**Chapter 3: Implementation Tools**

This e-commerce platform has been developed by using a set of languages.

*Front-End Development*

For designing front-end of the website, HTML 5 with Javascript has been used. HTML is a markup language, which is used to develop and design webpages. Javascript is a language, which makes a webpage more responsive and shifts less load on server.

*Back-end Development*

Python language as back-end server language is used. The framework, which has been used, is Django framework. It is a framework used for developing websites. It is like Model View Controller (MVC) framework where controller renders the webpages, models stores the data regarding website and view manages the interface of webpages.

*Integrated Development Environment*

For this project, JetBrains’ PyCharm 2017.1.2 has been used. IDE provides ease to develop and design websites. It is a licensed software providing a free trial version for 30 days.

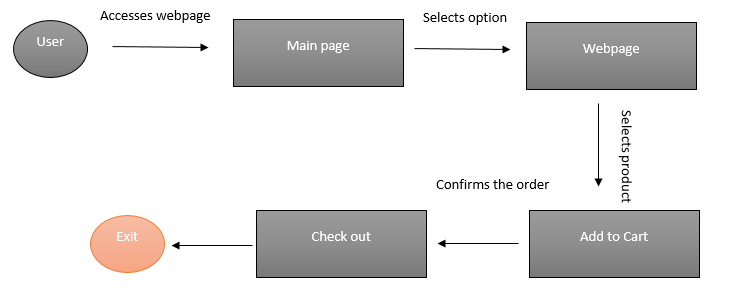
**Chapter 4: Application Development and Design**

**Software Development Process**

All the requirements of the e-commerce platform has been finalized initially so, before development, all the requirements were fixed. Based on those requirements, the models for the platform has been developed. Interfaces or webpages have been developed using HTML 5 incorporated with Javascript. The server side script has been developed using Python programming language.

**Algorithm Implementation with Code**

The generalized algorithm is shown in figure given below.



**Figure 1. Generalized Algorithm of Working of System**

The user accesses the main page of website that provides the options to user to search according to disease or not. After selecting the option, the relevant webpage is rendered from where user can select the products to be purchased. After confirming the product to be purchased, the user enters the details if the user is un-registered, otherwise, the user has to enter just contact number and email address. The user checks out the buying process and the receipt of the purchasing item is sent as email to the user.

**Front-End vs Back End Development**

Website Structure

Four HTML webpages named **index, addtoCart, fruits,** and **mainpage** have been designed. The **view.py** file acts as controller that controls the view of website. All the models are stored in the **models.py** file. The models that have been developed are given below.

**from** django.db **import** models  
  
*# Create your models here.***class** City(models.Model):  
 label=models.CharField(max\_length=250)  
  
 **def** \_\_str\_\_(self):  
 **return** self.label  
  
  
**class** Area(models.Model):  
 label=models.CharField(max\_length=250)  
 city\_label=models.ForeignKey(City,on\_delete=models.CASCADE)  
 **def** \_\_str\_\_(self):  
 **return** self.label+**"\t"**+self.city\_label.label  
  
**class** Veges(models.Model):  
 label=models.CharField(max\_length=250)  
 price=models.IntegerField()  
 img=models.CharField(max\_length=250)  
 **def** \_\_str\_\_(self):  
 **return** str(self.price)+**"\t"**+self.label  
  
**class** Fruits(models.Model):  
 label=models.CharField(max\_length=250)  
 price=models.IntegerField()  
 img=models.CharField(max\_length=250)  
 **def** \_\_str\_\_(self):  
 **return** self.label  
  
**class** farmDesciption(models.Model):  
 veges\_label=models.ForeignKey(Veges,on\_delete=models.CASCADE)  
 description=models.CharField(max\_length=500)  
 **def** \_\_str\_\_(self):  
 **return** self.veges\_label.label  
  
**class** nutrientsVeges(models.Model):  
 vege=models.ForeignKey(Veges,on\_delete=models.CASCADE)  
 carbohydrates=models.FloatField()  
 proteins=models.FloatField()  
 energy=models.FloatField()  
 fats = models.FloatField()  
 sugar = models.FloatField()  
 potassium = models.FloatField()  
 iron = models.FloatField()  
 calcium = models.FloatField()  
 **def** \_\_str\_\_(self):  
 **return** self.vege.label  
  
**class** registeredUser(models.Model):  
 userName=models.CharField(max\_length=400)  
 area=models.ForeignKey(Area,on\_delete=models.CASCADE)  
 address=models.CharField(max\_length=400)  
 contactNum=models.CharField(max\_length=400)  
 email=models.CharField(max\_length=250)  
 **def** \_\_str\_\_(self):  
 **return** self.userName+**"\t"**+self.area.label+**"\t"  
  
class** orderDetails(models.Model):  
 user = models.ForeignKey(registeredUser, on\_delete=models.CASCADE)  
 orderID = models.IntegerField()  
 status = models.CharField(max\_length=10)  
 cost = models.FloatField()  
 date = models.DateField()  
 **def** \_\_str\_\_(self):  
 **return** self.user.userName+**"\t"**+self.status+**"\t"**+str(self.cost)  
  
**class** fruitOrder(models.Model):  
 orderID=models.ForeignKey(orderDetails,on\_delete=models.CASCADE)  
 fruit=models.ForeignKey(Fruits,on\_delete=models.CASCADE)  
 **def** \_\_str\_\_(self):  
 **return** self.fruit.label+**"\t"**+str(self.orderID.orderID)  
  
**class** vegeOrder(models.Model):  
 orderID=models.ForeignKey(orderDetails,on\_delete=models.CASCADE)  
 vege=models.ForeignKey(Veges,on\_delete=models.CASCADE)  
**class** nutrientsFruits(models.Model):  
 fruit=models.ForeignKey(Fruits,on\_delete=models.CASCADE)  
 carbohydrates=models.FloatField()  
 proteins=models.FloatField()  
 energy=models.FloatField()  
 fats = models.FloatField()  
 sugar = models.FloatField()  
 potassium = models.FloatField()  
 iron = models.FloatField()  
 calcium = models.FloatField()  
 **def** \_\_str\_\_(self):  
 **return** self.fruit.label

Libraries

Controller has imported the following libraries.

1. Render (to control or render the webpages)
2. Models from **models.py**
3. Datetime (to select the date of order)
4. Smtplib (library to handle email process, dealings with SMTP server)
5. Q from django.db.models (for complex queries)
6. RequestContext (to send processed information to webpage)

**from** django.shortcuts **import** render  
**from import** City, Area,Veges,nutrientsVeges,registeredUser,orderDetails,Fruits,nutrientsFruits,fruitOrder,vegeOrder  
**import** datetime  
**import** smtplib  
**from** django.db.models **import** Q  
**from** django.template **import** RequestContext

Front End Development

The users accesses the main page.



Figure 2. Main Page

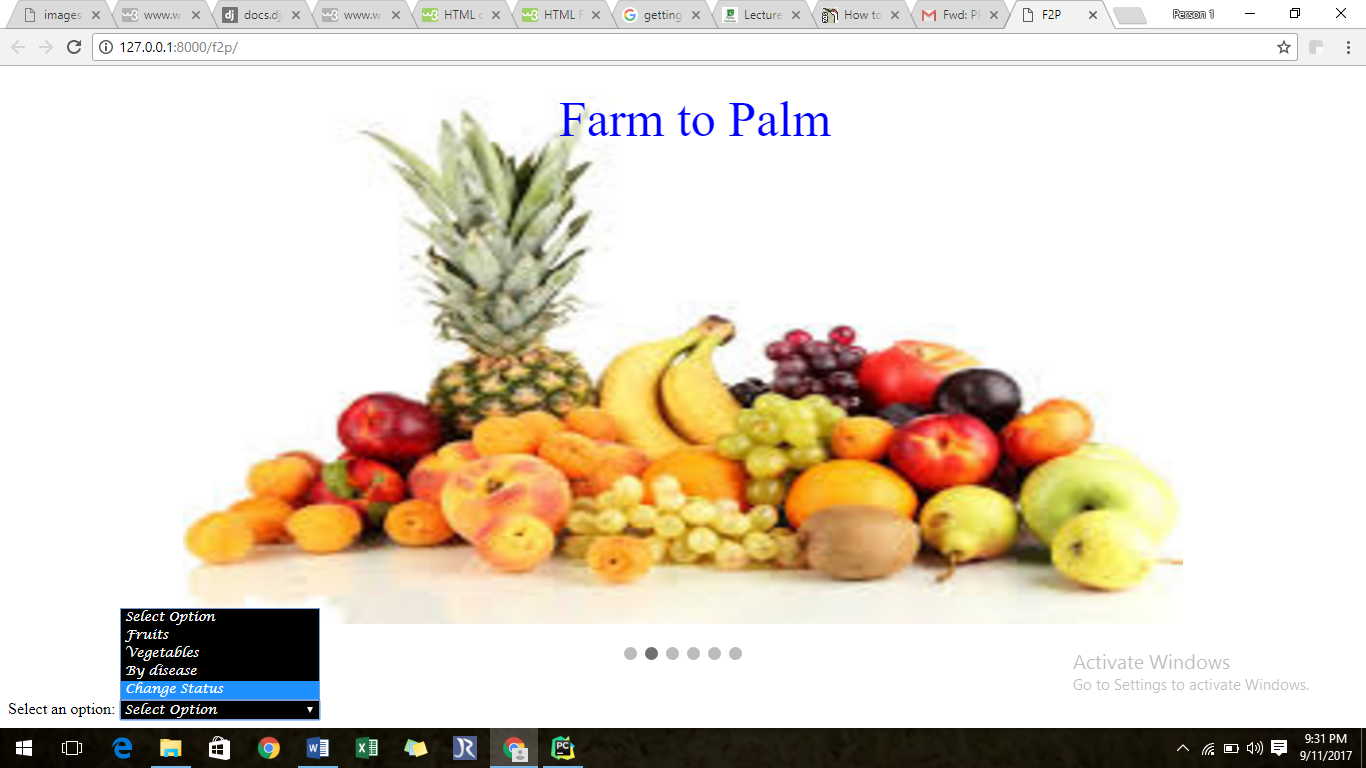
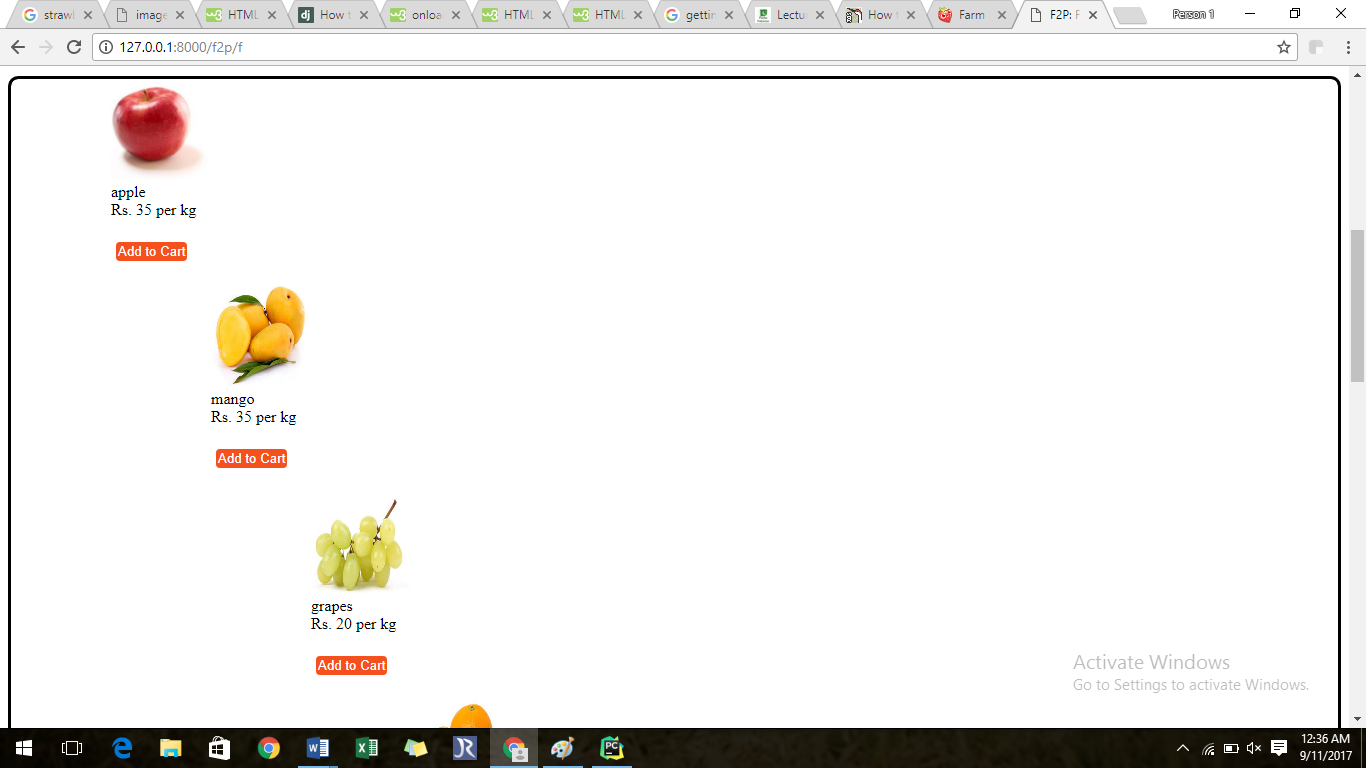


Figure 3. Selection of Option

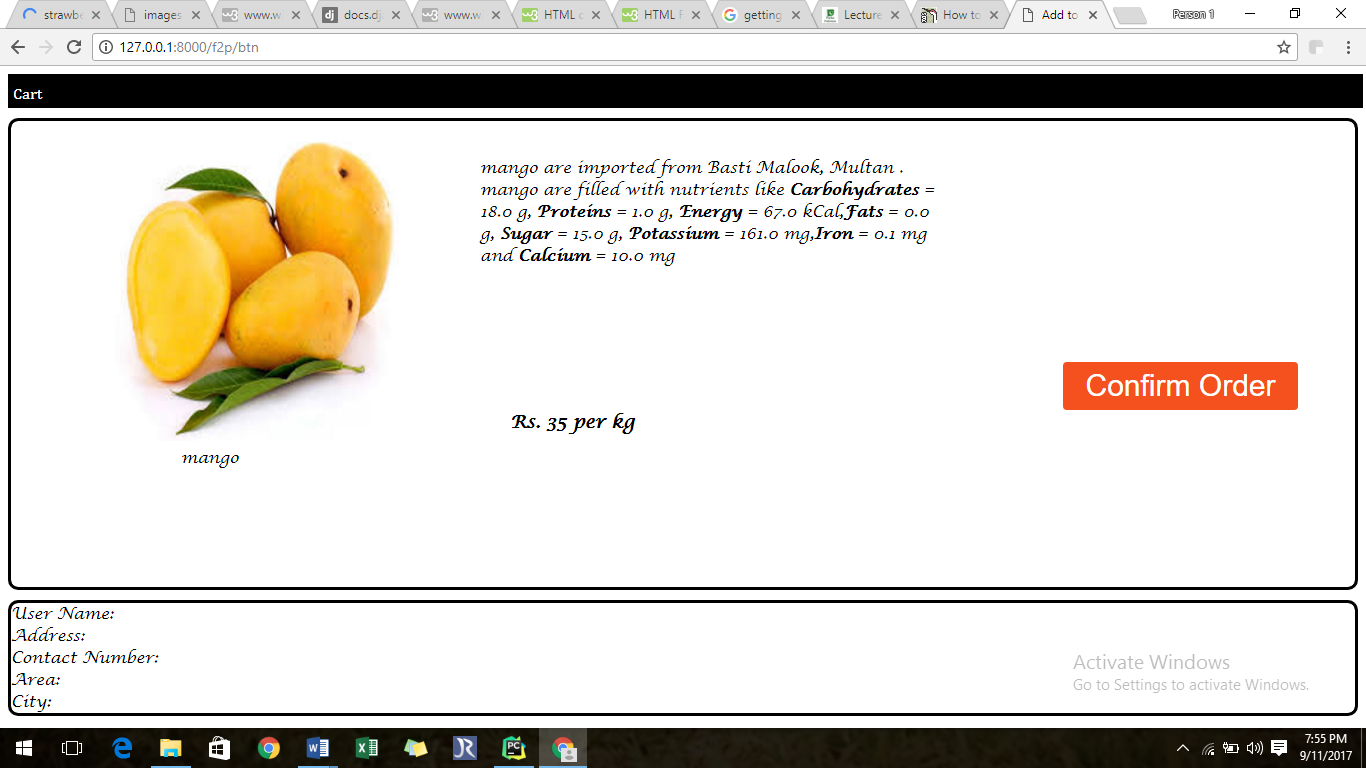
If the user selects the **fruits** option, then he is navigated to the fruits page.



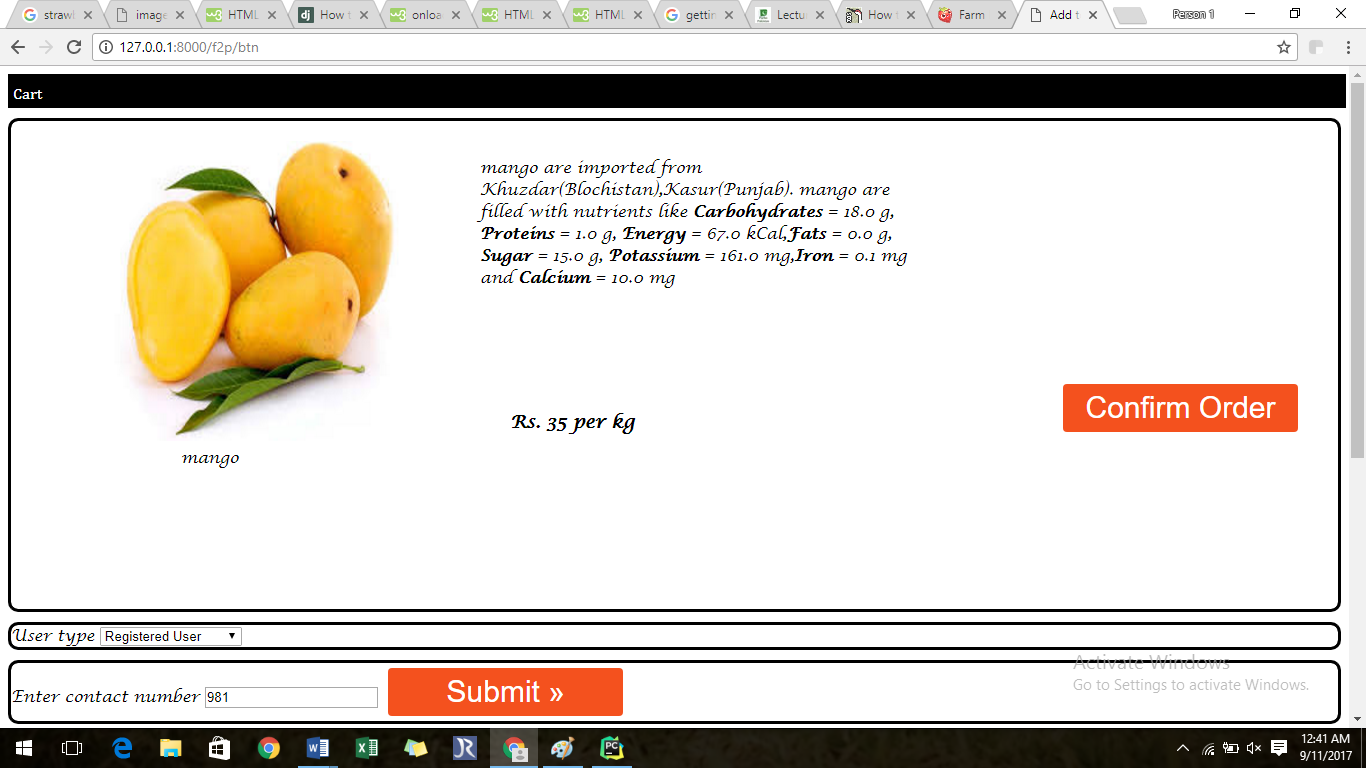
A list of fruits, with their prices are shown on the page. If the users hovers the image, description regarding the farmer is shown.



When the product is **added to the cart**, the system redirects to the page confirming about the order. The nutritional values of the product can also be viewed.



After confirming order, the user is asked to either provide details or to provide contact number if registered.

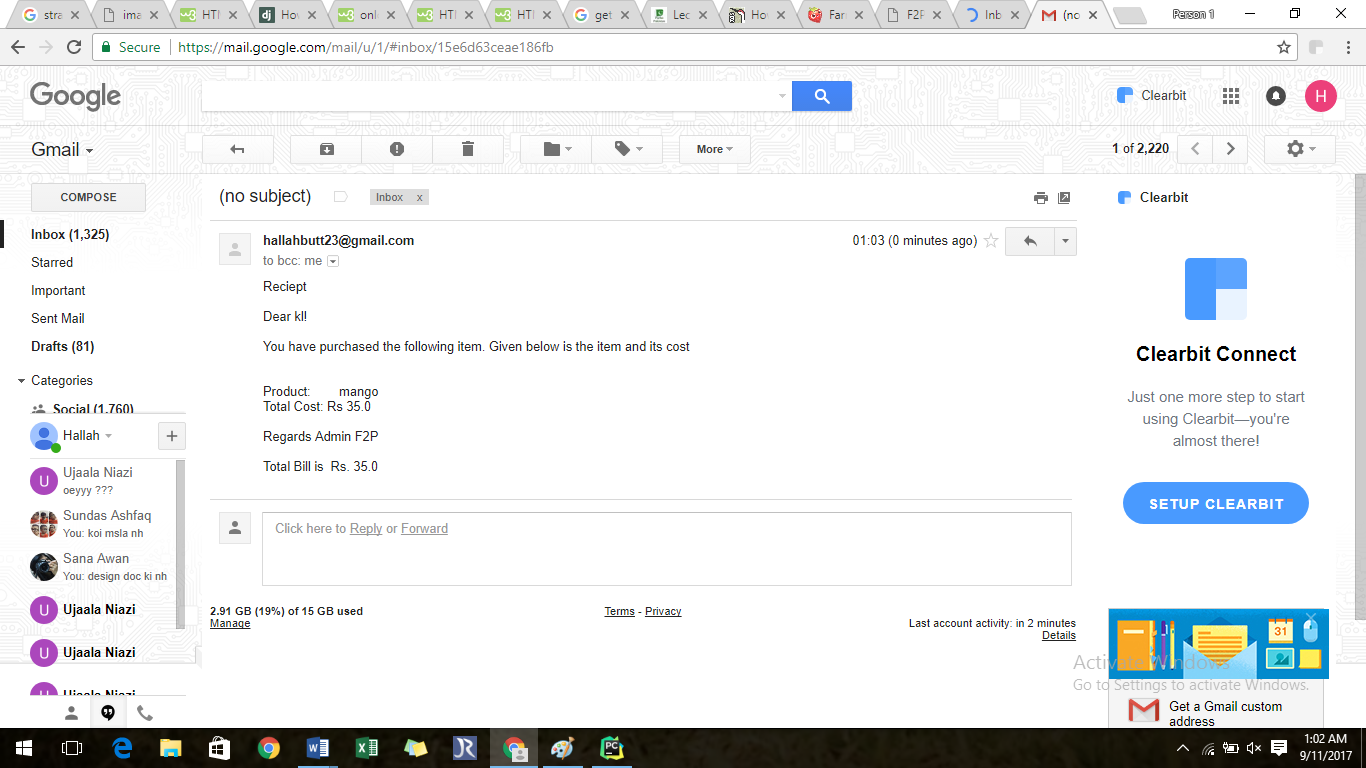




Details of registered user are shown after entering contact number. The quantity of product item can be changed and corresponding cost is also changed.

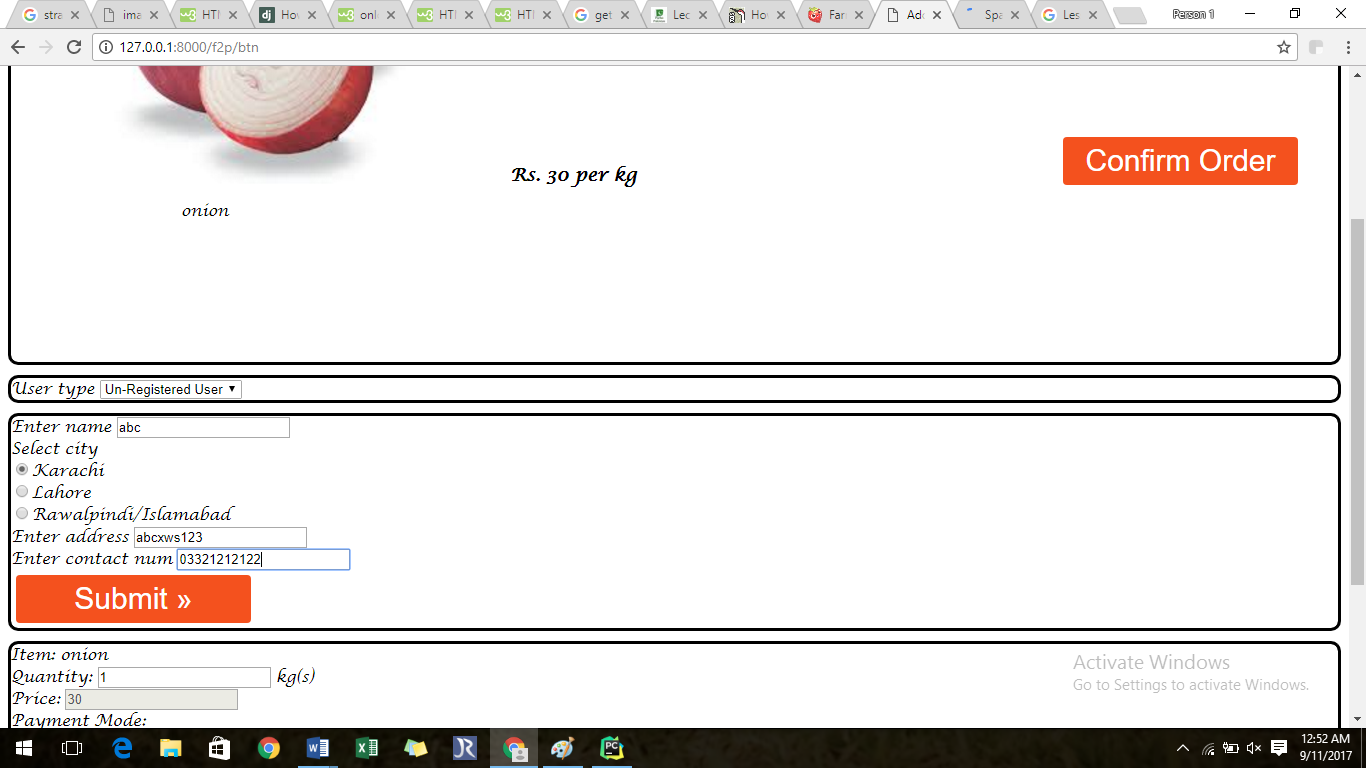
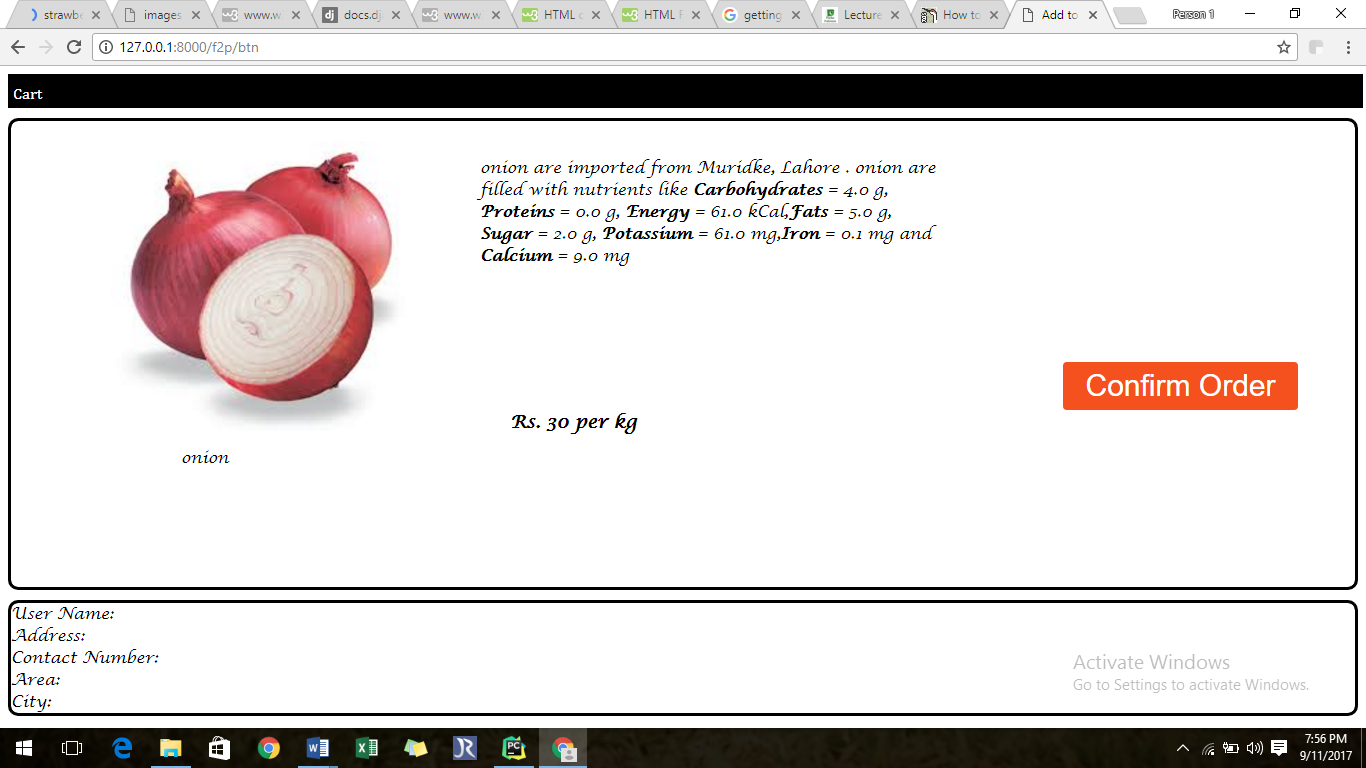


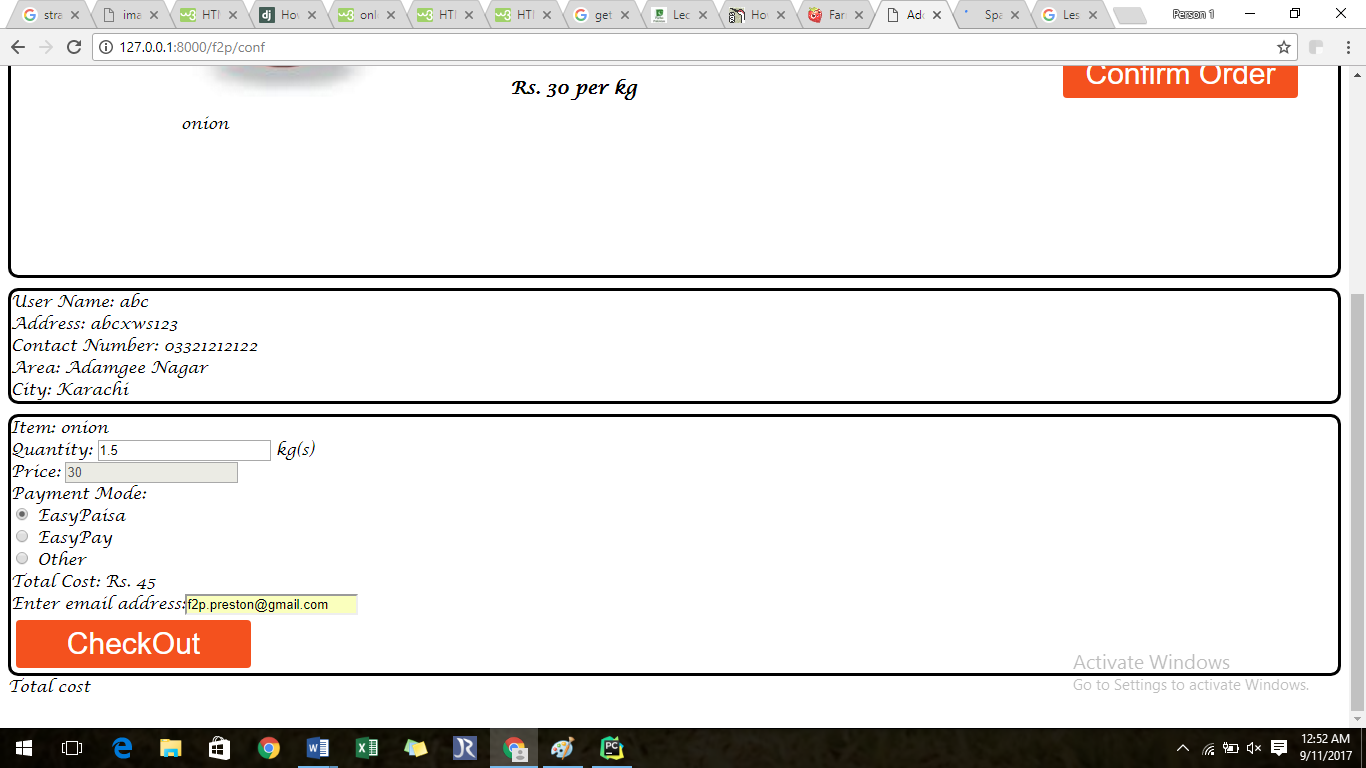
After **checking out**, the receipt has been sent as email to the specified address.



The same procedure is followed for the un-registered user. After providing the details, user become registered for the system for future buying process.

The same options are provided for the **vegetables**. 

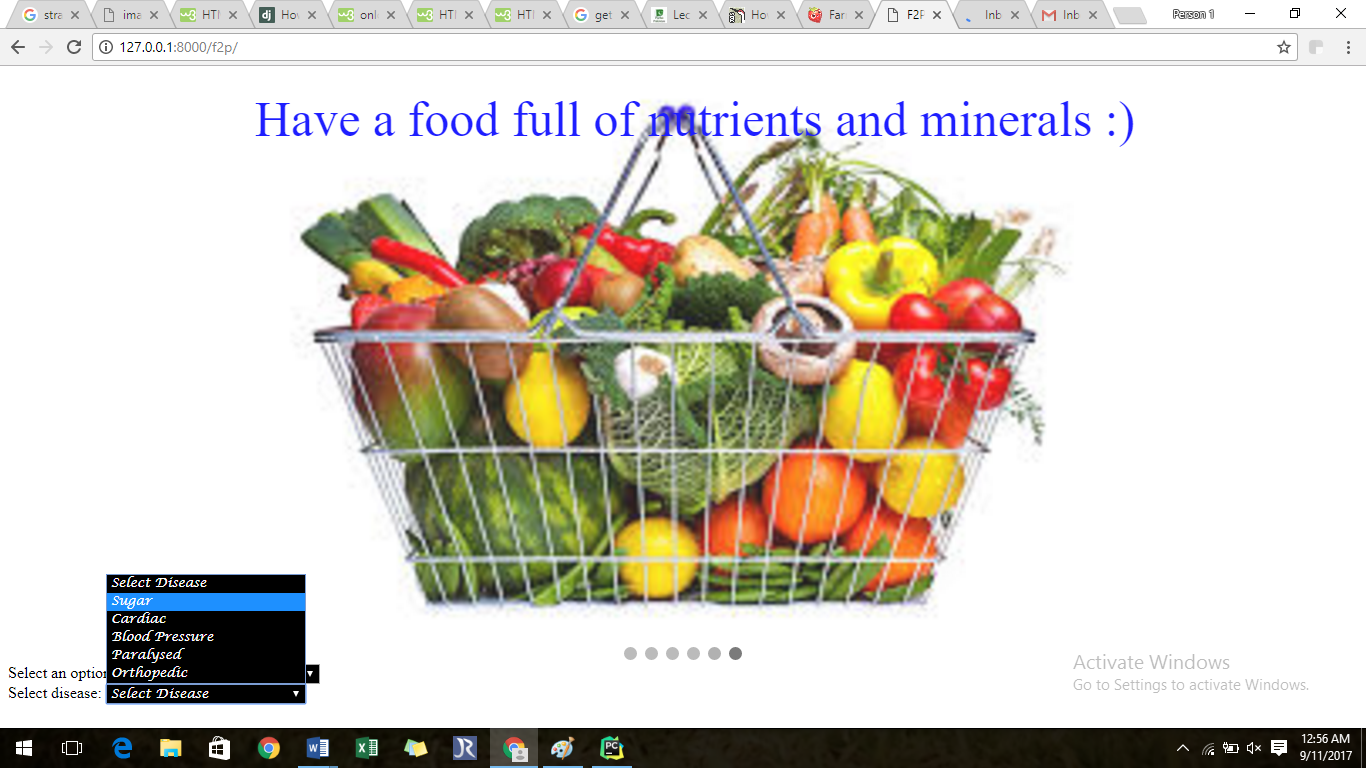




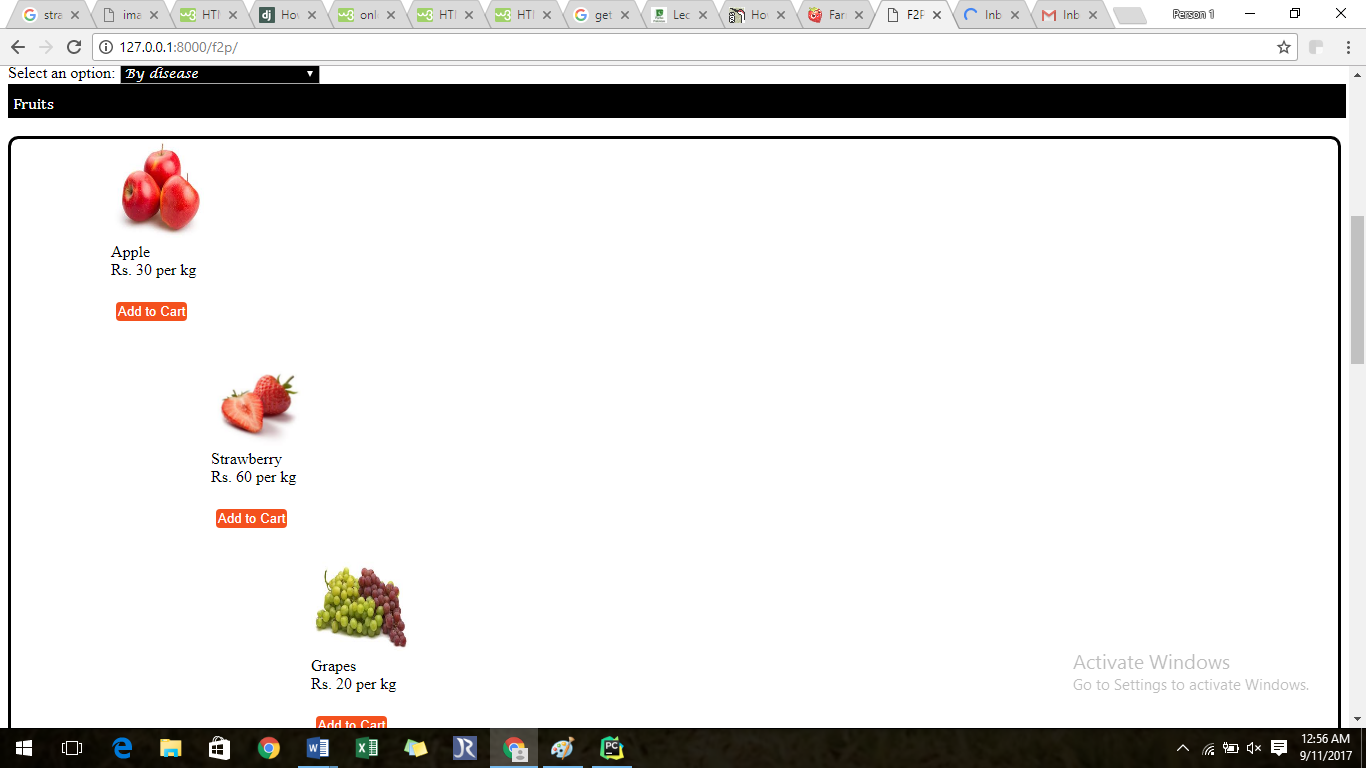
Search by Disease

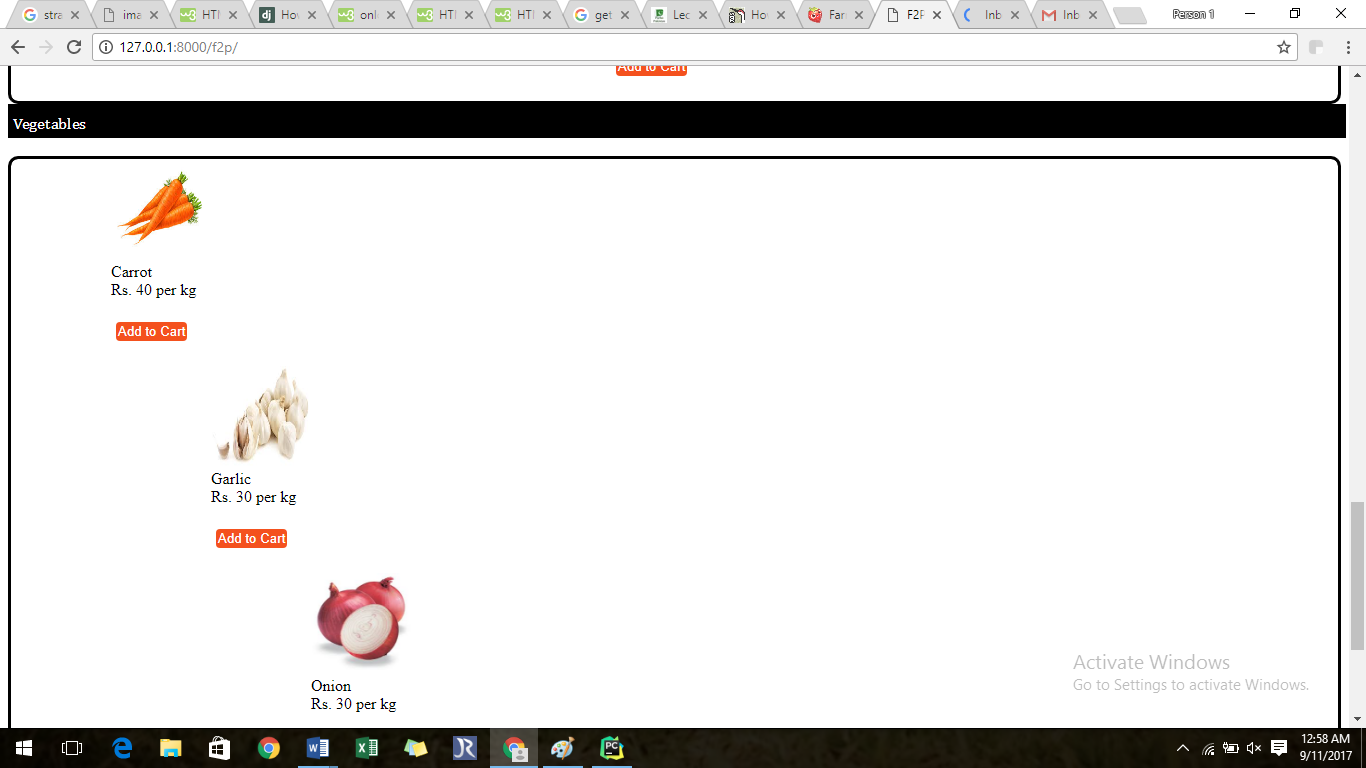


Scope for search by disease is limited upto five diseases.

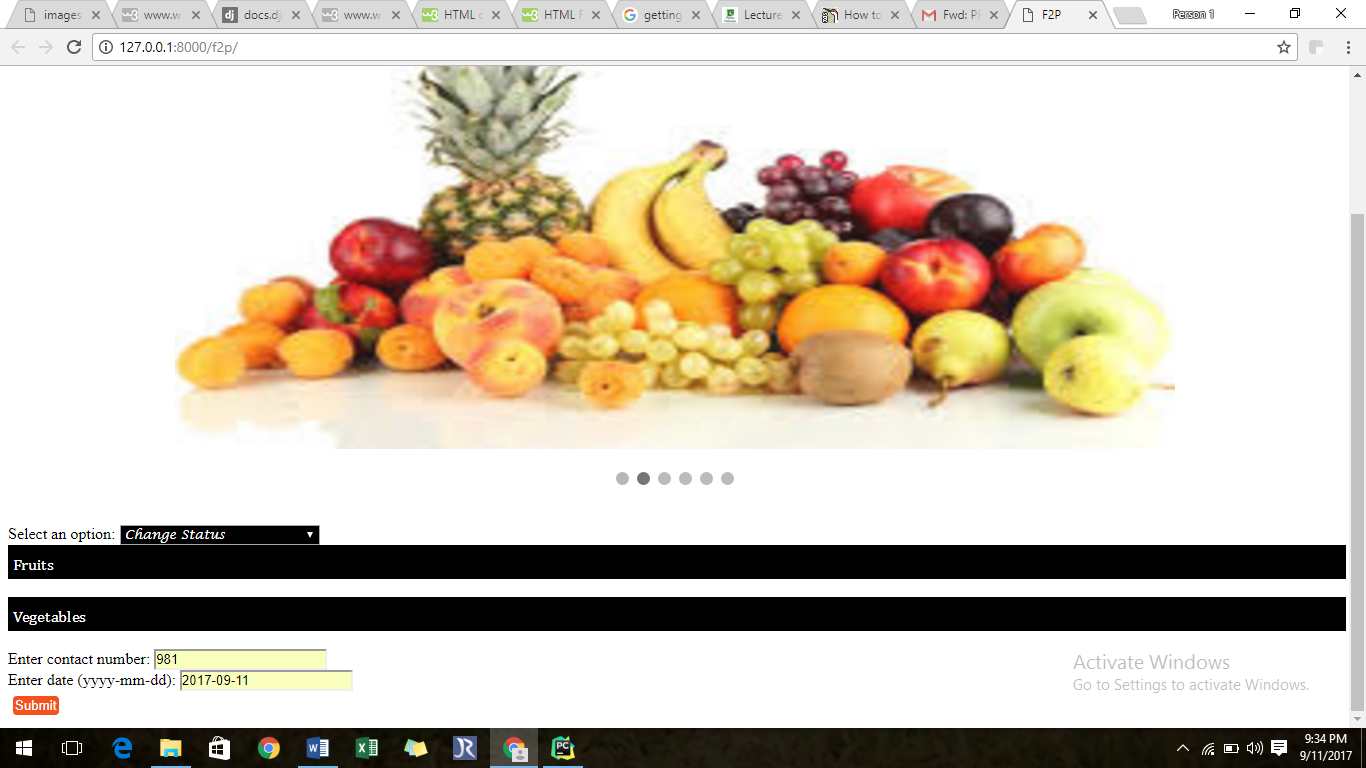


Recommended fruits and vegetables for the selected disease are shown. Here is the snap shot of food items suitable for **diabetic patients**.





Status of the order can be changed by entering date of order and contact number. This should be done to indicate that order has been delivered successfully. This will set **status as off**.



Code

Embedded and inline stylings styles have been used to style webpage. Code for styling the HTML webpages is given below.

<**style**>  
 .**button** {  
 **display**: **inline-block**;  
 **border-radius**: 4**px**;  
 **background-color**: **#f4511e**;  
 **border**: **none**;  
 **color**: **#FFFFFF**;  
 **text-align**: **center**;  
  
 **padding**: 2**px**;  
  
 **transition**: **all** 0.5**s**;  
 **cursor**:**pointer**;  
 **margin**: 5**px**;  
}  
  
.**button span** {  
 **cursor**: **pointer**;  
 **display**: **inline-block**;  
 **position**: **relative**;  
 **transition**: 0.5**s**;  
}  
  
.**button span**:**after** {  
 **content**: **'\00bb'**;  
 **position**: **absolute**;  
 **opacity**: 0;  
 **top**: 0;  
 **right**: -20**px**;  
 **transition**: 0.5**s**;  
}  
  
.**button**:**hover span** {  
 **padding-right**: 25**px**;  
}  
  
.**button**:**hover span**:**after** {  
 **opacity**: 1;  
 **right**: 0;  
}  
 **#hover** {  
 **display**: **none**;  
}  
  
**#image**:**hover** + **#hover** {  
 **display**: **block**;  
 **background-color**: **#A5A5A5**;  
 **color**: **white**;  
 **width**:350**px**;  
 **padding-left**: 20**px**;  
 **padding-right**: 20**px**;  
 **border-radius**: 5**px**;  
 **opacity**: 2.0;  
  
}  
  
**#bg**{  
 **opacity**: .1;  
  
  
}  
 .**overlay** {  
 **position**: **absolute**;  
 **bottom**: 0;  
 **left**: 100%;  
 **right**: 0;  
 **background-color**: **#008CBA**;  
 **overflow**: **hidden**;  
 **width**: 0;  
 **height**: 100%;  
 **transition**: .5**s ease**;  
}  
**#image**:**hover** + **#hover** {  
 **display**: **block**;  
}  
 .**mySlides** {**display**:**none**}  
  
*/\* Slideshow container \*/*.**slideshow-container** {  
 **max-width**: 1000**px**;  
 **position**: **relative**;  
 **margin**: **auto**;  
}  
  
*/\* Caption text \*/*.**text** {  
 **color**: **#0000FF**;  
 **font-size**: 50**px**;  
 **padding**: 8**px** 12**px**;  
 **position**: **absolute**;  
 **top**: 8**px**;  
 **width**: 100%;  
 **text-align**: **center**;  
}  
  
*/\* Number text (1/3 etc) \*/*.**numbertext** {  
 **color**: **#f2f2f2**;  
 **font-size**: 12**px**;  
 **padding**: 8**px** 12**px**;  
 **position**: **absolute**;  
 **top**: 0;  
}  
  
*/\* The dots/bullets/indicators \*/*.**dot** {  
 **height**: 13**px**;  
 **width**: 13**px**;  
 **margin**: 0 2**px**;  
 **background-color**: **#bbb**;  
 **border-radius**: 50%;  
 **display**: **inline-block**;  
 **transition**: **background-color** 2.2**s ease**;  
}  
  
.**active** {  
 **background-color**: **#717171**;  
}  
  
*/\* Fading animation \*/*.**fade** {  
 **-webkit-animation-name**: **fade**;  
 **-webkit-animation-duration**: 2.5**s**;  
 **animation-name**: **fade**;  
 **animation-duration**: 2.5**s**;  
}  
  
**@-webkit-keyframes fade** {  
 **from** {**opacity**: .4}  
 **to** {**opacity**: 1}  
}  
  
**@keyframes fade** {  
 **from** {**opacity**: .4}  
 **to** {**opacity**: 1}  
}  
  
*/\* On smaller screens, decrease text size \*/***@media only screen and** (**max-width**: 300**px**) {  
 .**text** {**font-size**: 11**px**}  
}  
.**lblHead**{  
 **background-color**: **black**;  
 **color**: **white**;  
 **padding-bottom**: 5**px**;  
 **width**:100%;  
 **padding-top**: 10**px**;  
 **font-family**: **Cambria**;  
 **padding-left**: 5**px**;  
  
}  
 </**style**>

HTML code for slider image for page **index.html** is given below.

<**div class="slideshow-container"**>  
  
<**div class="mySlides fade"**>  
*{# <div class="numbertext">1 / 3</div>#}* <**img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSV0dFEEoxkju\_ZyZPlhIhV\_SKc3lB8NP\_koC1bOfXTVPEx0ZyJ" style="width**:100%;**height**:550**px**;**"**>  
 <**div class="text"**>Farm to Palm</**div**>  
</**div**>  
  
<**div class="mySlides fade"**>  
*{# <div class="numbertext">2 / 3</div>#}* <**img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTQJRPV4mPm4EyEU1k4Kh77pXcGWsIeyi7NSJwTsGGz--7iKspx\_Q" style="width**:100%;**height**:550**px**;**"**>  
 <**div class="text"**>An easy way to access fresh fruits and vegetables at your door step</**div**>  
</**div**>  
  
<**div class="mySlides fade"**>  
*{# <div class="numbertext">3 / 3</div>#}* <**img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcR\_2EMWulAVm3BGykjQ1Wf5KdyAhYkTM6y5U1MoTH6baRBsbt\_L" style="width**:100%;**height**:550**px**;**"**>  
 <**div class="text"**>Have a food full of nutrients and minerals :)</**div**>  
</**div**>  
  
</**div**>

Server side code to send email:

**def** sendEmail(request,msg):  
 s = smtplib.SMTP(host=**'smtp.gmail.com'**, port=587)  
 s.starttls()  
 s.login(MY\_ADDRESS, PASSWORD)  
 s.sendmail(MY\_ADDRESS,request.session[**'email'**],msg)  
 s.quit()

Back end Development

For rendering the **fruits** webpage, the following server function is called.

**def** fruits(request):  
 fruit=Fruits.objects.all()  
 nutrients=nutrientsFruits.objects.all()  
 rc = RequestContext(request, {**'veges'**: fruit, **'nutrients'**: nutrients})  
 **return** render(request, **'f2p/fruits.html'**, {**'rc'**: rc})

This function gets all entries from **fruits** and **nutrientsFruits** model and have been sent as dictionary to the webpage.

**def** sendEmail(request,msg):  
 s = smtplib.SMTP(host=**'smtp.gmail.com'**, port=587)  
 s.starttls()  
 s.login(MY\_ADDRESS, PASSWORD)  
 s.sendmail(MY\_ADDRESS,request.session[**'email'**],msg)  
 s.quit()

The above mentioned piece of code is responsible to send emails to the specified address.

Testing Process

Each and every feature has been tested individually. The following test cases are given below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Case ID | Test Case Name | Description | Input | Expected output | Actual Output | Status |
|  | | | | | | |
| 1 | Selecting option | This test case checks whether the option are correctly selected or not. | Selection of options | Selected | Selected | PASS |
| 2 | Add to Cart | This test case is responsible for adding products in cart | Click on the button | Navigation | Navigated | PASS |
| 3 | Registering user | This test case checks whether the user has been registered or not | Enter details and click on button | Registered | Registered | PASS |
| 4 | View of farmer details | This test case is responsible to check whether the farmer details are shown or not. | Hover the product image | Details should be shown | Shown | PASS |
| 5 | Send email | This test case checks whether the system send receipt as email. | Click on Check Out button | Email sent | Sent | PASS |

**Chapter 5: Conclusion and Future Work**

In this document, **Farm to Palm** project has been described which is the e-commerce platform. This platform provides a user friendly and responsive interface to the user that allows him to purchase vegetables and fruits. The innovative feature is the provision of search of fruits and vegetables according to the disease. Our platform has solved the problem of searching food products according to their diseases. Consumers have been facilitating consumers to buy high quality and nutritionist products.

User can select to search according to disease or without disease. The desired fruit and vegetable can be added into the virtual cart and this can be checked out after entering posting address details. Un-registered users must enter details whereas the details of registered users have been provided by the system. The detailed description of products, i.e., farmer introduction, product’s nutrients and price has been provided by the system.

In this project, we have limited our scope to five diseases. This can be increased to fulfill the needs of maximum consumers. Moreover, the farmer circle can be enhanced who are trust-worthy in this field.