SMARTFARM

Submitted in partial fulfillment of the requirements for degree in
BACHELOR OF ENGINEERING IN
INFORMATION TECHNOLOGY

By

Joel Dsilva (5019116)
Aryan Koul (5019129)
Mithun Kuthully(5019131)

Supervisor

Prof. Suraj Khandare

Department of Information Technology

Fr. C. Rodrigues Institute of Technology

Vashi, Navi Mumbai – 400703

University of Mumbai

(AY 2020-2021)

CERTIFICATE

This is to certify that the Mini Project entitled **"SMARTFARM"** is a bonafide work of **Joel Dsilva (5019116), Aryan Koul (5019129) and Mithun Kuthully(5019131)** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering"** in **"Information Technology".**

(Prof._____)

Supervisor

(Prof._____)

Head of Department

(Prof._____)

Principal

Mini Project Approval

This Mini Project entitled " **SMARTFARM**" by **Joel Dsilva (5019116), Aryan Koul (5019129) and Mithun Kuthully(5019131)** is approved for the degree of **Bachelor of Engineering** in **Information Technology.**

Examiners

1……
(Internal Examiner Name & Sign)

2……
(External Examiner name & Sign)

Date:

Place:

CONTENTS

ABSTRACT

SmartFarm is a Python based application which gives an idea to the farmers how to use internet to sell their products. Farmers will get all the new ideas to improve their productivity and they can buy and sell their products online with getting a better profit percentage as they were getting before.

We are building a website which will help Indian farmers to make the effective cultivation by providing up-to-date information and make a path to earn more money from Indian villages by selling their products to different cities online.

SmartFarm will also help them to do one-to-one business by removing the middle men, which will gradually decrease the price of items and will benefit the common people too.

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of this project would be incomplete without the mention of the people who made it possible, without whose constant guidance and encouragement would have made efforts go in vain. I consider myself privileged to express gratitude and respect towards all those who have guided us through the completion of this project.

I convey thanks to my project guide **Prof. Suraj Khandare** of Information Technology Department for providing encouragement, constant support and guidance which was of a great help to complete this project successfully.

List of Abbreviations

GUI – GRAPHICAL USER INTERFACE

SQL - STRUCTURED QUERY LANGUAGE

h/w – HARDWARE

s/w – SOFTWARE

RAM – RANDOM ACCESS MEMORY

IDE – INTEGRATED DEVELOPMENT ENVIRONMENT

LIST OF FIGURES

1. INTRODUCTION

1.1 Introduction

The integration of Information Technology (IT) into farming industries has lagged in comparison to other sectors of the economy. There has been a slow adoption rate to utilizing the internet and integrating IT into business operations and transactions. However, in recent years farmers have proved they do have the confidence to leave the old ways behind and grab the technology bull by the horns as such. The broadening variety and availability of specialist farming technologies has allowed for increased efficiency in production and sales areas, and increasing adoption of existing technologies such as the internet has allowed for the more efficient management of information and lower transaction costs. For a country as agriculturally focused as INDIA, the application of IT into business processes not only positively affects the farmers themselves but the economy as a whole.

1.2 Motivation:

The incorporation of IT into farming involves the integration of many different technologies, with each positively impacting the efficiency in how farms conduct themselves. This includes adoption of the internet, information systems and management, GPS technologies and automating various activities through the use of IT. Traditional methods of farming incorporated little to no use of IT and some farms still stand by this today, despite the potential for dramatically increasing operations efficiency. With agriculture arguably being the most traditional industry, it is easy to see how the use of IT in conducting business is lagging in comparison to other industries. However, with the increasing availability of specialist IT equipment coupled with increasing marketing for these products, farmers are becoming more aware of the benefits that can be gained utilizing IT.

The E-farming allows the farmers to globalize their products. It gives training to farmer who does not have knowledge of basics of computer. It provides facility of scheduling classes for farmers who enrolled for basic courses. Sites are also available in their local languages as per states. Online sales and purchase details of both farmers and wholesales are should maintain in secured way. Report generation features is provided using to generate different kind of reports which are helpful to knowing information of sales and purchases.

1.3 Problem Statement and Objectives:

Problem Statement:

India's food production & productivity is declining whereas its food consumption is increasing. There is lack of provision in timely and adequate inputs such as fertilizers, seeds, pesticides and by making available affordable agricultural credit /crop insurance for increasing the production and ensuring the food and nutritional security to the Nation. Marketing of the agriculture is not satisfactory. No one is providing fair prices to farmers for their crops. Supply channel bottlenecks and lack of market understanding.

Objectives:

The main objective of this project is to build an interface which will help Indian farmers to make the effective cultivation by providing up-to-date information.

It will make a path to earn more money from Indian villages by selling their products to different cities online

On the other side, wholesaler from town can also register and buy products as per their needs.

1.4 Organization of the Report:

The material in this report is organized into 2 chapters. After the introductory chapter, the report consists of Literature Survey and Analaysis and Proposed System.

Literature Survey and Analysis presents a brief summary on the existing works and projects done previously on our topic. It also gives a brief description about papers and articles related to the subject. It states the limitations and the shortcomings of the existing projects and applications. It also summarizes the measures taken in this project to overcome these limitations.

The next part is the Proposed System. It can be further divided into 4 sections. The first part deals with the architectural framework and the algorithm followed in our project. The second part contains the sample code of our project. The third part contains the results of our project. Finally, the last part concludes the topic with future works that can be implemented to improvise the application according to the needs.

2. LITERATURE SURVEY

2.1 Survey of Existing System:

In paper "Krishi-Bharati: An Interface for Indian Farmer" studied that Nowadays, advancement of ICT make possible to retrieve almost any information from the global repository (internet). Farmers require information at the right stage of life cycle of farming to take right decision. Due to illiteracy they cannot get information. This paper states that user can interact with the system through agricultural information in Indian language text and spoken forms both. After selecting the icons, the icon to natural language generation module convert the selected icons to text in Indian language. In paper "Icon Based Information Retrieval and Disease Identification in Agriculture" Most of farmer are illiterate that's why they are not able to use internet for possible remedies of their infected crops. This paper discusses mainly two features one with an iconic interface where farmer can interact easily and in return system will return in native language. Another feature is an image processing technique in that farmer has to upload image of diseased crops and result will show disease name and possible solution for infected crop. In paper "Enhancement in Agro Expert System for Rice Crop" Some farmers don't have enough knowledge to identify exact diseases on crop by analyzing symptom on crop. The main point of study in system is that system background starts with by analyzing the number of disease symptoms of the rice plant appearing during the life cycle of plant and then the collected knowledge viewed to develop an expert system. In paper "A Model for Enhancing Empowerment in Farmers is using Mobile Based information system" states that farmers which are living in villages rural areas do not have proper access of information to make decisions related to farming, they use mobile phones to communicate using internet. It provides personalized information with the aim of empowering them to make appropriate decision and actions.

2.2 Limitations of existing system or research gap:

There is no computerized system for the farmer to directly sell their product. Currently, the farmer goes to nearest market and handovers his product to a particular consumer; the consumer then asks the farmer to visit the market after a specific time to collect the cash earned out of the sold product. Consumer sells the product to another consumer or a dealer at the cost of that market. Every Consumer tries to cuts his commission out of that. There is no way for farmer to know about the deal and the exact amount at which their product was sold.

There is no transparency. No facility is present for the farmers to know the product rates at different markets where they can sell their products for achieving high profits.
Many times, farmers are not even aware of the schemes and compensation provided by government. In spite of all the opportunities banging the doors the farmers are not able to benefit out of those. Current system does not provide the way of e-learning for farmer that will provide the knowledge of new techniques in farming. So he doesn't get the maximum profit through the current system.

BigBasket, Grofers, ZopNow and Amazon Pantry are some of the existing applications through which the farmers sell their products. But these applications require middlemen for selling the products. While, our application doesn't require any middlemen and the farmers can directly sell their products.

1. Maximum number of farmers should use the application.
2. Advanced techniques are not used to check the authorization.
3. Farmers cannot directly upload their products in this application.
4. Lack of privacy as we do not have high encryption for secure online transaction or to protect online identity.

2.3 Mini Project Contribution:

The development of this new system contains the following activities, which try to automate the entire process and produce awareness among the formers to globalize their products.

- ✓ User friendliness is provided in the application with various controls provided by system rich user interface.
- ✓ Authentication is provided for this application; only registered users can access transaction details.
- ✓ Online sales and purchase details of both farmers and wholesales are maintained in a secured way.
- ✓ Our system is more accurate and is better than the current existing applications.
  .

To implement the above goals, the following methodology needs to be followed:

1 Account Generation: It includes the creation of account, in which basic information of user, type of user, whether he is a farmer or a consumer is submitted. Through this module, user gets the Unique ID which serves as the identity of user.

2 Marketing: It includes Pricing, Billing and the Fund Transfer. Pricing will show the farmer at what price his commodity has been sold. Billing will create the bill after getting request from farmer for bill creation. Created bill will be displayed on the page. Bill will consist of unit price rate, total bill amount, commission of consumer, vehicle fare, other expenditure, etc. Farmer can download or print the bill for future reference. Using fund transfer, Consumer can transfer the invoice amount to farmers account and farmer can check whether amount has been transferred or not. One should be log in for using this facility.

3 Market Information: Farmer can see the market information of nearby market. This will consist of selling rates of different product, today's turnover, product-wise details like quantity, grading, selling cost, etc. It will give commodity-wise, market-wise daily report, commodity wise price during last week, community transaction below MSP(maximum sale price), date wise prices for specified community. Farmer can also search for specific product in particular duration of specific market.

4 Government Schemes: It lists all government schemes related to particular product and area and can apply in the same way as for compensation.

5 E-Learning:. It will educate farmers about new trends and techniques for farming. User can view as well as download the content.

Gantt Chart:

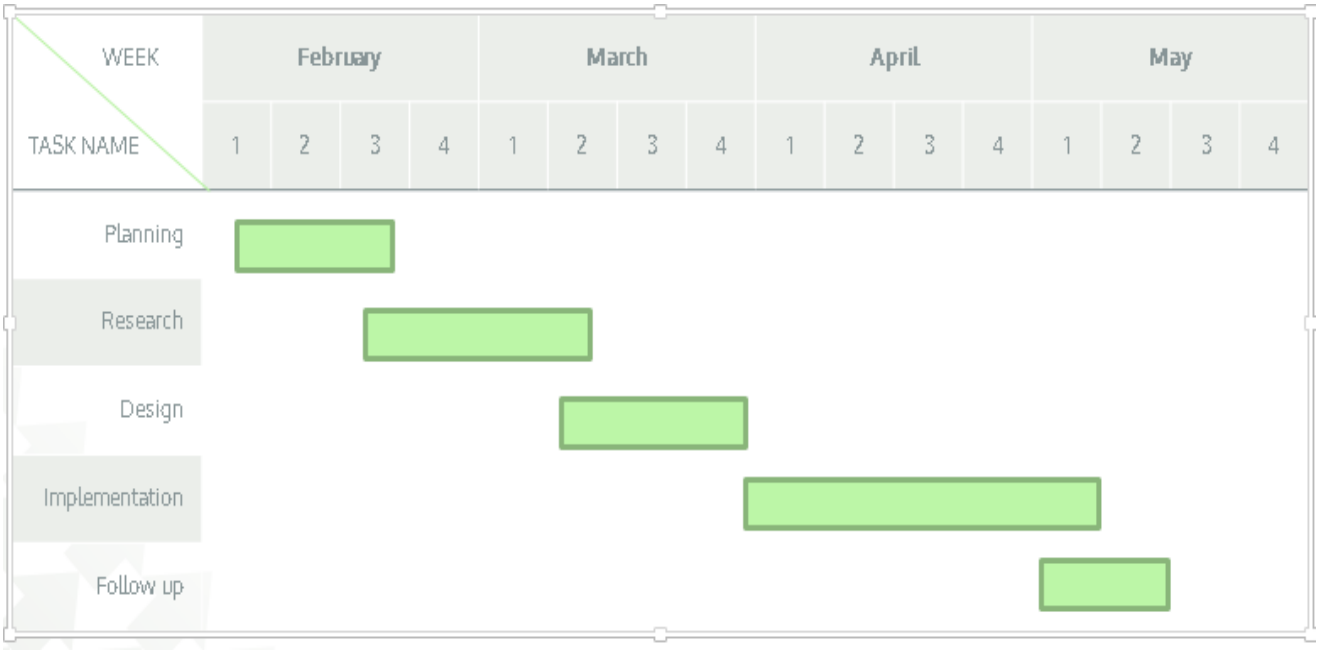


Fig 1) Gantt Chart

3. PROPOSED SYSTEM

3.1 Introduction:

Smartfarm can be described as the integration and utilization of IT in farming related operations. The requirement for each farming initiatives came about because “farming is an information rich activity”. The best way to manage information is through the use of IT, hence the foundation of Smartfarm.

The Smartfarm allows the farmers to globalize their products. It gives online sales and purchase details of both farmers and wholesales are should maintain in secured way. Report generation features is provided using to generate different kind of reports which are helpful to knowing information of sales and purchases.

3.2 Architecture/ Framework:

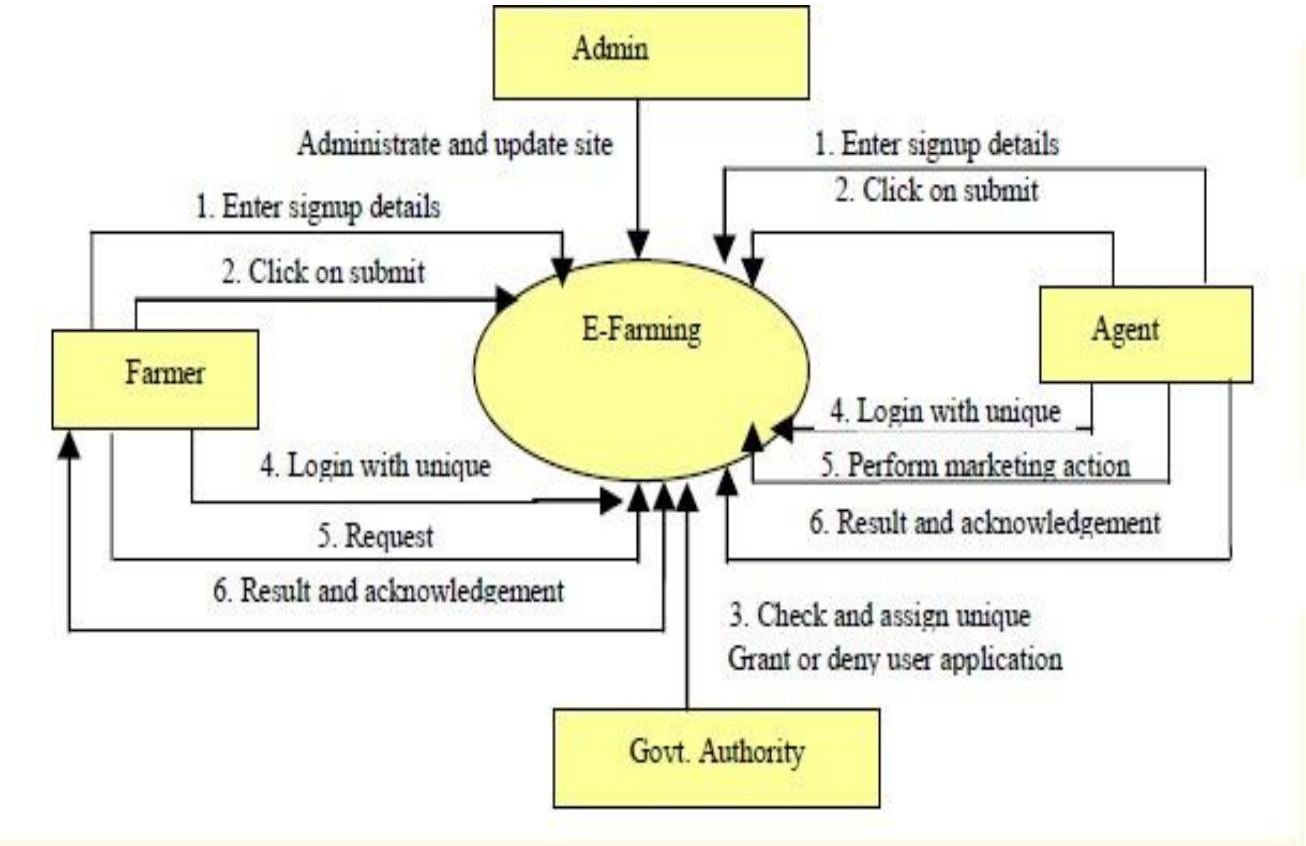


Fig 2) Architecture/Framework

3.3 Algorithm and Process Design:

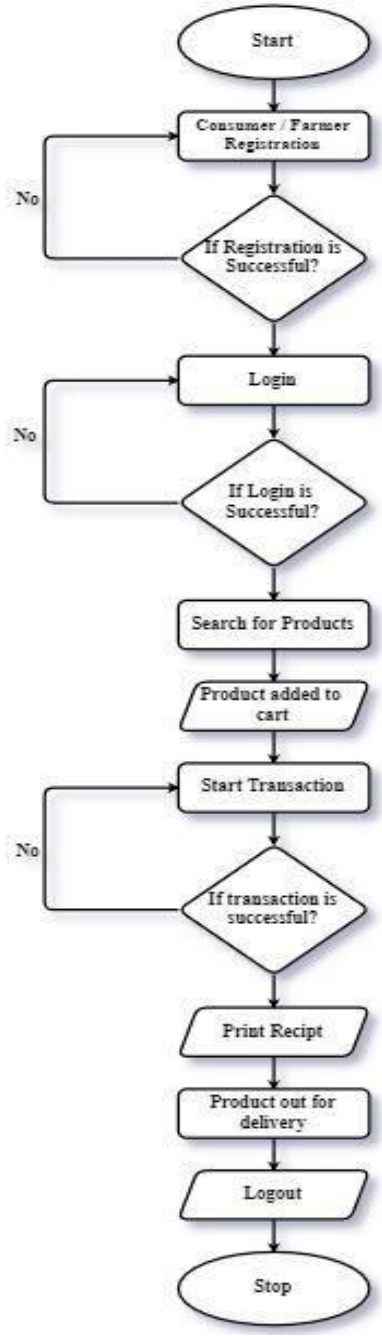


Fig 3) Algorithm/Process Design

3.4 Details of Hardware and Software

Software:

Operating System	:	Windows XP/2003 or Linux
User Interface	:	HTML, CSS
Client-side Scripting	:	JavaScript
Programming Language	:	Python
Web Applications	:	Django
IDE/Workbench	:	Visual Studio Code
Database	:	SQL,Xampp
Server Deployment	:	Tomcat Apache

Hardware:

Processor	:	Pentium IV
Hard Disk	:	40GB
RAM	:	512MB or more

3.5 Experiment And Results

3.5.1 CODE SAMPLE

- **Mysql code :**

```
--
-- Table structure for table `company`
--

CREATE TABLE `company` (
  `company_id` int(11) NOT NULL,
  `company_name` varchar(255) NOT NULL,
  `company_description` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `company`
--

INSERT INTO `company` (`company_id`, `company_name`, `company_description`) VALUES
(1, ‘Del Monte’, ‘’),
(2, ‘ParleAgro’, ‘’),
(3, ‘Tata’, ‘’),
(4, ‘BPB’, ‘’);

--------------------------------------------------------------

--
-- Table structure for table `django_migrations`
--

CREATE TABLE `django_migrations` (
  `id` int(11) NOT NULL,
  `app` varchar(255) NOT NULL,
  `name` varchar(255) NOT NULL,
  `applied` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--------------------------------------------------------------

--
-- Table structure for table `django_session`
--

CREATE TABLE `django_session` (
  `session_key` varchar(40) NOT NULL,
```

```
  `session_data` longtext NOT NULL,
  `expire_date` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `django_session`
--

INSERT INTO `django_session` (`session_key`, `session_data`, `expire_date`) VALUES
('1yszxy97tw2qhvxtkfzcipmqmxg5901o',
'OTJjZGRkY2E5ZjhhZDYwOTczNTQxZDlkNTJhNTYzNGY5ZDRjMDgwMjp7InVzZXJfaW
QiOjgsInVzZXJfbGV2ZwxfaWQiOjEsImF1dGhlbnRpY2F0ZWQiOnRydWUsInVzZXJfbmFtZ
SI6IkFtaXQgS3VtYXIifQ==', '2018-01-26 07:59:01'),
('2fwz2k18x3bllb08adu1e30cl9s61k3o',
'YjgzNDlkYWU2N2YyNjYwZTUyZGExOTZiNzlhYjg3NzM0ZmU2MDBmZDp7Im9yZGVy
X2lkIjoxMywidXNlcl9pZCI6ZmFsc2UsInVzZXJfbGV2ZWxfaWQiOmZhbHNlLCJhdXRoZW5
0aWNhdGVkIjpmYWxzZSwidXNlcl9uYW1lIjpmYWxzZX0=', '2019-10-19 07:33:36'),
('7ks0ebm8w26vswny9c0wfojyd5asv43i',
'NDE1ODNmMjY1ZjNlZDA2Y2ExYzc1ZGU5NWEyNGEzN2IzMWY2OGVjYTp7Im9yZGV
yX2lkIjoiMCIsImF1dGhlbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2xldmVsX2lkIjpmYWxzZS
widXNlcl9pZCI6ZmFsc2UsInVzZXJfbmFtZSI6ZmFsc2V9', '2019-05-25 15:49:49'),
('9bgzvan3fd73sdzjqy4fy90dxwzgmltg',
'MDZiNTU1MGVjZDFkNDliNDc3ZWY1OGExZDgwOTk5MWFkYTZjZmE3NDp7ImF1dGh
lbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2xldmVsX2lkIjpmYWxzZSwidXNlcl9pZCI6ZmFsc2
UsInVzZXJfbmFtZSI6ZmFsc2V9', '2018-01-29 14:09:38'),
('aavgs0xeieisikczpdeuragboqu5wddm',
'ODVlZmIxODczMzc4NWY2YjZhMWJmMGQxMzI0NjYzOWM0ZTFmYmVjZTp7Im9yZG
VyX2lkIjoxMiwidXNlcl9pZCI6ZmFsc2UsInVzZXJfbGV2ZWxfaWQiOmZhbHNlLCJhdXRoZ
W50aWNhdGVkIjpmYWxzZSwidXNlcl9uYW1lIjpmYWxzZX0=', '2019-07-19 10:39:59'),
('c7f2yysow67qjgtrgzabr8rx8eyvdnji',
'MDZiNTU1MGVjZDFkNDliNDc3ZWY1OGExZDgwOTk5MWFkYTZjZmE3NDp7ImF1dGh
lbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2xldmVsX2lkIjpmYWxzZSwidXNlcl9pZCI6ZmFsc2
UsInVzZXJfbmFtZSI6ZmFsc2V9', '2018-01-29 14:19:42'),
('ebqsosvupih3d6rfcy220w6eeoopoqt8',
'NDE1ODNmMjY1ZjNlZDA2Y2ExYzc1ZGU5NWEyNGEzN2IzMWY2OGVjYTp7Im9yZGV
yX2lkIjoiMCIsImF1dGhlbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2xldmVsX2lkIjpmYWxzZS
widXNlcl9pZCI6ZmFsc2UsInVzZXJfbmFtZSI6ZmFsc2V9', '2018-02-21 10:22:08'),
('eqny6tpfjj5p36yu9okbr7s61swwy0sk',
'OGI2N2UxYzlmN2YwMDdlNTQxYjhhMmUwYzZkMzJiOGEwNjE1ZmM0NDp7Im9yZGVy
X2lkIjo5LCJhdXRoZW50aWNhdGVkIjp0cnVlLCJ1c2VyX2xldmVsX2lkIjoyLCJ1c2VyX2lkIjo
xMSwidXNlcl9uYW1lIjoiQW1hbiBLdW1hciJ9', '2019-07-10 09:39:25'),
('f7vkj1ssawqqjkp470wbgzmqf8pnpuun',
'NDE1ODNmMjY1ZjNlZDA2Y2ExYzc1ZGU5NWEyNGEzN2IzMWY2OGVjYTp7Im9yZGV
yX2lkIjoiMCIsImF1dGhlbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2xldmVsX2lkIjpmYWxzZS
widXNlcl9pZCI6ZmFsc2UsInVzZXJfbmFtZSI6ZmFsc2V9', '2019-05-29 17:53:16'),
('j1unuxzc2z846m0r1xmkioa3xd63spfg',
'ODFkZmU0YjE3MzI5ODQ5NzQyNzc4Nzc0ODNjZjlkYTlhZWEwMmMxOTp7InVzZXJfaW
QiOjI1LCJ1c2VyX2xldmVsX2lkIjoyLCJhdXRoZW50aWNhdGVkIjp0cnVlLCJ1c2VyX25hbW
UiOiJLYXVzaGFsIEtpc2hvcmUifQ==', '2018-02-21 09:19:01'),
```

(‘pm9ifc6usfn38cwfcpuget8cu0g48c3k’,
‘OTJjZGRkY2E5ZjhhZDYwOTczNTQxZDlkNTJhNTYzNGY5ZDRjMDgwMjp7InVzZXJfaW
QiOjgsInVzZXJfbGV2ZwxfaWQiOjEsImF1dGhlbnRpY2F0ZWQiOnRydWUsInVzZXJfbmFtZ
SI6IkFtaXQgS3VtYXIifQ==’, ‘2018-01-29 13:36:24’),
(‘qi4jui1wag7y5kjd3nal07b1h2jlc9ia’,
‘OTJjZGRkY2E5ZjhhZDYwOTczNTQxZDlkNTJhNTYzNGY5ZDRjMDgwMjp7InVzZXJfaW
QiOjgsInVzZXJfbGV2ZwxfaWQiOjEsImF1dGhlbnRpY2F0ZWQiOnRydWUsInVzZXJfbmFtZ
SI6IkFtaXQgS3VtYXIifQ==’, ‘2017-07-21 11:40:27’),
(‘tfndahzufxmpyuy7ko8x3nqp08spjwx5’,
‘YjQ3ZWE3MWQ2YTMwNjc4MWE0NTQzNmI0ZTk0MWUxNTE0MWQwMWQ4ZDp7ImF
1dGhlbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2lkIjpmYWxzZSwidXNlcl9sZXZlbF9pZCI6Zm
Fsc2UsInVzZXJfbmFtZSI6ZmFsc2UsIm9yZGVyX2lkIjoiMCJ9’, ‘2019-06-24 13:33:15’),
(‘vhvspuaom6ewud0zx7o3h3r60xxm32w2’,
‘YjQ3ZWE3MWQ2YTMwNjc4MWE0NTQzNmI0ZTk0MWUxNTE0MWQwMWQ4ZDp7ImF
1dGhlbnRpY2F0ZWQiOmZhbHNlLCJ1c2VyX2lkIjpmYWxzZSwidXNlcl9sZXZlbF9pZCI6Zm
Fsc2UsInVzZXJfbmFtZSI6ZmFsc2UsIm9yZGVyX2lkIjoiMCJ9’, ‘2019-06-24 13:49:12’),
(‘xqitzy5mm8bz740ja8unqi2yzmdyj7ed’,
‘MDMwNWRjNWZmMGI3MjYyOWE1ZDI2YjE1NWEzMjg3OWVkYzM3MjEwNDp7Im9yZ
GVyX2lkIjozNCwiYXV0aGVudGljYXRlZCI6ZmFsc2UsInVzZXJfbGV2ZWxfaWQiOmZhbH
NlLCJ1c2VyX2lkIjpmYWxzZSwidXNlcl9uYW1lIjpmYWxzZX0=’, ‘2018-02-20 13:24:04’),
(‘xurl12qi1iu47jd87pw8da016ma877mh’,
‘NzU0ZDkyODRlOTI3N2Q5YjQ4ZWFiZDhkY2MxNGI2ZDU2NDc1MdhmOTp7Im9yZGVy
X2lkIjoxMSwiYXV0aGVudGljYXRlZCI6dHJ1ZswidXNlcl9sZXZlbF9pZCI6MiwidXNlcl9pZC
I6MTEsInVzZXJfbmFtZSI6IkFtYW4gS3VtYXIifQ==’, ‘2019-07-11 03:47:56’);

-- ----------

--
-- Table structure for table `order`
--

```
CREATE TABLE `order` (
  `order_id` int(11) NOT NULL,
  `order_user_id` varchar(255) NOT NULL,
  `order_date` varchar(255) NOT NULL,
  `order_status` varchar(255) NOT NULL,
  `order_total` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--
-- Dumping data for table `order`
--

```
INSERT INTO `order` (`order_id`, `order_user_id`, `order_date`, `order_status`, `order_total`)
VALUES
(1, ‘25’, ‘01:49PM on February 06, 2018’, ‘2’, ‘0’),
(2, ‘25’, ‘01:50PM on February 06, 2018’, ‘5’, ‘0’),
(3, ‘25’, ‘04:06PM on February 06, 2018’, ‘5’, ‘0’),
(4, ‘25’, ‘09:19AM on February 07, 2018’, ‘5’, ‘0’),
```

```
(5, ‘11’, ‘03:46PM on May 11, 2019’, ‘5’, ‘0’),
(6, ‘11’, ‘05:45PM on May 15, 2019’, ‘1’, ‘0’),
(7, ‘11’, ‘07:02PM on June 10, 2019’, ‘1’, ‘0’),
(8, ‘11’, ‘07:17PM on June 10, 2019’, ‘1’, ‘0’),
(9, ‘11’, ‘09:21AM on June 21, 2019’, ‘1’, ‘0’),
(10, ‘11’, ‘03:44AM on June 27, 2019’, ‘1’, ‘0’),
(11, ‘11’, ‘03:47AM on June 27, 2019’, ‘1’, ‘0’),
(12, ‘11’, ‘10:25AM on July 05, 2019’, ‘1’, ‘0’);

-- --------------------------------------------------------

--
-- Table structure for table `order_item`
--

CREATE TABLE `order_item` (
  `oi_id` int(11) NOT NULL,
  `oi_order_id` varchar(255) NOT NULL,
  `oi_product_id` varchar(255) NOT NULL,
  `oi_price_per_unit` varchar(255) NOT NULL,
  `oi_cart_quantity` varchar(255) NOT NULL,
  `oi_total` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `order_item`
--

INSERT INTO `order_item` (`oi_id`, `oi_order_id`, `oi_product_id`, `oi_price_per_unit`,
`oi_cart_quantity`, `oi_total`) VALUES
(1, ‘1’, ‘7’, ‘1200’, ‘1’, ‘1200’),
(2, ‘1’, ‘8’, ‘1200’, ‘1’, ‘1200’),
(3, ‘2’, ‘6’, ‘1170’, ‘1’, ‘1170’),
(4, ‘2’, ‘2’, ‘1100’, ‘1’, ‘1100’),
(5, ‘2’, ‘3’, ‘1170’, ‘4’, ‘4680’),
(6, ‘3’, ‘8’, ‘1200’, ‘1’, ‘1200’),
(7, ‘4’, ‘1’, ‘1200’, ‘1’, ‘1200’),
(8, ‘4’, ‘5’, ‘1170’, ‘1’, ‘1170’),
(9, ‘4’, ‘7’, ‘1200’, ‘2’, ‘2400’),
(10, ‘4’, ‘9’, ‘3423’, ‘1’, ‘3423’),
(11, ‘5’, ‘4’, ‘1170’, ‘1’, ‘1170’),
(13, ‘5’, ‘1’, ‘1200’, ‘1’, ‘1200’),
(14, ‘5’, ‘9’, ‘3423’, ‘1’, ‘3423’),
(15, ‘6’, ‘1’, ‘1200’, ‘1’, ‘1200’),
(16, ‘6’, ‘5’, ‘1170’, ‘1’, ‘1170’),
(17, ‘6’, ‘3’, ‘1170’, ‘1’, ‘1170’),
(19, ‘7’, ‘5’, ‘1170’, ‘1’, ‘1170’),
(20, ‘7’, ‘6’, ‘1170’, ‘1’, ‘1170’),
(21, ‘8’, ‘1’, ‘1200’, ‘1’, ‘1200’),
```

```
(22, ‘8’, ‘9’, ‘3423’, ‘1’, ‘3423’),
(23, ‘8’, ‘4’, ‘1170’, ‘1’, ‘1170’),
(26, ‘9’, ‘1’, ‘120’, ‘2’, ‘240’),
(27, ‘9’, ‘2’, ‘110’, ‘1’, ‘110’),
(28, ‘9’, ‘2’, ‘110’, ‘1’, ‘110’),
(30, ‘10’, ‘6’, ‘117’, ‘1’, ‘117’),
(31, ‘10’, ‘10’, ‘3000’, ‘1’, ‘3000’),
(32, ‘11’, ‘5’, ‘350’, ‘1’, ‘350’),
(33, ‘12’, ‘1’, ‘1200’, ‘1’, ‘1200’),
(34, ‘13’, ‘1’, ‘1200’, ‘1’, ‘1200’),
(36, ‘13’, ‘5’, ‘350’, ‘1’, ‘350’);

-- --------------------------------------------------------

--
-- Table structure for table `order_status`
--

CREATE TABLE `order_status` (
  `os_id` int(11) NOT NULL,
  `os_title` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `order_status`
--

INSERT INTO `order_status` (`os_id`, `os_title`) VALUES
(1, ‘Confirmed’),
(2, ‘Processing’),
(3, ‘Packed’),
(4, ‘Dispatched’),
(5, ‘Cancelled’);

-- --------------------------------------------------------

--
-- Table structure for table `products_product`
--

CREATE TABLE `products_product` (
  `product_id` int(11) NOT NULL,
  `product_vendor_id` varchar(255) NOT NULL,
  `product_name` varchar(255) NOT NULL,
  `product_type_id` varchar(255) NOT NULL,
  `product_company_id` varchar(255) NOT NULL,
  `product_price` varchar(255) NOT NULL,
  `product_image` varchar(255) NOT NULL,
  `product_description` text NOT NULL,
```

```
  `product_stock` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;

--
-- Dumping data for table `products_product`
--

INSERT INTO `products_product` (`product_id`, `product_vendor_id`, `product_name`,
`product_type_id`, `product_company_id`, `product_price`, `product_image`,
`product_description`, `product_stock`) VALUES
(1, '1002', 'Kobyashi', '2', '1', '1200', '/uploads/kobyashi seeds.jpeg', 'Vegetable seeds are
those seeds which can be grown in gardens,privately or commercially. They grow to become
edible nutrients. Vegetable seeds have high protein content and are highly nutritious. Majority of
the human calories come from vegetable seeds. They are also used for making cooking oils and
food additives.They are also eaten by animals and are fed to livestock or provided as birdseed.',
'5'),
(2, '1002', 'Home Farming', '1', '1', '1100', '/uploads/home farming Seeds.jpeg', 'A seed is an
embryonic plant enclosed in a seed coat normally including some form of stored food. Usually
the seed is formed inside a fruit – which may look like a pod, husk, or cone. Flower seeds are of
many different sizes. They are planted in flower patches, which are mostly found alongside
allotment patches.', '91'),
(3, '1002', 'Gralic Seeds', '4', '2', '199', '/uploads/garlic seeds.jpeg', 'Organic seed are the seeds
which are organically grown. In other words, they are grown using sustainable methods from
start to finish. Organically grown seed produces hearty and robust plants. The seeds are cultivated
on land for at least three years without the use of chemical fertilizers and pesticides under the
standards established for Certified Organic Farming.', '199'),
(4, '1017', 'Seeds Medicine', '4', '1', '240', '/uploads/Insecticides Medicine.jpeg', 'Organic seed
are the seeds which are organically grown. In other words, they are grown using sustainable
methods from start to finish. Organically grown seed produces hearty and robust plants. The
seeds are cultivated on land for at least three years without the use of chemical fertilizers and
pesticides under the standards established for Certified Organic Farming.', '87'),
(5, '1017', 'Hybrid Seeds', '1', '3', '350', '/uploads/hybrid seeds.jpeg', 'A seed is an embryonic
plant enclosed in a seed coat normally including some form of stored food. Usually the seed is
formed inside a fruit – which may look like a pod, husk, or cone. Flower seeds are of many
different sizes. They are planted in flower patches, which are mostly found alongside allotment
patches.', '12'),
(6, '1018', 'Vegetable Seeds', '2', '1', '1170', '/uploads/vegetable seed.jpeg', 'Vegetable seeds
are those seeds which can be grown in gardens,privately or commercially. They grow to become
edible nutrients. Vegetable seeds have high protein content and are highly nutritious. Majority of
the human calories come from vegetable seeds. They are also used for making cooking oils and
food additives.They are also eaten by animals and are fed to livestock or provided as birdseed.',
'15'),
(7, '1018', 'Vitamin Seeds', '2', '3', '1200', '/uploads/Vitamin Seeds.jpeg', 'Vegetable seeds are
those seeds which can be grown in gardens,privately or commercially. They grow to become
edible nutrients. Vegetable seeds have high protein content and are highly nutritious. Majority of
the human calories come from vegetable seeds. They are also used for making cooking oils and
food additives.They are also eaten by animals and are fed to livestock or provided as birdseed.',
'97'),
```

(8, ‘1017’, ‘Jute Orgainc’, ‘1’, ‘2’, ‘250’, ‘/uploads/jute organic.jpeg’, ‘A seed is an embryonic plant enclosed in a seed coat normally including some form of stored food. Usually the seed is formed inside a fruit – which may look like a pod, husk, or cone. Flower seeds are of many different sizes. They are planted in flower patches, which are mostly found alongside allotment patches.’, ‘100’),
(9, ‘1001’, ‘Grain Seeds’, ‘3’, ‘2’, ‘342’, ‘/uploads/grain.jpeg’, ‘Grain seeds are grain-producing substances. The two main types of commercial grain seeds are cereals and legumes(pulses). Cereal grains contain a substantial amount of starch, while legumes or pulses contain high amount of protein. Because grain seeds are small, hard and dry, they can be stored, measured, and transported more readily than can other kinds of food crops such as fresh fruits, roots and tubers.’, ‘32’),
(10, ‘1018’, ‘Zukur Seeds’, ‘3’, ‘3’, ‘300’, ‘/uploads/zukur seeds.jpeg’, ‘Grain seeds are grain-producing substances. The two main types of commercial grain seeds are cereals and legumes(pulses). Cereal grains contain a substantial amount of starch, while legumes or pulses contain high amount of protein. Because grain seeds are small, hard and dry, they can be stored, measured, and transported more readily than can other kinds of food crops such as fresh fruits, roots and tubers.’, ‘23’),
(11, ‘1015’, ‘Tomatoes’, ‘6’, ‘1’, ‘20’, ‘/uploads/tomatoes.jpg’, ‘Vegetables are parts of plants that are consumed by humans or other animals as food. Vegetables can be eaten either raw or cooked and play an important role in human nutrition, being mostly low in fat and carbohydrates, but high in vitamins, minerals and dietary fiber. They are also an important source for essential vitamins, minerals, and trace elements.’, ‘100’),
(12, ‘1015’, ‘Potatoes’, ‘6’, ‘2’, ‘20’, ‘/uploads/potatoes.jpg’, ‘Vegetables are parts of plants that are consumed by humans or other animals as food. Vegetables can be eaten either raw or cooked and play an important role in human nutrition, being mostly low in fat and carbohydrates, but high in vitamins, minerals and dietary fiber. They are also an important source for essential vitamins, minerals, and trace elements.’, ‘100’),
(13, ‘1015’, ‘Onion’, ‘6’, ‘3’, ‘15’, ‘/uploads/onions.jpg’, ‘Vegetables are parts of plants that are consumed by humans or other animals as food. Vegetables can be eaten either raw or cooked and play an important role in human nutrition, being mostly low in fat and carbohydrates, but high in vitamins, minerals and dietary fiber. They are also an important source for essential vitamins, minerals, and trace elements.’, ‘100’),
(14, ‘1015’, ‘Drumsticks’, ‘6’, ‘4’, ‘240’, ‘/uploads/drumsticks.jpg’, ‘Vegetables are parts of plants that are consumed by humans or other animals as food. Vegetables can be eaten either raw or cooked and play an important role in human nutrition, being mostly low in fat and carbohydrates, but high in vitamins, minerals and dietary fiber. They are also an important source for essential vitamins, minerals, and trace elements.’, ‘100’),
(15, ‘1015’, ‘Carrots’, ‘6’, ‘1’, ‘50’, ‘/uploads/carrots.jpg’, ‘Vegetables are parts of plants that are consumed by humans or other animals as food. Vegetables can be eaten either raw or cooked and play an important role in human nutrition, being mostly low in fat and carbohydrates, but high in vitamins, minerals and dietary fiber. They are also an important source for essential vitamins, minerals, and trace elements.’, ‘100’),
(16, ‘1016’, ‘Apples’, ‘5’, ‘2’, ‘160’, ‘/uploads/apples.jpg’, ‘Fruits are the seed-bearing structure in flowering plants formed from the ovary after flowering. “Fruit” normally means the fleshy seed-associated structures of a plant that are sweet or sour, and edible in the raw state, such as apples, bananas, grapes, lemons, oranges, etc. Fruits have high fibre and vitamin content. Regular consumption of fruit is generally associated with reduced risks of several diseases and functional declines associated with aging. They are one of the most affordable sources for carbohydrates and proteins.’, ‘100’),

(17, ‘1016’, ‘Bananas’, ‘5’, ‘3’, ‘60’, ‘/uploads/bananas.jpg’, ‘Fruits are the seed-bearing structure in flowering plants formed from the ovary after flowering. “Fruit” normally means the fleshy seed-associated structures of a plant that are sweet or sour, and edible in the raw state, such as apples, bananas, grapes, lemons, oranges, etc. Fruits have high fibre and vitamin content. Regular consumption of fruit is generally associated with reduced risks of several diseases and functional declines associated with aging. They are one of the most affordable sources for carbohydrates and proteins.’, ‘100’),
(18, ‘1016’, ‘Strawberries’, ‘5’, ‘4’, ‘150’, ‘/uploads/strawberries.jpg’, ‘Fruits are the seed-bearing structure in flowering plants formed from the ovary after flowering. “Fruit” normally means the fleshy seed-associated structures of a plant that are sweet or sour, and edible in the raw state, such as apples, bananas, grapes, lemons, oranges, etc. Fruits have high fibre and vitamin content. Regular consumption of fruit is generally associated with reduced risks of several diseases and functional declines associated with aging. They are one of the most affordable sources for carbohydrates and proteins.’, ‘100’),
(19, ‘1016’, ‘Watermelons’, ‘5’, ‘1’, ‘100’, ‘/uploads/watermelons.jpg’, ‘Fruits are the seed-bearing structure in flowering plants formed from the ovary after flowering. “Fruit” normally means the fleshy seed-associated structures of a plant that are sweet or sour, and edible in the raw state, such as apples, bananas, grapes, lemons, oranges, etc. Fruits have high fibre and vitamin content. Regular consumption of fruit is generally associated with reduced risks of several diseases and functional declines associated with aging. They are one of the most affordable sources for carbohydrates and proteins.’, ‘100’),
(20, ‘1016’, ‘Pomeogranates’, ‘5’, ‘2’, ‘120’, ‘/uploads/pomegranates.jpg’, ‘Fruits are the seed-bearing structure in flowering plants formed from the ovary after flowering. “Fruit” normally means the fleshy seed-associated structures of a plant that are sweet or sour, and edible in the raw state, such as apples, bananas, grapes, lemons, oranges, etc. Fruits have high fibre and vitamin content. Regular consumption of fruit is generally associated with reduced risks of several diseases and functional declines associated with aging. They are one of the most affordable sources for carbohydrates and proteins.’, ‘100’),
(21, ‘1019’, ‘Rice’, ‘7’, ‘3’, ‘140’, ‘/uploads/rice.jpg’, ‘A grain is a small, hard, dry seed, with or without an attached hull or fruit layer, harvested for human or animal consumption. There are two types of grains namely whole grains and refined grains. Grains are well suited for industrial agriculture. After being harvested, dry grains are more durable than other staple foods.’, ‘100’),
(22, ‘1019’, ‘Wheat’, ‘7’, ‘4’, ‘100’, ‘/uploads/wheat.jpg’, ‘A grain is a small, hard, dry seed, with or without an attached hull or fruit layer, harvested for human or animal consumption. There are two types of grains namely whole grains and refined grains. Grains are well suited for industrial agriculture. After being harvested, dry grains are more durable than other staple foods.’, ‘100’),
(23, ‘1019’, ‘Jowar’, ‘7’, ‘1’, ‘125’, ‘/uploads/jowar.jpg’, ‘A grain is a small, hard, dry seed, with or without an attached hull or fruit layer, harvested for human or animal consumption. There are two types of grains namely whole grains and refined grains. Grains are well suited for industrial agriculture. After being harvested, dry grains are more durable than other staple foods.’, ‘100’),
(24, ‘1019’, ‘Bajra’, ‘7’, ‘2’, ‘450’, ‘/uploads/bajra.jpg’, ‘A grain is a small, hard, dry seed, with or without an attached hull or fruit layer, harvested for human or animal consumption. There are two types of grains namely whole grains and refined grains. Grains are well suited for industrial agriculture. After being harvested, dry grains are more durable than other staple foods.’, ‘100’),
(25, ‘1019’, ‘Ragi’, ‘7’, ‘3’, ‘75’, ‘/uploads/ragi.jpg’, ‘A grain is a small, hard, dry seed, with or without an attached hull or fruit layer, harvested for human or animal consumption. There are two types of grains namely whole grains and refined grains. Grains are well suited for industrial agriculture. After being harvested, dry grains are more durable than other staple foods.’, ‘100’),

(26, ‘1013’, ‘Chillie’, ‘8’, ‘4’, ‘50’, ‘/uploads/chillie.jpg’, ‘A spice is a seed, fruit, root, bark, or other plant substance primarily used for flavoring or coloring food. They are primarily used for food flavoring. They are also used for perfume cosmetics and incense. They contain high amount of fat, protein, carbohydrates and also some amount of minerals and micronutrients. Most herbs and spices have substantial antioxidant activity.’, ‘100’),
(27, ‘1013’, ‘Turmeric’, ‘8’, ‘1’, ‘50’, ‘/uploads/turmeric.jpg’, ‘A spice is a seed, fruit, root, bark, or other plant substance primarily used for flavoring or coloring food. They are primarily used for food flavoring. They are also used for perfume cosmetics and incense. They contain high amount of fat, protein, carbohydrates and also some amount of minerals and micronutrients. Most herbs and spices have substantial antioxidant activity.’, ‘100’),
(28, ‘1013’, ‘Pepper’, ‘8’, ‘2’, ‘50’, ‘/uploads/pepper.jpg’, ‘A spice is a seed, fruit, root, bark, or other plant substance primarily used for flavoring or coloring food. They are primarily used for food flavoring. They are also used for perfume cosmetics and incense. They contain high amount of fat, protein, carbohydrates and also some amount of minerals and micronutrients. Most herbs and spices have substantial antioxidant activity.’, ‘100’),
(29, ‘1013’, ‘Garam Masala’, ‘8’, ‘3’, ‘50’, ‘/uploads/garam.jpg’, ‘A spice is a seed, fruit, root, bark, or other plant substance primarily used for flavoring or coloring food. They are primarily used for food flavoring. They are also used for perfume cosmetics and incense. They contain high amount of fat, protein, carbohydrates and also some amount of minerals and micronutrients. Most herbs and spices have substantial antioxidant activity.’, ‘100’),
(30, ‘1013’, ‘Star Anise’, ‘8’, ‘4’, ‘100’, ‘/uploads/star-anise.jpg’, ‘A spice is a seed, fruit, root, bark, or other plant substance primarily used for flavoring or coloring food. They are primarily used for food flavoring. They are also used for perfume cosmetics and incense. They contain high amount of fat, protein, carbohydrates and also some amount of minerals and micronutrients. Most herbs and spices have substantial antioxidant activity.’, ‘100’);

-- ----------

--
-- Table structure for table `type`
--

```
CREATE TABLE `type` (
  `type_id` int(11) NOT NULL,
  `type_name` varchar(255) NOT NULL,
  `type_description` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

--
-- Dumping data for table `type`
--

INSERT INTO `type` (`type_id`, `type_name`, `type_description`) VALUES
(1, ‘Flower Seeds’, ‘Computer’),
(2, ‘Vegetable Seeds’, ‘Maths’),
(3, ‘Grain Seeds’, ‘CBSE’),
(4, ‘Organic Seeds’, ‘UGC’),
(5, ‘Fruits’, ‘Science’),
(6, ‘Vegetables’, ‘English’),

```
(7, ‘Grains’, ‘IEEE’),
(8, ‘Spices\r\n’, ‘Btech’);

-- --------------------------------------------------------

--
-- Table structure for table `users_city`
--

CREATE TABLE `users_city` (
  `city_id` int(11) NOT NULL,
  `city_name` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;

--
-- Dumping data for table `users_city`
--

INSERT INTO `users_city` (`city_id`, `city_name`) VALUES
(1, ‘Mumbai’),
(2, ‘Thane’),
(3, ‘Chandhigarh’),
(4, ‘Amritsar’),
(5, ‘Kolkata’),
(6, ‘Guwahati’),
(7, ‘Kalyan’),
(8, ‘Pune’),
(9, ‘Thrivanthapuram’),
(10, ‘Jaiput’),
(11, ‘Allahabad’),
(12, ‘Varanasi’),
(13, ‘Jammu’),
(14, ‘Kashmir’),
(15, ‘Udhampur’);

-- --------------------------------------------------------

--
-- Table structure for table `users_country`
--

CREATE TABLE `users_country` (
  `country_id` int(11) NOT NULL,
  `country_name` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;

--
-- Dumping data for table `users_country`
--
```

```
INSERT INTO `users_country` (`country_id`, `country_name`) VALUES
(1, ‘India’),
(2, ‘Norway’),
(3, ‘Canada’),
(4, ‘Japan’),
(5, ‘Switzerland’),
(6, ‘Bulgaria’),
(7, ‘Russia’),
(8, ‘Spain’),
(9, ‘Italy’),
(10, ‘USA’);

-- --------------------------------------------------------

--
-- Table structure for table `users_role`
--

CREATE TABLE `users_role` (
  `role_id` int(11) NOT NULL,
  `role_title` varchar(255) NOT NULL,
  `role_description` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `users_role`
--

INSERT INTO `users_role` (`role_id`, `role_title`, `role_description`) VALUES
(1, ‘System Admin’, ‘Admin Roles and Permissions’),
(2, ‘Normal User’, ‘Users Roles and Permissions’),
(3, ‘Doctor’, ‘Doctors User Permission and Role’),
(4, ‘Patient’, ‘Patient User Permission and Role’);

-- --------------------------------------------------------

--
-- Table structure for table `users_state`
--

CREATE TABLE `users_state` (
  `state_id` int(11) NOT NULL,
  `state_name` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Dumping data for table `users_state`
--
```

```
INSERT INTO `users_state` (`state_id`, `state_name`) VALUES
(1, ‘Uttar Pradesh’),
(2, ‘Arunachal Pradesh’),
(3, ‘Goa’),
(4, ‘Karnataka’),
(5, ‘Maharashtra’),
(6, ‘Kerala’),
(7, ‘Chattisghar’),
(8, ‘J&K’),
(9, ‘Punjab’),
(10, ‘Madhya Pradesh’);

-- --------------------------------------------------------------

--
-- Table structure for table `users_user`
--

CREATE TABLE `users_user` (
  `user_id` int(11) NOT NULL,
  `user_level_id` int(11) NOT NULL DEFAULT ‘2’,
  `user_username` varchar(255) NOT NULL,
  `user_password` varchar(255) NOT NULL,
  `user_name` varchar(255) NOT NULL,
  `user_add1` varchar(255) NOT NULL,
  `user_guy` varchar(255) NOT NULL,
  `user_city` int(11) NOT NULL,
  `user_state` int(11) NOT NULL,
  `user_country` int(11) NOT NULL,
  `user_email` varchar(255) NOT NULL,
  `user_mobile` varchar(255) NOT NULL,
  `user_gender` varchar(255) NOT NULL,
  `user_dob` varchar(255) NOT NULL,
  `user_image` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 ROW_FORMAT=COMPACT;

--
-- Dumping data for table `users_user`
--

INSERT INTO `users_user` (`user_id`, `user_level_id`, `user_username`, `user_password`,
`user_name`, `user_add1`, `user_guy`, `user_city`, `user_state`, `user_country`, `user_email`,
`user_mobile`, `user_gender`, `user_dob`, `user_image`) VALUES
(11, 2, ‘customer’, ‘test’, ‘Aryan Koul’, ‘House No : 355’, ‘Farmer’, 1, 2, 1, ‘aryan@gmail.com’,
‘987654321’, ‘Male’, ‘18 January,1968’, ‘/uploads/p3.jpg’),
(14, 2, ‘kaushal’, ‘test’, ‘Joel Dsilva’, ‘House No : 355’, ‘Consumer’, 1, 1, 2, ‘joel@gmail.com’,
‘987654321’, ‘Male’, ‘16 January,2001’, ‘/uploads/p5.jpg’);
```

```
--
-- Indexes for dumped tables
--

--
-- Indexes for table `company`
--
ALTER TABLE `company`
  ADD PRIMARY KEY (`company_id`);

--
-- Indexes for table `django_migrations`
--
ALTER TABLE `django_migrations`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `django_session`
--
ALTER TABLE `django_session`
  ADD PRIMARY KEY (`session_key`),
  ADD KEY `django_session_expire_date_a5c62663` (`expire_date`);

--
-- Indexes for table `order`
--
ALTER TABLE `order`
  ADD PRIMARY KEY (`order_id`);

--
-- Indexes for table `order_item`
--
ALTER TABLE `order_item`
  ADD PRIMARY KEY (`oi_id`);

--
-- Indexes for table `order_status`
--
ALTER TABLE `order_status`
  ADD PRIMARY KEY (`os_id`);

--
-- Indexes for table `products_product`
--
ALTER TABLE `products_product`
  ADD PRIMARY KEY (`product_id`);

--
-- Indexes for table `type`
```

```
--
ALTER TABLE `type`
  ADD PRIMARY KEY (`type_id`);

--
-- Indexes for table `users_city`
--
ALTER TABLE `users_city`
  ADD PRIMARY KEY (`city_id`);

--
-- Indexes for table `users_country`
--
ALTER TABLE `users_country`
  ADD PRIMARY KEY (`country_id`);

--
-- Indexes for table `users_role`
--
ALTER TABLE `users_role`
  ADD PRIMARY KEY (`role_id`);

--
-- Indexes for table `users_state`
--
ALTER TABLE `users_state`
  ADD PRIMARY KEY (`state_id`);

--
-- Indexes for table `users_user`
--
ALTER TABLE `users_user`
  ADD PRIMARY KEY (`user_id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `company`
--
ALTER TABLE `company`
  MODIFY `company_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;

--
-- AUTO_INCREMENT for table `django_migrations`
--
ALTER TABLE `django_migrations`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT;
```

```
--
-- AUTO_INCREMENT for table `order`
--
ALTER TABLE `order`
  MODIFY `order_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=14;

--
-- AUTO_INCREMENT for table `order_item`
--
ALTER TABLE `order_item`
  MODIFY `oi_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=37;

--
-- AUTO_INCREMENT for table `order_status`
--
ALTER TABLE `order_status`
  MODIFY `os_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;

--
-- AUTO_INCREMENT for table `products_product`
--
ALTER TABLE `products_product`
  MODIFY `product_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;

--
-- AUTO_INCREMENT for table `type`
--
ALTER TABLE `type`
  MODIFY `type_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=6;

--
-- AUTO_INCREMENT for table `users_city`
--
ALTER TABLE `users_city`
  MODIFY `city_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `users_country`
--
ALTER TABLE `users_country`
  MODIFY `country_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `users_role`
--
ALTER TABLE `users_role`
  MODIFY `role_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
```

```
--
-- AUTO_INCREMENT for table `users_state`
--
ALTER TABLE `users_state`
  MODIFY `state_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=3;

--
-- AUTO_INCREMENT for table `users_user`
--
ALTER TABLE `users_user`
  MODIFY `user_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=15;
COMMIT;
```

- **Python code:**

1) **Users-**

```
from django.shortcuts import render, redirect
from django.http import HttpResponse
from django.conf import settings
from django.db.models import Q
from django.core.files.storage import FileSystemStorage
from .models import role, user, city, state, country
from django.contrib import messages

# Create your views here.
Def index(request):
    if(request.session.get('authenticated', False) == True):
        return redirect('/users/report')

    context = {
        "message": "Please Log in",
        "error": False
    }

    if(request.method == "POST"):
        try:
            getUser = user.objects.get(user_username=request.POST['username'])
            context['msg'] = getUser
        except:
            context['message'] = "Wrong username"
            context['error'] = True
            return render(request,'login.html', context)
        if(getUser.user_password == request.POST['password']):
            request.session['authenticated'] = True
            request.session['user_id'] = getUser.user_id
            request.session['user_level_id'] = getUser.user_level_id
            request.session['user_name'] = getUser.user_name
```

```
            return redirect('/users/dashboard')
        else:
            context['message'] = "Wrong Password"
            context['error'] = True
            return render(request,'login.html', context)
    else:
        return render(request,'login.html', context)

def listing(request, userId):
    if(request.session.get('authenticated', False) != True):
        return redirect('/')
    try:
        userlist = user.objects.filter(Q(user_level_id=userId))
    except Exception as e:
        return HttpResponse('Something went wrong. Error Message : '+ str€)

    context = {
        "showmsg": True,
        "message": "User Updated Successfully",
        "userlist": userlist
    }
    # Message according Users Role #
    if(userId == "1"):
        context['heading'] = "System Admin Report";
    if(userId == "2"):
        context['heading'] = "Normal User Report";
    if(userId == "3"):
        context['heading'] = "Doctors Report";
    if(userId == "4"):
        context['heading'] = "Patient Report";
    return render(request,'user-report.html',context)

def dashboard(request):
    return render(request,'dashboard.html')

def forgot(request):
    return render(request,'forgotpass.html')

def update(request, userId):
    context = {
    "fn":"update",
    "citylist":city.objects.all(),
    "statelist":state.objects.all(),
    "rolelist":role.objects.all(),
    "countrylist":country.objects.all(),
    "userdetails":user.objects.get(user_id=userId)
    }
    currentUserDetails = user.objects.get(user_id = userId)
    context['sub_heading'] = "Update Details of"+currentUserDetails.user_name;
```

```
    # Message according Users Role #
    if(currentUserDetails.user_level_id == 1):
        context['heading'] = "System Admin Management";
    if(currentUserDetails.user_level_id == 2):
        context['heading'] = "Normal User Management";
    if(currentUserDetails.user_level_id == 3):
        context['heading'] = "Doctors Management";
    if(currentUserDetails.user_level_id == 4):
        context['heading'] = "Patient Management";

    if (request.method == "POST"):
        try:
            user_image = None
            user_image = currentUserDetails.user_image
            if(request.FILES and request.FILES['user_image']):
                userImage = request.FILES['user_image']
                fs = FileSystemStorage()
                filename = fs.save(userImage.name, userImage)
                user_image = fs.url(userImage)

            addUser = user(
            user_id = userId,
            user_name = request.POST['user_name'],
            user_username = request.POST['user_username'],
            user_email = request.POST['user_email'],
            user_password = request.POST['user_password'],
            user_mobile = request.POST['user_mobile'],
            user_gender = request.POST['user_gender'],
            user_dob = request.POST['user_dob'],
            user_add1 = request.POST['user_add1'],
            user_guy = request.POST['user_guy'],
            user_city = request.POST['user_city'],
            user_country = request.POST['user_country'],
            user_state = request.POST['user_state'],
            user_image = user_image)

            addUser.save()
        except Exception as e:
            return HttpResponse('Something went wrong. Error Message : '+ str€)

        if (request.session.get('user_level_id', None) == 1):
            messages.add_message(request, messages.INFO, "User Updated Successfully
!!!")
            return redirect('/users/report/'+request.POST['user_level_id'])

        context["userdetails"] = user.objects.get(user_id=userId)
        messages.add_message(request, messages.INFO, "Your Account Updated
Successfully !!!")
        return render(request,'user.html', context)
```

```
    else:
        return render(request,'user.html', context)

def add(request):
    context = {
    "fn":"add",
    "citylist":city.objects.all(),
    "rolelist":role.objects.all(),
    "heading":'Users Management',
    "sub_heading": 'Users',
    "statelist":state.objects.all(),
    "countrylist":country.objects.all()
    }
    context['heading'] = "Customer Registration";
    context['sub_heading'] = "Register to Account";
    if (request.method == "POST"):
        try:
            user_image = None

            if(request.FILES and request.FILES['user_image']):
                userImage = request.FILES['user_image']
                fs = FileSystemStorage()
                filename = fs.save(userImage.name, userImage)
                user_image = fs.url(userImage)

            addUser = user(user_name = request.POST['user_name'],
            user_username = request.POST['user_username'],
            user_email = request.POST['user_email'],
            user_password = request.POST['user_password'],
            user_mobile = request.POST['user_mobile'],
            user_gender = request.POST['user_gender'],
            user_dob = request.POST['user_dob'],
            user_add1 = request.POST['user_add1'],
            user_guy = request.POST['user_guy'],
            user_city = request.POST['user_city'],
            user_country = request.POST['user_country'],
            user_state = request.POST['user_state'],
            user_image = user_image)
            addUser.save()
        except Exception as e:
            return HttpResponse('Something went wrong. Error Message : '+ str€)
        messages.add_message(request, messages.INFO, "Your account has been registered
successfully. Login with your credentials !!!")
        return redirect('/users')

    else:
        return render(request,'user.html', context)

def logout(request):
```

```
    request.session['authenticated']= False
    request.session['user_id'] = False
    request.session['user_level_id']= False
    request.session['user_name']= False
    return redirect('/')

def changepassword(request):
    if (request.method == "POST"):
        try:
            addUser = user(
                user_id = request.session.get('user_id', None),
                user_password = request.POST['user_new_password']
            )
            addUser.save(update_fields=["user_password"])
        except Exception as e:
            return HttpResponse('Something went wrong. Error Message : '+ str€)
        messages.add_message(request, messages.INFO, "Your Password has been changed
successfully !!!")
        return render(request,'change-password.html')

    else:
        return render(request,'change-password.html')
    return render(request,'change-password.html')

def delete(request, userId):
    try:
        deleteUser = user.objects.get(user_id = userId)
        deleteUser.delete()
    except Exception as e:
        return HttpResponse('Something went wrong. Error Message : '+ str€)
    messages.add_message(request, messages.INFO, "User Deleted Successfully !!!")
    return redirect('listing')
```

2) **Products-**

```
from django.shortcuts import render, redirect
from django.http import HttpResponse
from django.conf import settings
from django.db.models import Q
from django.core.files.storage import FileSystemStorage
from .models import product
from django.contrib import messages
from django.db import connection
from online_book_store.utils import getDropDown, dictfetchall
import datetime

# Create your views here.
Def orderlisting(request):
```

```
    cursor = connection.cursor()
    if (request.session.get('user_level_id', None) == 1):
        SQL = "SELECT * FROM `order`,`users_user`,`order_status` WHERE order_status
= os_id AND order_user_id = user_id"
    else:
        customerID = str(request.session.get('user_id', None))
        SQL = "SELECT * FROM `order`,`users_user`,`order_status` WHERE order_status
= os_id AND order_user_id = user_id AND user_id = " + customerID
    cursor.execute(SQL)
    orderlist = dictfetchall(cursor)

    context = {
        "orderlist": orderlist
    }

    # Message according Product #
    context['heading'] = "Order Reports";
    return render(request, 'order-listing.html', context)

# Create your views here.
Def productlisting(request):
    cursor = connection.cursor()
    cursor.execute(
        "SELECT * FROM products_product, company, type WHERE company_id =
product_company_id AND type_id = product_type_id")
    productlist = dictfetchall(cursor)

    context = {
        "productlist": productlist
    }

    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'products-listing.html', context)

# Create your views here.
Def payment(request):
    orderID = request.session.get('order_id', None);
    cursor = connection.cursor()
    cursor.execute("SELECT SUM(oi_total) as TotalCartValue FROM order_item
WHERE oi_order_id = " + str(orderID))
    orderTotal = dictfetchall(cursor)
    context = {
        "orderTotal": orderTotal[0]
    }
    if (request.method == "POST"):
        request.session['order_id'] = "0"
        return redirect('order-items/'+str(orderID))
    # Message according Product #
```

```
    context['heading'] = "Products Details";
    return render(request, 'payment.html', context)

# Create your views here.
Def cancel_order(request, orderID):
    cursor = connection.cursor()
    cursor.execute("""
            UPDATE `order`
            SET order_status= '5' WHERE order_id = %s
        """, (
      orderID
    ))
    messages.add_message(request, messages.INFO, "Your order has been cancelled
successfully !!!")
    return redirect('orderlisting')

# Create your views here.
Def order_items(request, orderID):
    cursor = connection.cursor()
    ### Get the Cart Details Listing ####
    cursor.execute("SELECT * FROM `products_product`, `order`, order_item, company,
type WHERE product_id = oi_product_id AND oi_order_id = order_id AND
company_id = product_company_id AND type_id = product_type_id AND order_id = "+
str(orderID))
    productlist = dictfetchall(cursor)

     ### Get the Cart Details Listing ####
    cursor.execute("SELECT * FROM `order`, `users_user`,`order_status` WHERE
order_status = os_id AND user_id = order_user_id AND order_id = "+ str(orderID))
    customerOrderDetails = dictfetchall(cursor)

    ### Get the Total Cart  ####
    cursor.execute("SELECT SUM(oi_total) as totalCartCost FROM `products_product`,
`order`, order_item, company, type WHERE product_id = oi_product_id AND
oi_order_id = order_id AND company_id = product_company_id AND type_id =
product_type_id AND order_id = "+ str(orderID))
    totalCost = dictfetchall(cursor)

    context = {
      "productlist": productlist,
      "customerOrderDetails": customerOrderDetails[0],
      "totalCost":totalCost[0]
    }

    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'order-items.html', context)

# Create your views here.
```

```
Def order_edit(request, orderID):
    cursor = connection.cursor()
    ### Get the Cart Details Listing ####
    cursor.execute("SELECT * FROM `products_product`, `order`, order_item, company,
type WHERE product_id = oi_product_id AND oi_order_id = order_id AND
company_id = product_company_id AND type_id = product_type_id AND order_id = "+
str(orderID))
    productlist = dictfetchall(cursor)

     ### Get the Cart Details Listing ####
    cursor.execute("SELECT * FROM `order`, `users_user`,`order_status` WHERE
order_status = os_id AND user_id = order_user_id AND order_id = "+ str(orderID))
    customerOrderDetails = dictfetchall(cursor)
    customerOrderDetails = customerOrderDetails[0]

    ### Get the Total Cart  ####
    cursor.execute("SELECT SUM(oi_total) as totalCartCost FROM `products_product`,
`order`, order_item, company, type WHERE product_id = oi_product_id AND
oi_order_id = order_id AND company_id = product_company_id AND type_id =
product_type_id AND order_id = "+ str(orderID))
    totalCost = dictfetchall(cursor)

    context = {
        "productlist": productlist,
        "protypelist":getDropDown('order_status', 'os_id', 'os_title',
customerOrderDetails['order_status'], '1'),
        "customerOrderDetails": customerOrderDetails,
        "totalCost":totalCost[0]
    }
    if (request.method == "POST"):
        cursor = connection.cursor()
        cursor.execute("""
                UPDATE `order`
                SET order_status= %s WHERE order_id = %s
            """, (
            request.POST['order_status'],
            request.POST['order_id']
        ))
        messages.add_message(request, messages.INFO, "Your order has been cancelled
successfully !!!")
        return redirect('orderlisting')
    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'order-edit.html', context)

# Create your views here.
Def cart_listing(request):
    orderID = request.session.get('order_id', None);
    cursor = connection.cursor()
```

```
    ### Get the Cart Details Listing ####
    cursor.execute("SELECT * FROM `products_product`, `order`, order_item, company,
type WHERE product_id = oi_product_id AND oi_order_id = order_id AND
company_id = product_company_id AND type_id = product_type_id AND order_id = "+
str(orderID))
    productlist = dictfetchall(cursor)

    ### Get the Total Cart  ####
    cursor.execute("SELECT SUM(oi_total) as totalCartCost FROM `products_product`,
`order`, order_item, company, type WHERE product_id = oi_product_id AND
oi_order_id = order_id AND company_id = product_company_id AND type_id =
product_type_id AND order_id = "+ str(orderID))
    totalCost = dictfetchall(cursor)

    context = {
        "productlist": productlist,
        "totalCost":totalCost[0]
    }

    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'carts.html', context)

# Create your views here.
Def products(request):
    cursor = connection.cursor()
    cursor.execute(
        "SELECT * FROM products_product, company, type WHERE company_id =
product_company_id AND type_id = product_type_id")
    productlist = dictfetchall(cursor)

    context = {
        "productlist": productlist
    }

    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'products.html', context)

# Create your views here.
Def product_filter(request, typeID):
    cursor = connection.cursor()
    cursor.execute(
        "SELECT * FROM products_product, company, type WHERE company_id =
product_company_id AND type_id = product_type_id AND type_id = "+ str(typeID))
    productlist = dictfetchall(cursor)

    context = {
        "productlist": productlist
```

```
    }

    # Message according Product #
    context['heading'] = "Products Details";
    return render(request, 'products.html', context)

def update(request, productId):
    productdetails = product.objects.get(product_id=productId)
    context = {
        "fn": "add",
        "procompanylist":getDropDown('company', 'company_id', 'company_name',
productdetails.product_company_id, '1'),
        "protypelist":getDropDown('type', 'type_id', 'type_name',
productdetails.product_type_id, '1'),
        "productdetails":productdetails
    }
    if (request.method == "POST"):
        try:
            product_image = None
            product_image = productdetails.product_image
            if(request.FILES and request.FILES['product_image']):
                productImage = request.FILES['product_image']
                fs = FileSystemStorage()
                filename = fs.save(productImage.name, productImage)
                product_image = fs.url(productImage)

            addProduct = product(
            product_id = productId,
            product_name = request.POST['product_name'],
            product_type_id = request.POST['product_type_id'],
            product_company_id = request.POST['product_company_id'],
            product_price = request.POST['product_price'],
            product_image = product_image,
            product_description = request.POST['product_description'],
            product_stock = request.POST['product_stock'])
            addProduct.save()
        except Exception as e:
            return HttpResponse('Something went wrong. Error Message : '+ str€)

        context["productdetails"] = product.objects.get(product_id = productId)
        messages.add_message(request, messages.INFO, "Product updated 41uccessfully
!!!")
        return redirect('productlisting')

    else:
        return render(request,'products-add.html', context)

def product_details(request, productId):
    if(request.session.get('authenticated', False) == False):
```

```
        messages.add_message(request, messages.ERROR, “Login to your account, to buy
the product !!!”)
        return redirect(‘/users’)
    cursor = connection.cursor()
    cursor.execute(
        “SELECT * FROM products_product, company, type WHERE company_id =
product_company_id AND type_id = product_type_id AND product_id = “+productId)
    productdetails = dictfetchall(cursor)

    context = {
        “fn”: “add”,
        “productdetails”:productdetails[0]
    }
    if (request.method == “POST”):
        try:
            if(request.session.get(‘order_id’, None) == “0” or request.session.get(‘order_id’,
False) == False):
                customerID = request.session.get(‘user_id’, None)
                orderDate = datetime.datetime.now().strftime(“%I:%M%p on %B %d, %Y”)
                cursor = connection.cursor()
                cursor.execute(“”””
                INSERT INTO `order`
                SET order_user_id=%s, order_date=%s, order_status=%s, order_total=%s
                “””, (
                    customerID,
                    orderDate,
                    1,
                    0))
                request.session[‘order_id’] = cursor.lastrowid

            orderID = request.session.get(‘order_id’, None);
            cursor = connection.cursor()
            totalAmount = int(request.POST[‘product_price’]) *
int(request.POST[‘product_quantity’]);
            cursor.execute(“”””
            INSERT INTO order_item
            SET oi_order_id=%s, oi_product_id=%s, oi_price_per_unit=%s,
oi_cart_quantity=%s, oi_total=%s
        “””, (
            orderID,
            request.POST[‘product_id’],
            request.POST[‘product_price’],
            request.POST[‘product_quantity’],
            totalAmount))
        except Exception as e:
            return HttpResponse(‘Something went wrong. Error Message : ‘+ str€)

        context[“productdetails”] = product.objects.get(product_id = productId)
```

```
        messages.add_message(request, messages.INFO, “Product updated 43uccessfully
!!!”)
        return redirect(‘cart_listing’)
    else:
        return render(request,’products-details.html’, context)

def add(request):
    context = {
        “fn”: “add”,
        “procompanylist”:getDropDown(‘company’, ‘company_id’, ‘company_name’,0,
‘1’),
        “protypelist”:getDropDown(‘type’, ‘type_id’, ‘type_name’,0, ‘1’),
        “heading”: ‘Product add’
    };
    if (request.method == “POST”):
        try:
            product_image = None

            if(request.FILES and request.FILES[‘product_image’]):
                productImage = request.FILES[‘product_image’]
                fs = FileSystemStorage()
                filename = fs.save(productImage.name, productImage)
                product_image = fs.url(productImage)

            addProduct = product(product_name = request.POST[‘product_name’],
            product_type_id = request.POST[‘product_type_id’],
            product_company_id = request.POST[‘product_company_id’],
            product_price = request.POST[‘product_price’],
            product_image = product_image,
            product_description = request.POST[‘product_description’],
            product_stock = request.POST[‘product_stock’])
            addProduct.save()
        except Exception as e:
            return HttpResponse(‘Something went wrong. Error Message : ‘+ str€)

        return redirect(‘productlisting’)

    else:
        return render(request,’products-add.html’, context)

def delete_item(request, itemId):
    cursor = connection.cursor()
    sql = ‘DELETE FROM order_item WHERE oi_id=’ + itemId
    cursor.execute(sql)
    return redirect(‘cart_listing’)

def delete(request, prodId):
    try:
        deleteProduct = product.objects.get(product_id = prodId)
```

```
        deleteProduct.delete()
    except Exception as e:
        return HttpResponse('Something went wrong. Error Message : '+ str€)
    messages.add_message(request, messages.INFO, "Product Deleted Successfully !!!")
    return redirect('productlisting')

def stock(request):
    cursor = connection.cursor()
    cursor.execute(
        "SELECT * FROM stock, products_product WHERE product_id =
stock_product_id")
    stocklist = dictfetchall(cursor)

    context = {
        "stocklist": stocklist
    }

    # Message according Product #
    context['heading'] = "Products Stock Details";
    return render(request, 'stock.html', context)

def deletestock(request, id):
    cursor = connection.cursor()
    sql = 'DELETE FROM stock WHERE stock_id=' + id
    cursor.execute(sql)
    return redirect('stock')

def companylisting(request):
    cursor = connection.cursor()
    cursor.execute(
        "SELECT * FROM company")
    companylist = dictfetchall(cursor)

    context = {
        "companylist": companylist
    }

    # Message according Product #
    context['heading'] = "Products Company";
    return render(request, 'viewcompany.html', context)

def deletecompany(request, id):
    cursor = connection.cursor()
    sql = 'DELETE FROM company WHERE company_id=' + id
    cursor.execute(sql)
    return redirect('company')

def addcompany(request):
    context = {
```

```
            "fn": "add",
            "heading": 'Add Company'
        };
        if (request.method == "POST"):
            cursor = connection.cursor()
            cursor.execute("""
                        INSERT INTO company
                        SET company_name=%s
                    """, (
                request.POST['company_name']))
            return redirect('companylisting')
        return render(request, 'addcompany.html', context)

    def order(request):
        cursor = connection.cursor()
        cursor.execute(
            "SELECT * FROM order_item")
        orderlist = dictfetchall(cursor)

        context = {
            "orderlist": orderlist
        }

        # Message according Orders #
        context['heading'] = "Products Order Details";
return render(request, 'orders
```

3.5.2 Results:

1) Home Page

Fig 4) Home Page

2) About Page

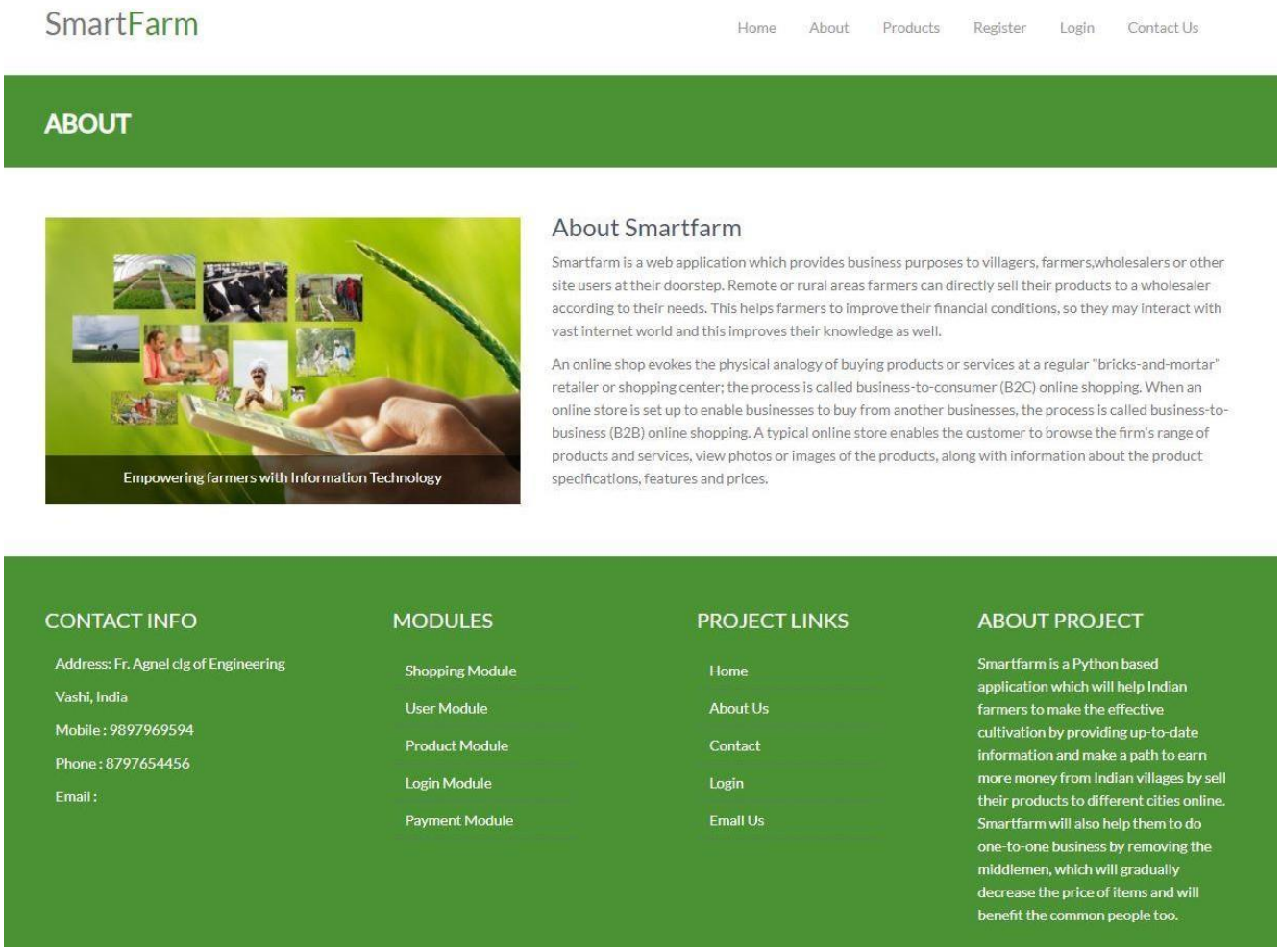
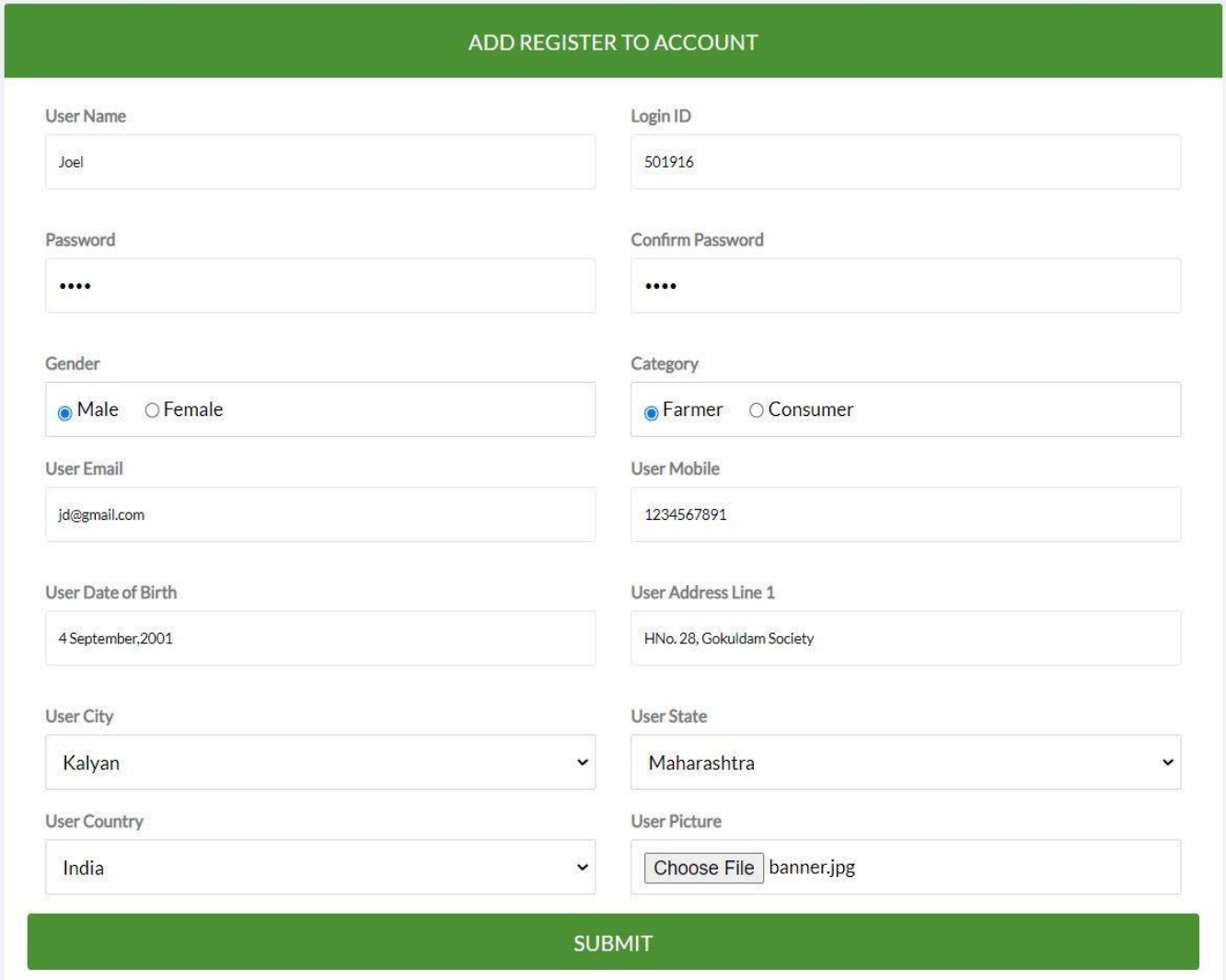


Fig 5) About Page

3) Register and Update Profile Page



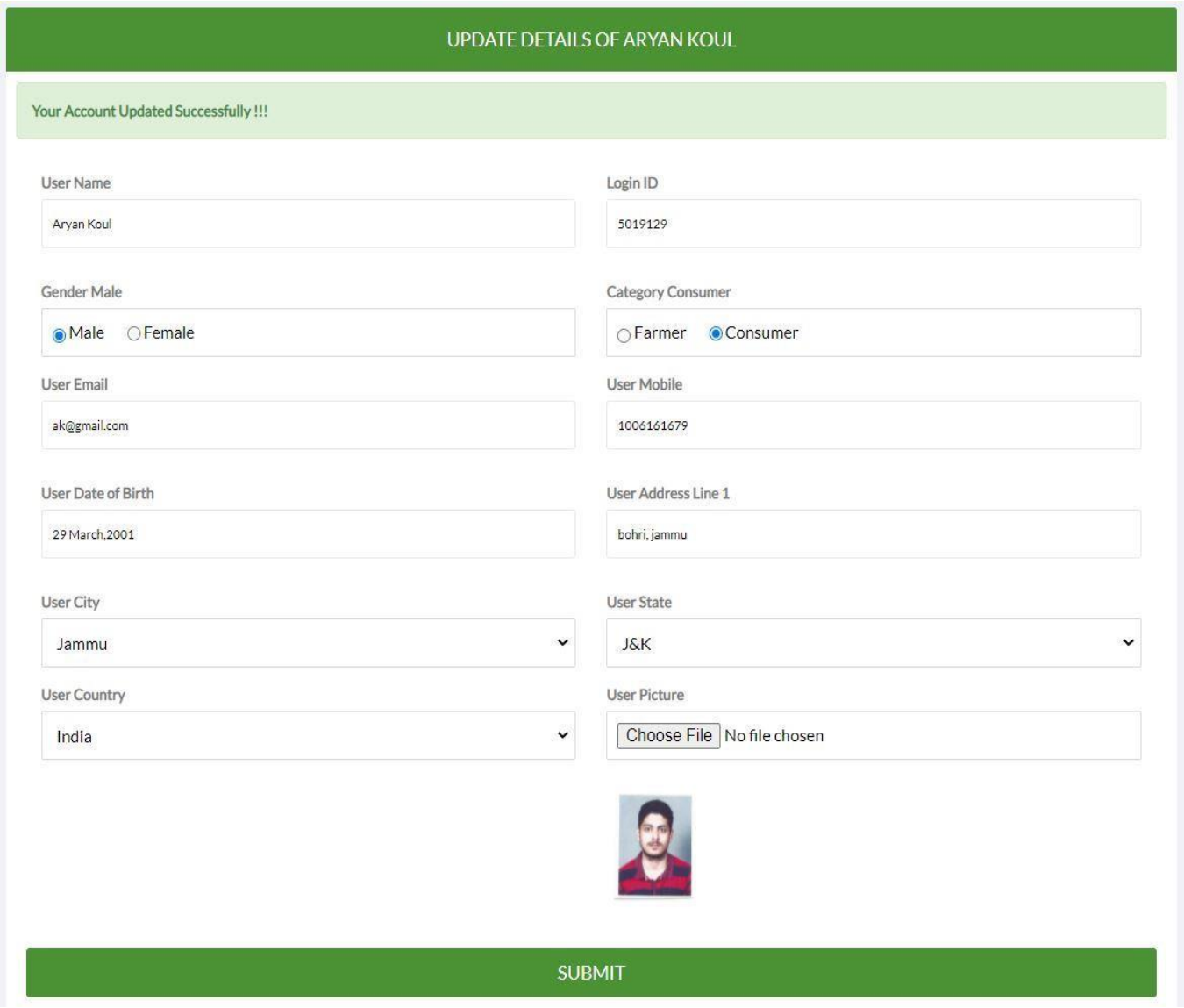


Fig 6) Register and Update Profile Page

4) Products Page

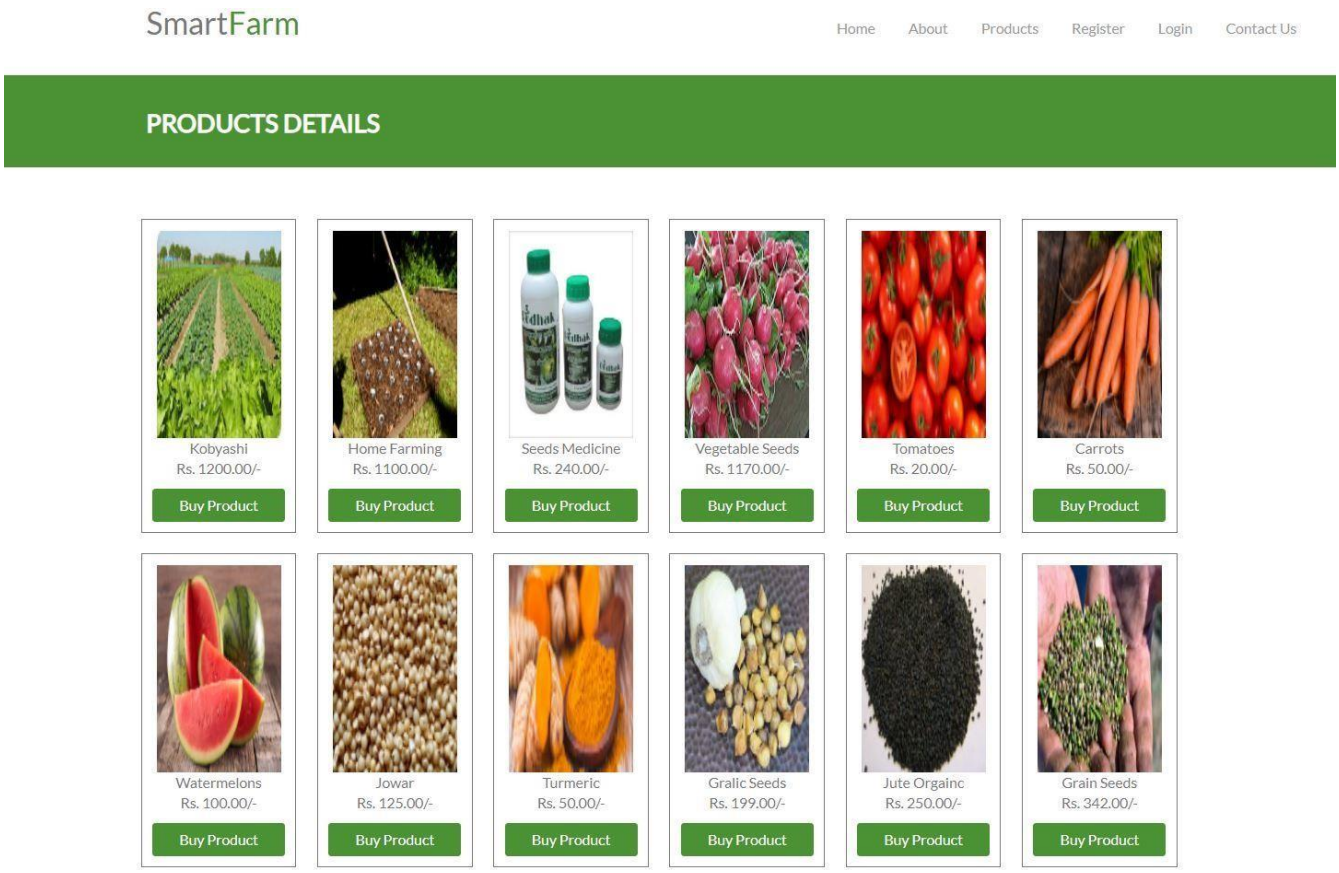


Fig 7) Products Page

5) Shopping Cart Page

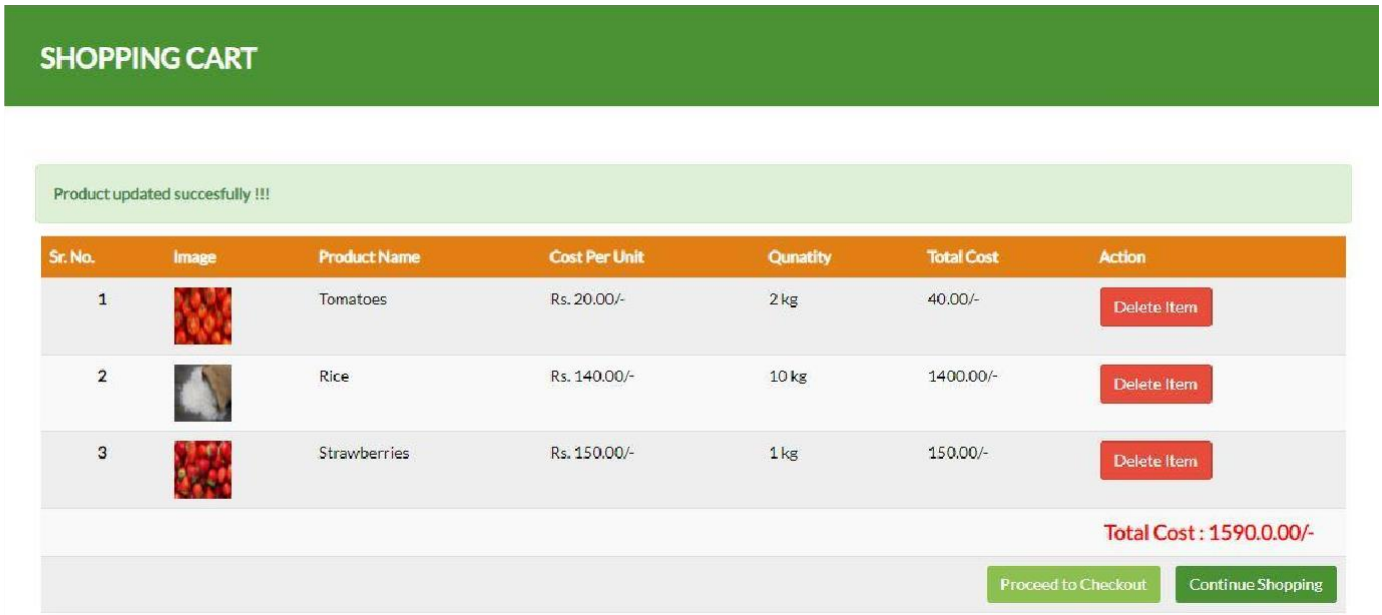
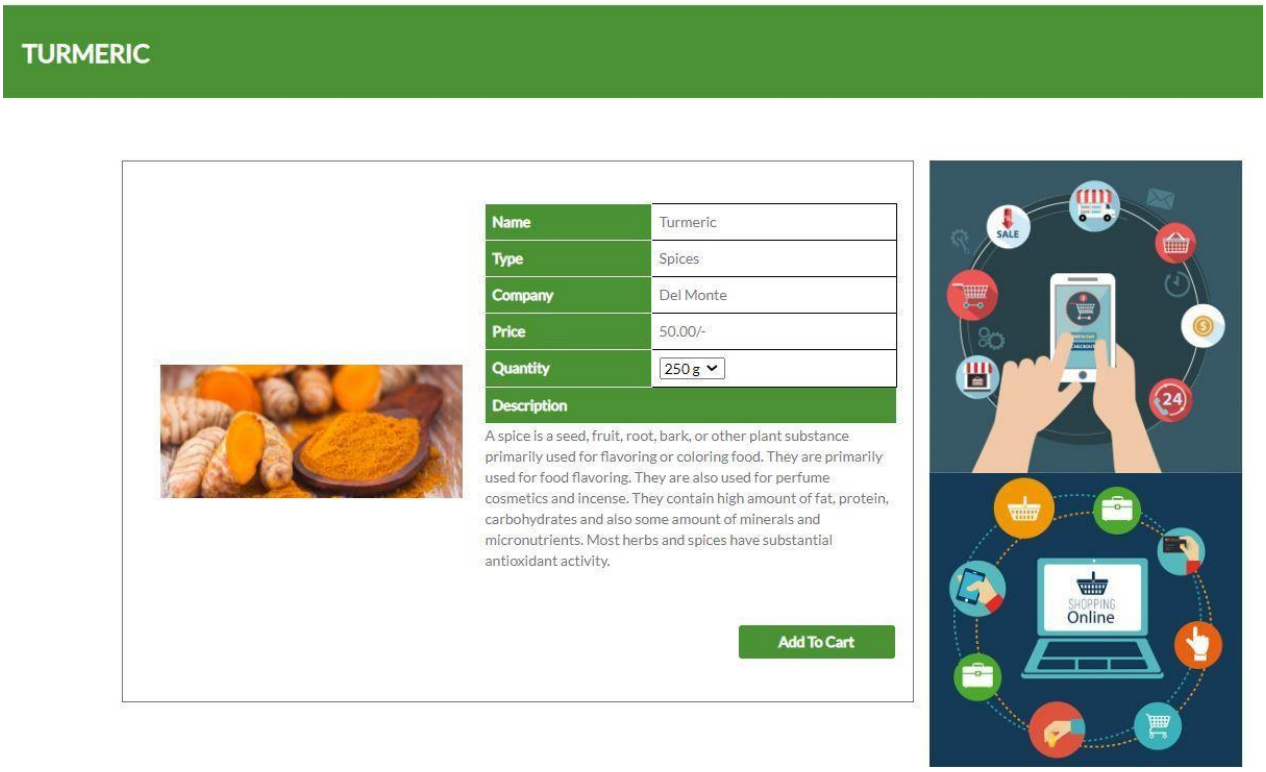


Fig 8) Shopping Cart Page

6) Prodcuts- Details Page



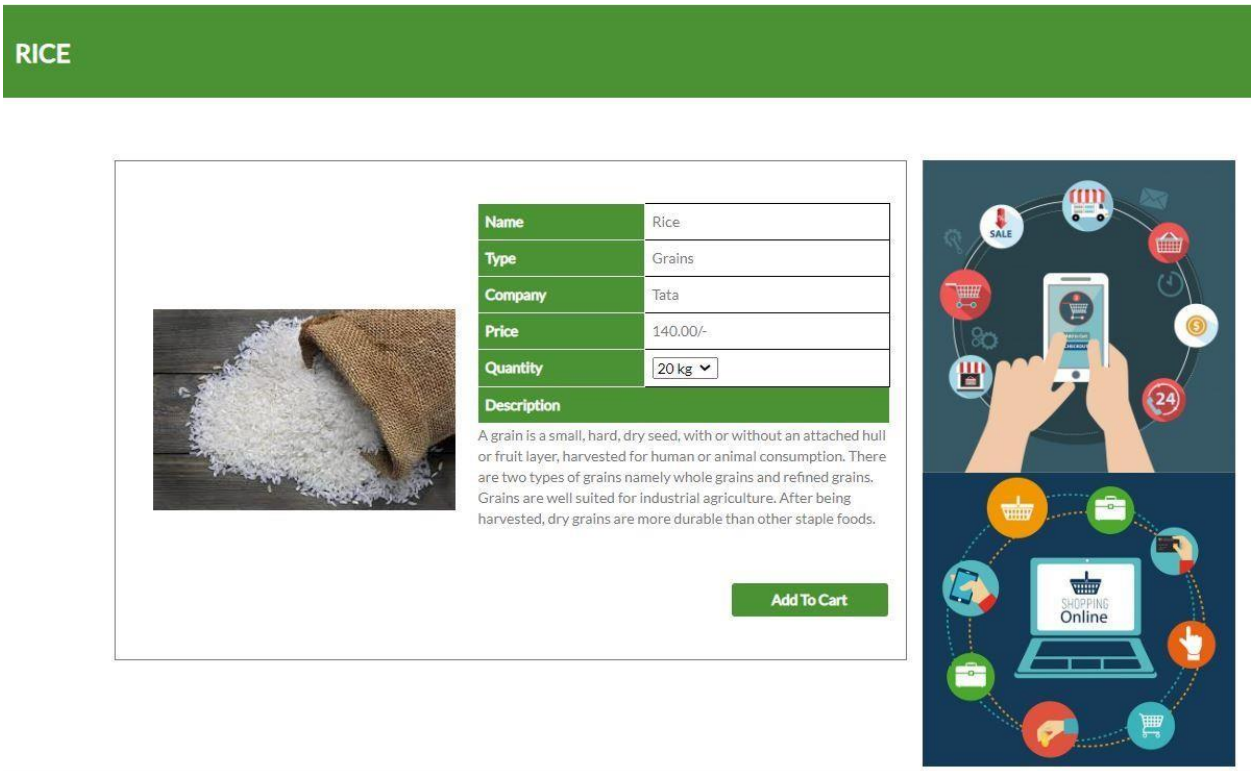


Fig 9) Product Detalis Page

7) Payment Page

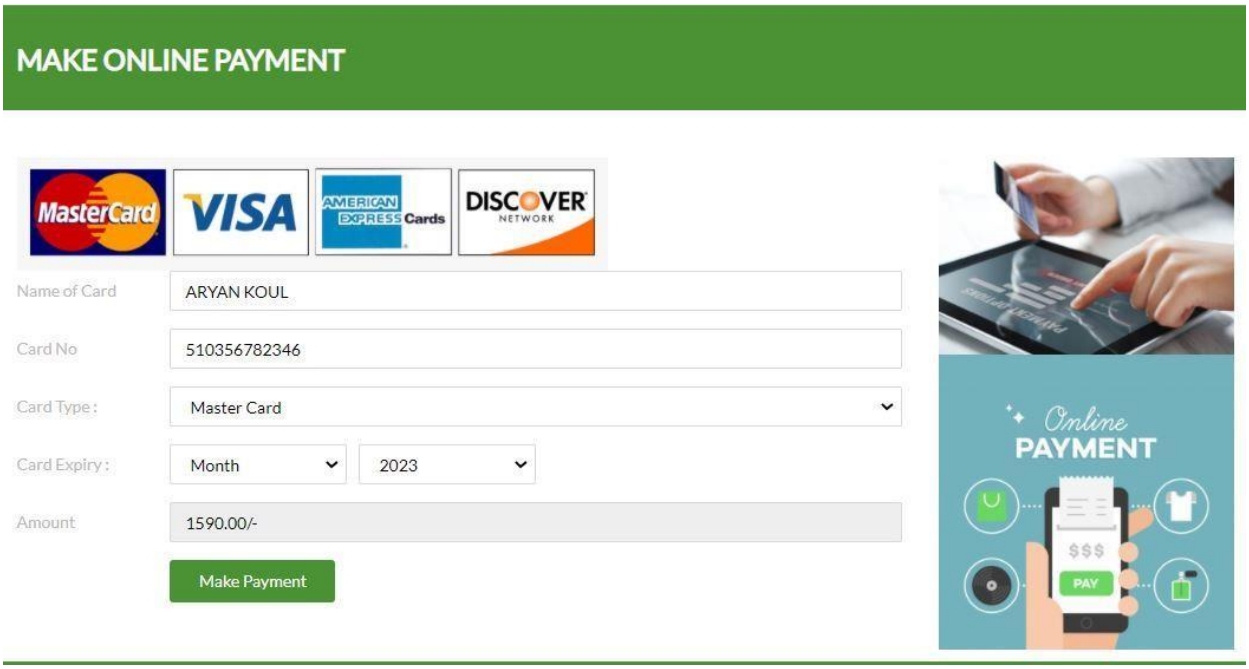
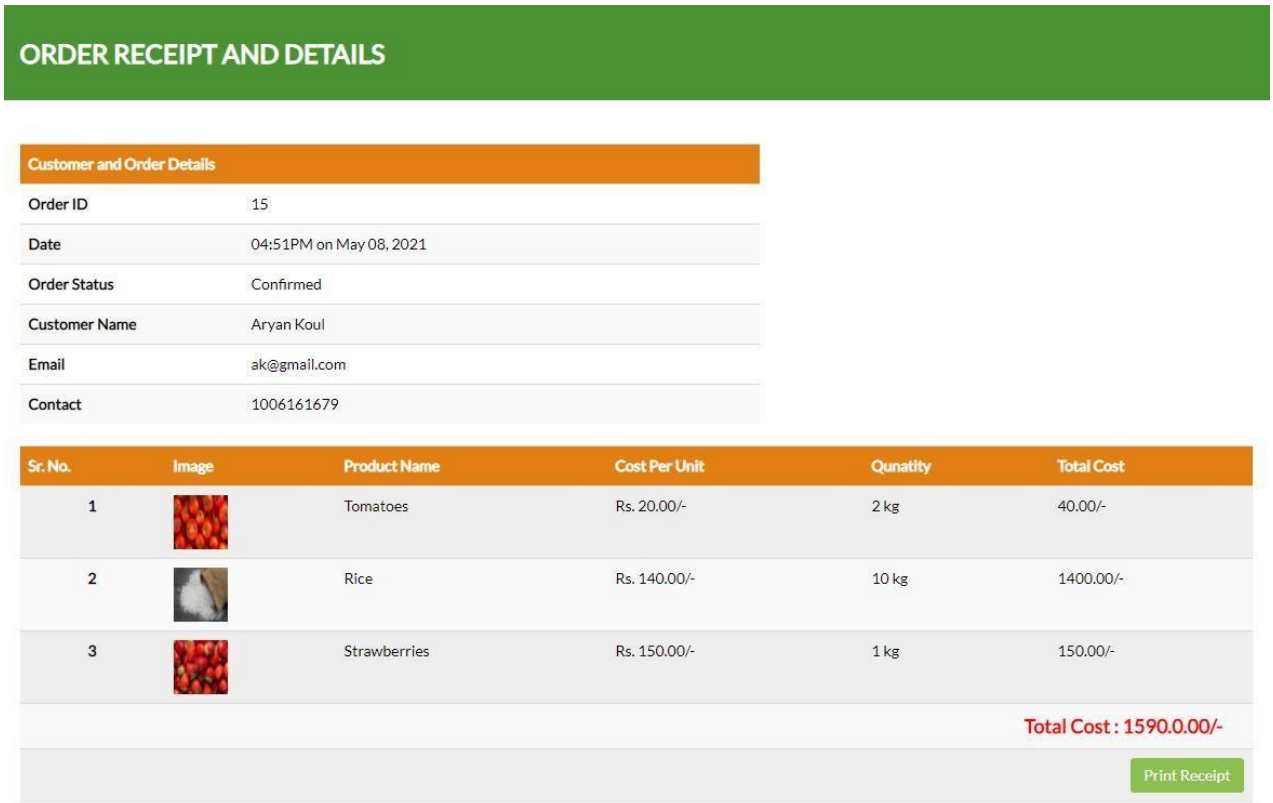


Fig 10) Payment Page

8) Order Details Page



ORDER RECEIPT AND DETAILS

Customer and Order Details	
Order ID	15
Date	04:51PM on May 08, 2021
Order Status	Confirmed
Customer Name	Aryan Koul
Email	ak@gmail.com
Contact	1006161679

Sr. No.	Image	Product Name	Cost Per Unit	Qunatity	Total Cost
1		Tomatoes	Rs. 20.00/-	2 kg	40.00/-
2		Rice	Rs. 140.00/-	10 kg	1400.00/-
3		Strawberries	Rs. 150.00/-	1 kg	150.00/-
					Total Cost : 1590.0.00/-
					Print Receipt

Fig 11) Order Details Page

3.6 Conclusion And Future Work:

CONCLUSION:-

- The application was successfully designed and is tested for accuracy and quality.
- During this project we have accomplished all the objectives and have learnt a lot about the farmers struggle. This project meets the needs of the organization. It will be used in searching, retrieving and generating information for the concerned requests.
- This Project will pave the way for an efficient means to carry out the buying and selling of the products. Farmers will earn money as per the work they have done and will not suffer losses. This system is proposed to replace the existing system where the farmer has to suffer between the manufacturers and the traders.
- It is also very useful to farmers and agricultural students in teaching them and keeping them aware of the new schemes and techniques regarding farming.
- This project is also beneficial for common people, as they will get all the goods comparatively at fewer prices.
- Also one of the major advantages of this project is that it uses Information Technology as the backbone instead of the traditional methods.

FUTURE WORK:-

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:-

- As the technology emerges, it is possible to upgrade the system and can be adaptable to the desired environment.Our application is quite flexible.So, any changes or improvements can be done without much effort.
- Based on the future security issues, security can be improved using emerging technologies.
- Attendance module can be added
- Sub admin module can be added
- Delivery tracking system can also be added
- We have designed false transaction which will show only that transaction is successful. We can implement the payment gateway in future.

References:-

[1] Python in a Nutshell, 3rd Edition- Alex Martelli

[2] SQL Cookbook- O'Reilly

[3] Head-First Python, 2nd edition- Paul Barry

[4] Google, URL: http://www.google.co.in

[5] Wikipedia, URL: http://www.wikipedia.org

[6] Youtube, URL: https://www.youtube.com

[7] GeeksforGeeks, URL: https://www.geeksforgeeks.org