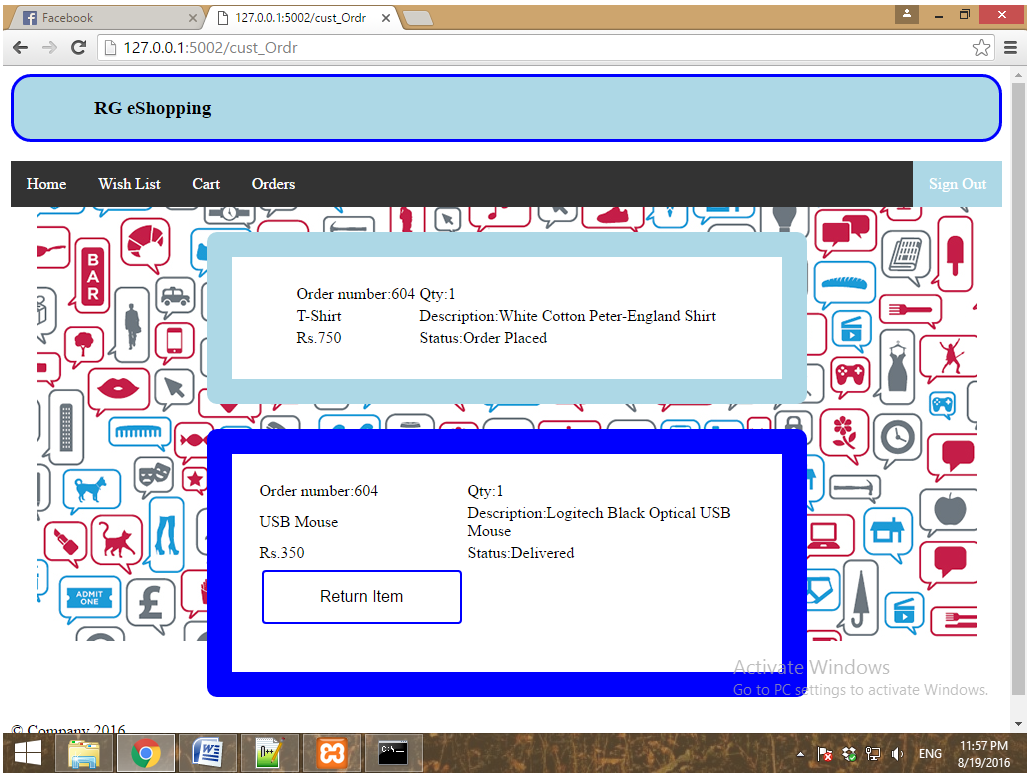
```
@app.route('/cust_Cart')
def custCart():
      _user=session.get('user')

      conn = mysql.connect()
      cursor = conn.cursor()
```

```
	print "Connection successful"
	total=0
	try:
		cursor = conn.cursor()
		cursor.execute("select count(*) from tbl_cart where
cart_cust=%s;"%(_user))
		data = cursor.fetchall()
		if data[0][0] == 0:
			return render_template('error1.html',error = 'No Item(s)
Selected')
			cursor.close()
		else:
			cursor.execute("select * from tbl_cart where
cart_cust=%s;"%(_user))
			data = cursor.fetchall()
			if len(data) > 0:
				cursor.close()
				cursor = conn.cursor()

				str=""
				for i in range(len(data)):
					if i == 0:
						str = "%s"%(data[i][2])
					else:
						str = "%s,%s"%(str,data[i][2])

				cursor.execute("select * from tbl_prod where prod_id
in (%s);"%(str))
				data = cursor.fetchall()
				for i in range(len(data)):
					total = total + data[i][3]
				cursor.close()
				if len(data) > 0:
					return
render_template('CustCart.html',total=total,data = data,len=len(data))
				else:
					return render_template('error1.html',error =
'Unauthorised Customer')
	except Exception as e:
		return render_template('error1.html',error = e)
	finally:
		conn.close()

@app.route('/cust_Ordr')
def custOrdr():
	_user=session.get('user')

	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"
```

```
	total=0
	try:
		cursor = conn.cursor()
		cursor.execute("select count(*) from tbl_ordr where
ordr_cust=%s;"%(_user))
		data = cursor.fetchall()
		if data[0][0] == 0:
			return render_template('error1.html',error = 'No Item(s)
Ordered')
			cursor.close()
		else:
			cursor.execute("select * from tbl_ordr,tbl_prod where
ordr_prod=prod_id and ordr_cust=%s;"%(_user))
			data = cursor.fetchall()
			print data
			if len(data) > 0:
				return
render_template('CustOrdr.html',total=total,data = data,len=len(data))
			else:
				return render_template('error1.html',error =
'Unauthorised Customer')
	except Exception as e:
		return render_template('error1.html',error = e)
	finally:
		conn.close()
```



Cart Page



Orders Page

In the wish list page, each product in the wish list have two actions with them, either one can remove the product from wish list or include the product to the cart. In the cart page, one can either remove a product from the cart or can place the order. The subroutines following the above tasks are as follows.

```
@app.route('/CustWishRev', methods=['POST'])
def custWishRev():
	_user = session.get('user')
	_prodId = request.form['id']

	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"
	try:
		cursor.execute("select wish_id from tbl_wish where wish_prod=%s
and wish_cust=%s;"%(_prodId,_user))
		data = cursor.fetchall()
		if len(data) > 0:
			cursor.close()
			cursor = conn.cursor()

			cursor.execute("delete from tbl_wish where
wish_prod=%s;"%(_prodId))
			data = cursor.fetchall()

			if len(data) == 0:
```

```
                        conn.commit()
                        cursor.close()
                        return redirect('/cust_Wish')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()

@app.route('/CustCartRev', methods=['POST'])
def custCartRev():
      _user = session.get('user')
      _prodId = request.form['id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("select cart_id from tbl_cart where cart_prod=%s
and cart_cust=%s;"%(_prodId,_user))
            data = cursor.fetchall()
            if len(data) > 0:
                  cursor.close()
                  cursor = conn.cursor()

                  cursor.execute("delete from tbl_cart where
cart_prod=%s;"%(_prodId))
                  data = cursor.fetchall()

                  if len(data) == 0:
                        conn.commit()
                        cursor.close()
                        return redirect('/cust_Cart')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()

@app.route('/WishCartAdd', methods=['POST'])
def custWishCartAdd():
      _user = session.get('user')
      _prodId = request.form['id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("delete from tbl_wish where
wish_prod=%s"%(_prodId))
            data = cursor.fetchall()
            if len(data) == 0:
```

```
                  cursor.close()
                  cursor = conn.cursor()

                  cursor.execute("select max(cart_id) from tbl_cart")
                  data = cursor.fetchall()
                  _id = data[0][0]+1

                  cursor.close()
                  cursor = conn.cursor()
                  cursor.execute("insert into tbl_cart values
(%s,%s,%s)"%(_id,_user,_prodId))
                  data = cursor.fetchall()
                  if len(data) == 0:
                        conn.commit()
                        cursor.close()
                        return redirect('/cust_Wish')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()

@app.route('/custOrdrPlcd', methods=['POST'])
def custOrderPlaced():
      _user = session.get('user')

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("select * from tbl_cart where
cart_cust=%s;"%(_user))
            data1 = cursor.fetchall()
            print "select cart_prod"
            if len(data1) > 0:
                  cursor.close()

                  cursor = conn.cursor()
                  cursor.execute("select max(ordr_id) from tbl_ordr;")
                  data = cursor.fetchall()
                  print "order id max"
                  if len(data) > 0:
                        cursor.close()
                        _id = data[0][0]+1

                        cursor = conn.cursor()
                        cursor.execute("delete from tbl_cart where
cart_cust=%s"%(_user))
                        data = cursor.fetchall()
                        print "delete cart"
                        if len(data) == 0:
```

```
                              cursor.close()

                              for i in range(len(data1)):
                                    cursor = conn.cursor()
                                    print "select prod_sell from tbl_prod
where prod_id=%s;"%(data1[i][2])
                                    cursor.execute("select prod_sell from
tbl_prod where prod_id=%s;"%(data1[i][2]))
                                    data2 = cursor.fetchall()
                                    print "select sell_id"
                                    if len(data2) > 0:
                                          cursor.close()
                                          cursor = conn.cursor()
                                          cursor.execute("insert into
tbl_ordr values(%s,%s,%s,%s,'Order
Placed');"%(_id,_user,data1[i][2],data2[0][0]))
                                          data3 = cursor.fetchall()
                                          print "insert ordr"
                                          if len(data3) == 0:
                                                cursor.close()
                                                conn.commit()
                              else:
                                    return
render_template('error1.html',error = 'Order Placed...Thank You!!')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            conn.close()
```

In the orders page, each order can be in five different status and actions on them depend on the status of one order. If an order is in ‘Order Placed’, ‘Shipped’, ‘Return Initiated’ or ‘Returned’ state, no action is associated with that order. Whereas if an order is in ‘Delivered’ state it has an action to return the order using the following subroutine.

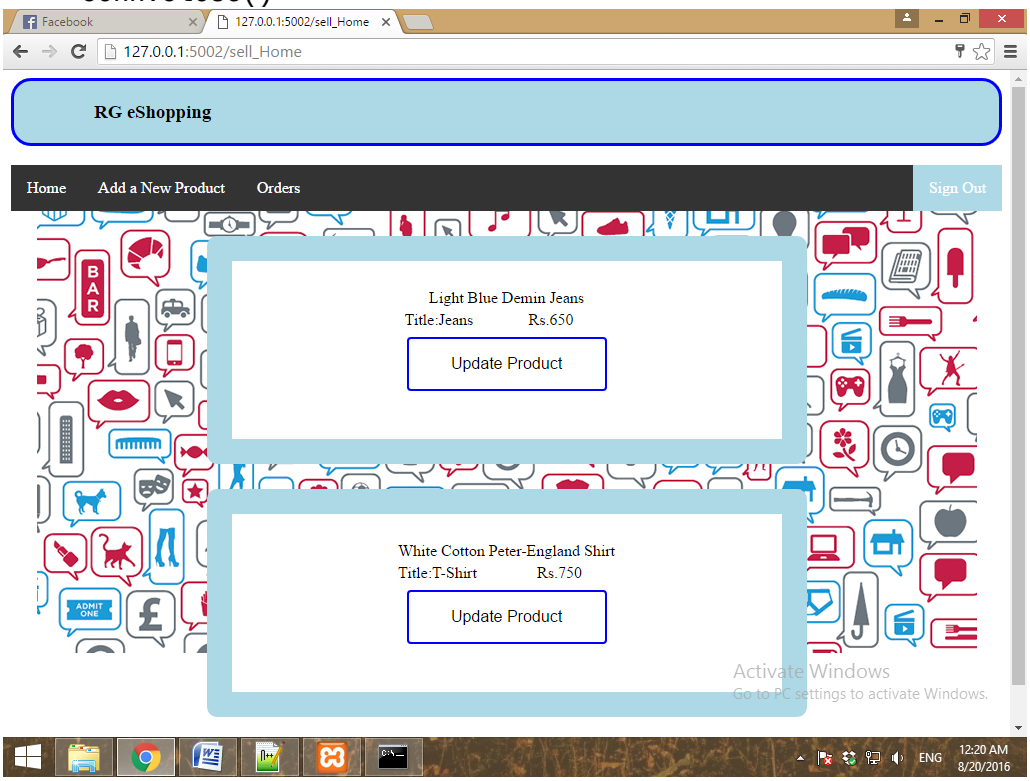
```
@app.route('/cust_Return', methods=['POST'])
def custOrderReturned():
      _user = session.get('user')
      _prodId = request.form['prod_id']
      _ordrId = request.form['ordr_id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("update tbl_ordr set ordr_stat='Return Initiated'
where ordr_prod=%s and ordr_id=%s;"%(_prodId,_ordrId))
            data = cursor.fetchall()
            print "update tbl_ordr set ordr_stat='Return Initiated' where
ordr_id=%s and ordr_prod=%s;"%(_prodId,_ordrId)
```

```
            if len(data) == 0:
                  conn.commit()
                  return redirect('/cust_Ordr')
      except Exception as e:
            return render_template('error1.html',error = e)
      finally:
            cursor.close()
            conn.close()
```

In the sellers module, after authenticate login, it is taken to the seller's home page where all the products sold by that seller is shown. Each product can be updated (Price only). One can also add a new product and manage the orders placed with the help the following subroutine.

```
@app.route('/sell_Home')
def sellHome():
      _user=session.get('user')

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful:%s"%(_user)
      try:
            cursor.execute("select * from tbl_prod where
prod_sell=%s;"%(_user))
            data = cursor.fetchall()
            print  data
            if len(data) > 0:
                  return render_template('SellHome.html',data =
data,len=len(data))
            else:
                  return render_template('error2.html',error = "No Products")
      except Exception as e:
            return render_template('error2.html',error = e)
      finally:
            cursor.close()
            conn.close()

@app.route('/sell_AddProd')
def sellAddProd():
      return render_template('SellAddProd.html')

@app.route('/New_Prod', methods=['POST'])
def NewProd():
      _user=session.get('user')
      _name=request.form['name']
      _desc=request.form['desp']
      _mrp=request.form['mrp']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
```

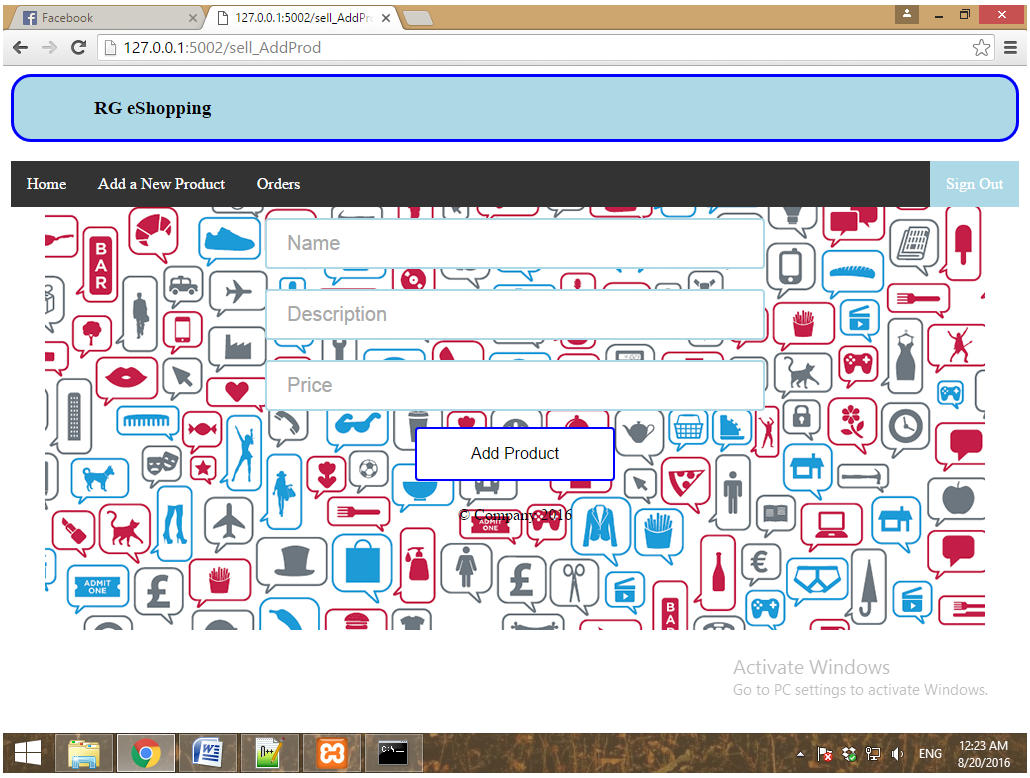
```
	try:
		cursor.execute("select max(prod_id) from tbl_prod;")
		data = cursor.fetchall()
		_id=data[0][0]+1

		cursor.close()
		cursor = conn.cursor()
		cursor.execute("insert into tbl_prod
values(%s,'%s','%s',%s,%s);"%(_id,_name,_desc,_mrp,_user))
		data = cursor.fetchall()
		print "1"
		if len(data) == 0:
			conn.commit()
			print "2"
			return redirect('/sell_Home')
		else:
			return render_template('error2.html',error = "Wrong
Credentials supplied")
	except Exception as e:
		return render_template('error2.html',error = e)
	finally:
		cursor.close()
		conn.close()
```



Seller Home Page

```
@app.route('/sell_Ordr')
def sellOrdr():
```

```
	_user=session.get('user')

	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"
	total=0
	try:
		cursor = conn.cursor()
		cursor.execute("select count(*) from tbl_ordr where
ordr_sell=%s;"%(_user))
		data = cursor.fetchall()
		if data[0][0] == 0:
			return render_template('error2.html',error = 'No Item(s)
Ordered')
			cursor.close()
		else:
			cursor.execute("select * from tbl_ordr,tbl_prod,tbl_cust
where ordr_prod=prod_id and ordr_sell=%s and ordr_cust=cust_id;"%(_user))
			data = cursor.fetchall()
			print data
			if len(data) > 0:
				return
render_template('SellOrdr.html',total=total,data = data,len=len(data))
			else:
				return render_template('error2.html',error =
'Unauthorised Customer')
	except Exception as e:
		return render_template('error2.html',error = e)
	finally:
		conn.close()
@app.route('/SellUpdProd', methods=['POST'])
def prodUpdt():
	_user=session.get('user')
	_id=request.form['id']

	conn = mysql.connect()
	cursor = conn.cursor()
	print "Connection successful"
	try:
		cursor.execute("select * from tbl_prod where prod_id=%s"%(_id))
		data = cursor.fetchall()
		if len(data) > 0:
			return
render_template('sellUpdt.html',idp=data[0][0],name=data[0][1],desc=data[0][2
],mrp=data[0][3])
	except Exception as e:
		return render_template('error2.html',error = e)
	finally:
		cursor.close()
		conn.close()
```



Add Product Page

Orders Page

```
@app.route('/sellUpdtMRP', methods=['POST'])
def prodUpdtMRP():
      _user=session.get('user')
      _id=request.form['id']
      _mrp=request.form['mrp']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("update tbl_prod set prod_mrp=%s where
prod_id=%s;"%(_mrp,_id))
            data = cursor.fetchall()
            if len(data) == 0:
                  conn.commit()
                  return redirect('sell_Home')
      except Exception as e:
            return render_template('error2.html',error = e)
      finally:
            cursor.close()
            conn.close()
```

Here too, in the order's page, one has an action related to a order depending upon its status. When an order is in 'Order Placed', 'Shipped' or 'Return Initiated' state, a seller has an option to update its status to the next. When in 'Delivered' or 'Returned' state no action is associated with it.

```
@app.route('/sell_Updt', methods=['POST'])
def ordrUpdt():
      _user=session.get('user')
      _prodid=request.form['prod_id']
      _ordrid=request.form['ordr_id']

      conn = mysql.connect()
      cursor = conn.cursor()
      print "Connection successful"
      try:
            cursor.execute("select ordr_stat from tbl_ordr where ordr_prod=%s
and ordr_id=%s;"%(_prodid,_ordrid))
            data = cursor.fetchall()
            if len(data) > 0:
                  _status = "Order Placed"
                  if data[0][0] == "Order Placed":
                        _status = "Shipped"
                  elif data[0][0] == "Shipped":
                        _status = "Delivered"
                  elif data[0][0] == "Return Initiated":
                        _status = "Returned"
                  cursor.close()
```

```
                  cursor = conn.cursor()
                  cursor.execute("update tbl_ordr set ordr_stat='%s' where
ordr_prod=%s and ordr_id=%s;"%(_status,_prodid,_ordrid))
                  data = cursor.fetchall()
                  if len(data) == 0:
                        conn.commit()
                        cursor.close()
                        cursor = conn.cursor()
                        cursor.execute("select count(*) from tbl_ordr where
ordr_sell=%s;"%(_user))
                        data = cursor.fetchall()
                        if data[0][0] == 0:
                              return render_template('error2.html',error =
'No Item(s) Ordered')
                              cursor.close()
                        else:
                              cursor.execute("select * from
tbl_ordr,tbl_prod,tbl_cust where ordr_prod=prod_id and ordr_sell=%s and
ordr_cust=cust_id;"%(_user))
                              data = cursor.fetchall()
                              print data
                              if len(data) > 0:
                                    return
render_template('SellOrdr.html',data = data,len=len(data))
                              else:
                                    return
render_template('error2.html',error = 'Unauthorised Customer')
      except Exception as e:
            return render_template('error2.html',error = e)
      finally:
            cursor.close()
            conn.close()
```

Conclusion

The application aims at providing an interface for both customer and seller an interactive shopping experience, though it has various room for improvement including adding pictures of products and making searching of products an easier task by using a search bar.