



**NEW HORIZON
COLLEGE OF ENGINEERING**

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

for

Mini Project in Web Frameworks or Operating System (20CSE68)

on

AGRIBUZZ

Submitted by

RANGANATH.K

USN: 1NH19CS739, Sem-Sec: 6-E

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year: 2021-22(EVEN SEM)



NEW HORIZON COLLEGE OF ENGINEERING

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

CERTIFICATE

This is to certify that the mini project work titled

AGRIBUZZ

submitted in partial fulfillment of the degree of Bachelor of Engineering
in Computer Science and Engineering by

RANGNATH.K
USN:1NH19CS739

DURING

EVEN SEMESTER 2021-2022

for

*Course: Mini Project in Web Frameworks or
Operating System-20CSE68*

Signature of Reviewer
HOD

Signature of

SEMESTER END EXAMINATION

Name of the Examiner

Signature with date

1. _____

2. _____

PLAGARISM CERTIFICATE

1NH19CS739_AGRIBUZZ

ORIGINALITY REPORT

14%	%	14%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

- | | | |
|-----------|---|---------------|
| 1 | Dipta Voumick, Prince Deb, Sourav Sutradhar, Mohammad Monirujjaman Khan. "Development of Online Based Smart House Renting Web Application", Journal of Software Engineering and Applications, 2021
Publication | 2% |
| 2 | Sufyan bin Uzayr. "Chapter 4 Working with Python Frameworks", Springer Science and Business Media LLC, 2021
Publication | 2% |
| 3 | Imet Ristemi, Marika Apostolova Trpkovska, Betim Cico. "MyGitIssues Web Application as a Solution in Dealing with Issues on GitHub", 2019 8th Mediterranean Conference on Embedded Computing (MECO), 2019
Publication | 2% |
| 4 | Khondoker Aminuzzaman, Md. Junayed Miah, Md. Anisur Rahman, Mohammad Monirujjaman Khan. "Development of Online Home Sharing Web Application", 2021 IEEE | 1% |
| 20 | T. Sasikala, S. Vigneshwari, Bandreddi Sandeep. "Automating Guide selection process of the Department through Web Application", 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), 2020
Publication | <1% |
| 21 | Touhid Bhuiyan. "Chapter 2 Literature Review", Springer Science and Business Media LLC, 2013
Publication | <1% |

Exclude quotes Off
Exclude bibliography On

Exclude matches Off

ABSTRACT

Developing a Agribuzz project to help the farmers in the digital way to increase their productivity and adopting them to use of the modernized techniques and methods in pricing the crops, purchasing the products and online query solving etc. In this project front end will be designed using the HTML,CSS, JavaScript with the JavaScript framework express and backend storage will be supported by the Google Firebase for data storage and user authentication.

SCOPE: Our project includes

- Ecommerce website
- Price Prediction
- Blogs and articles
- Query section

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha**, Principal, New Horizon College of Engineering, for his constant support and encouragement.

I am grateful to **Dr. Anandhi R J**, Professor and Dean - Academics, NHCE, for her unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I also express my gratitude to **Ms. Jayashree M**, Sr. Assistant Professor, Department of Computer Science and Engineering, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

RANGANATH.K
USN: 1NH19CS739

INDEX

Sl.NO.	TITLE	PAGE NO
	ABSTRACT	I
	ACKNOWLEDGEMENT	II
	LIST OF FIGURES	V
1.	INTRODUCTION	
1.1.	PROBLEM DEFINITION	1
1.2.	OBJECTIVES	1
1.3.	METHODOLOGY TO BE FOLLOWED	2
1.4.	EXPECTED OUTCOMES	2
1.5.	HARDWARE AND SOFTWARE REQUIREMENTS	3
2.	WEB FRAMEWORK	
2.1.	INTRODUCTION	4-5
2.2.	WORLD WIDE WEB	5-6
2.3.	WEB BROWSER	7
2.4.	HTML	7-9
2.5.	XHTML	10
2.6.	CSS	10-11
2.7.	FRAMEWORK	11-15
2.8.	EXPRESS JS	15-16
2.9.	JAVASCRIPT	16-19
2.10.	JSP	19-20

3. DESIGN

3.1. DESIGN GOALS	21
-------------------	----

4. IMPLEMENTATION

4.1. MODULE 1 FUNCTIONALITY	22-23
4.2. MODULE 2 FUNCTIONALITY	24
4.3. MODULE 3 FUNCTIONALITY	25
4.4. MODULE 4 FUNCTIONALITY	26
4.5. MODULE 5 FUNCTIONALITY	27
4.6. MODULE 6 FUNCTIONALITY	28

5. RESULTS

5.1. HOME PAGE	29
5.2. HOME OPTION	29-30
5.3. LOGIN AND REGISTER PAGE	31
5.4. MAIN PAGE	32
5.5. CART PAGE	33
5.6. ORDER DETAILS PAGE	33
5.7. CHECK-OUT PAGE	34

6. CONCLUSION

REFERENCES	36
-------------------	-----------

LIST OF FIGURES

<u>Figure No</u>	<u>Figure Description</u>	<u>Page No</u>
2.4.1	HTML Tags	9
2.4.2	HTML Tags Attributes	9
2.6.1	Types of CSS	10
3.1.1	Designing goals of project	21
5.1.1	Home Page	29
5.2.1	About-Us Page	29
5.2.2	Contact-Us Page	30
5.2.3	Feedback Form	30
5.3.1	Register Form	31
5.3.2	Login Form	31
5.4.1	Main page	32
5.5.1	Cart page	33
5.6.1	Order Details page	33
5.7.1	Check-out Page	34

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

Developing a Agribuzz project to help the farmers in the digital way to increase their productivity and adopting them to use of the modernized techniques and methods in pricing the crops, purchasing the products and online query solving etc.

Here when we enter into the website in the main page we can see only feedback, contact us and information about website i.e. About us and it contains the login and registration page.

When login is successful we can see the four main section ecommerce, price, blog and articles and query solving and each section will function separately.

In this project front end will be designed using the HTML,CSS, JavaScript with the JavaScript framework express and backend storage will be supported by the Google Firebase for data storage and user authentication.

1.2 OBJECTIVES

- The main objective of developing this project is get know about Web frameworks and their documentation like HTML, CSS and JavaScript.
- By developing mini projects students can be capable of solving a real time problems related to computer science field.
- This projects helps to get more knowledge about HTML, CSS and JavaScript by implementing it in own way.
- The objective of developing such a computerized program reduces the paper work and it's safe.

1.3 METHODOLOGY TO BE FOLLOWED

- In this project we need follow some methodology while dealing with this project.
- In the main screen you can select Login option after the registration
- After selecting the login option you must remember the admin id and password, so after entering the admin password only it will allow doing the further process.
- You can give feedback about the website using Feedback option
- This project is developed using the visual studio code so its show better performance in this software.

1.4 EXPECTED OUTCOMES

- In the main screen it is going to display welcome to AGRIBUZZ and the buttons for user selection and the menu contain the following list.
 1. Login - user needs to login for accessing the all features
 2. Registration - register page for creating user account
 3. About us – to know about whole website
 4. Feedback – giving the feedback about website
- After registration and login it will redirect to other page. And this page contains all features of project.
 - a. Price buddy- price calculation for their crops
 - b. E-com – ecommerce website for agri products
 - c. Blogs – contains blogs about the agriculture

1.5 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

- A laptop/Desktop
- Windows 7/8/10,linux or any operating system
- 500 MB or higher RAM
- Minimum 1 GB ROM.
- Keyboard , mouse and monitor

SOFTWARE REQUIREMENTS

- Web browser (Chrome, Microsoft edge, Firefox..)
- Visual studio code software
- Notepad
- Firebase connection

CHAPTER 2

WEB FRAMEWORKS

2.1 INTRODUCTION

A web framework is a software tool that provides a way to build and run web applications. It helps frontend and backend by being the mediator. With this user can create any website. Now it became very easy to make website. In front end we can use html pages to get input from user and css to do styles to make effective website. There are two types of web frame:

There are two main functions of frameworks: to work on the server side (backend), or on the client-side (frontend), corresponding to their type.

2.1.1 Server side framework

This will allow users to create simple web pages. For example django, ruby, flask, seaside. A server- side framework deals with backend. It can be used for security purpose. It can be used for database access and file storage, and for message queuing, and any other logic or process that may need to be on the server.

2.1.2 Client side framework

In this client side framework users can write their code and can create their own website. A client side framework deals with frontend concerns: Showing controls, rendering, accepting and routing input, playing sound or video. For example angular, amber.js.

2.1.3 Resources of web framework

The building your own Python web applications is an awesome way to learn how the WSGI works and many other pieces that combines to make web frameworks using to web developers.

The requests per second examines how the traditionally we use web framework Flask compares to and a sync frame applications like Sanic in and the artificial, simple benchmark.

When we are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing. Also the post is awesome even though the resulting framework is a simplification of what frameworks such as Flask, Django, and Pyramid allow developers to accomplish.

There is another, more recent multi-part tutorial about the building you are own web framework in Python. The series is based on the alcazar project the author is coding for learning purpose:-

1. Handling requests.
2. Routes, class-Based handlers and unit testing.
3. Exception Handling, Static Files and middleware

2.2 WORLD WIDE WEB

It is also known as web. It is a collection of web pages and website stored in the web servers and it helps to local computer through internet. These Web applications fall on the spectrum from the executing a single use case to providing every known web applications feature to every developer.

Few frameworks take the batteries-included approach where everything which is possible comes bundled with the applications while others have a minimal core package that is amenable to extensions provided by many other packages.

There are some other webs frameworks such as Flask and Pyramid are easier to use with non-relational databases by incorporating external Python libraries. There is spectrum

between the minimal functionality with the easy extensibility on one end and including the everything in the framework with tight integration on the other end.

The beginners just want to work on a web application as a learning project then a framework can help you understand the concepts listed above, such as URL routing, data manipulation and authentication that are more common to the majority of the web framework applications.

And if you are experienced programmer with significant web applications the experience you may feel like the existing frameworks do not match your project's requirements. You can mix and match open source libraries such as Werkzeug for WSGI plumbing with your own code to create your own framework. There is still plenty of room in the Python ecosystem for new frameworks to satisfy the needs of web developers that are unmet by Flask, Django, Bottle, Pyramid and many others.

We can also use web framework to build a web application depends on your experience and also what your trying to accomplish. By using a web framework to build a web applications certainly is not required, but it will make most developer's lives easier in many cases.

2.2.1 OPERATIONS OF WWW

Nowadays this web application involves many specialists, but it takes many people in web ops to ensure that everything will work together throughout an applications lifetime. Specialists have emerged that understand the complexities of running a web application. Previously IT operations teams exist, such as the network operation centre and Database Administration function.

2.3 Web Browser

Basically a web browser is an application that is used to access and view the websites. Common web browsers include Microsoft Edge, Mozilla Firefox, Apple Safari and Internet Explorer.

Primary function of a web browser is to render HTML, the code that is used to design web pages.

The web browsers are basically used on many devices, including desktops, laptops, tablets, and smart phones.

2.3.1 WEB 2.0

This website allows users to collaborate, interact with people, with the help of social media dialogue as creators of user generated content in a virtual community. Now contrast the first generation of web 1.0 version of websites where people were limited to the viewing content in the passive manner.

Web 2.0 features include most of social networking sites for example Facebook, blogs, wikis, folksonomies video sharing sites, hosted services, web applications, collaborative consumption platforms, and mash-up applications.

The original version of the web was a collaborative medium, a place where we all meet and read and write. The term semantic web was given by Berners-Lee to a web of content where the meaning can be processed by machines.

2.4 HTML

Hyper Text mark-up language for documents designed to be displayed in web browser. It can be assisted by technologies such as cascading style sheets and scripting languages such as JavaScript. All the tags in html has inbuilt tags. We cannot create new tags in the html.

The web browsers receive HTML documents from a web server and render the documents into multimedia web applications, HTML describes the structure of a web applications

semantically and originally included cues for the appearance of the document. The HTML elements are the building blocks of HTML pages. The HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. The HTML provides a means to create structured documents by denoting structural semantics for text such as heading, paragraph, lists, links, quotes and other items. HTML the elements are delineated by tags, written using angle brackets.

Tags such as ``, `<input/>` directly introduce content into the page. `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but they use them to interpret the content of page.

It can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content. The www (world web wide) consortium, former maintainer of the CSS standards, has encouraged the use of CSS over explicit presentation of HTML.

2.4.1 HTML TAGS

HTML tags describe the structure of pages. With the help of tags, a web browser can distinguish between HTML content. They are like keywords which define that how web browser will format and display the content. This tag contains three main parts: opening tag, closing tag and content. But some tags in HTML are unclosed tags.

When a web browser reads an HTML document, browser reads it from top to bottom and left to right. HTML tags are used to create HTML documents and render their properties. Each HTML tag has different properties.

An HTML file must have some essential tags so that web browser can differentiate between a HTML text and simple text. We can use as many tags we want as per our code requirement.

All HTML tags must be enclosed within `< >` these brackets.

HTML – Tables

HTML Table Tags

Tag	Description
<code><table></code>	Defines a table
<code><th></code>	Defines a header cell in a table
<code><tr></code>	Defines a row in a table
<code><td></code>	Defines a cell in a table
<code><caption></code>	Defines a table caption
<code><colgroup></code>	Specifies a group of one or more columns in a table for formatting
<code><col></code>	Specifies column properties for each column within a <code><colgroup></code> element
<code><thead></code>	Groups the header content in a table
<code><tbody></code>	Groups the body content in a table
<code><tfoot></code>	Groups the footer content in a table

Fig 2.4.1 HML Tags.

HTML: Table Tag Attributes

Attribute	Value	Description
<code><table border= "1"></code>	number (0 to n)	Specifies the thickness of table border (0 = no border)
<code><table align= "center"></code>	left, center, right	Specifies the horizontal position of a table
<code><tr align= "center"></code>	left, center, right	Specifies the horizontal position of table row content
<code><tr valign= "top"></code>	top, middle, bottom	Specifies the vertical position of table row content
<code><td align= "center"></code>	left, center, right	Specifies the horizontal position of table cell content
<code><td valign= "top"></code>	top, middle, bottom	Specifies the vertical position of table cell content
<code><td colspan= "2"></code>	number (0 to n)	Specifies the number of columns a cell should span
<code><td rowspan= "2"></code>	number (0 to n)	Specifies the number of rows a cell should span
<code><td width= "20%"></code>	number (pixel) or %	Specifies the width of a cell
<code><td height= "20%"></code>	number (pixel) or %	Specifies the height of a cell

Example: <http://widit.knu.ac.kr/~kiyang/share/tabledemo.htm>

Fig 2.4.2 HML Tags attributes.

2.5 XHTML

The XHTML stands for Extensible Hyper Text Mark-up language. XHTML is almost identical to HTML but it is stricter than HTML. It is a cross between HTML and XML language. XHTML is HTML defined as an XML application. It is supported by all major browsers.

The HTML is mainly used to create web pages but we can see that many pages on the internet contain “bad” HTML (not follow the HTML rule).

This HTML code works fine in most browsers (even if it does not follow the HTML rules). HTML is introduced to combine the strengths of HTML and XML. The XHTML IS HTML redesigned as XML. It helps you to create better formatted code on your site. XHTML doesn't facilitate you to make badly formed code to be XHTML compatible.

Unlike the HTML (where simple errors like missing out a closing tag are ignored by the browser), XHTML code must be exactly how it is specified to be.

2.6 CSS

The CSS is the language we use to style an HTML document. The CSS describes how HTML elements should be displayed. The CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. Cascading Style Sheets is style sheet language used for describing the presentation of a document which is written in Mark-up language such as HTML. The CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

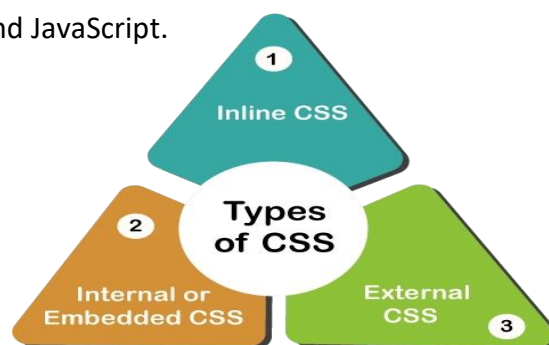


Fig 2.6.1 Types of CSS.

Types of CSS:-

There are three types: -

- inline CSS
- Internal/ Embedded CSS
- External CSS

2.6.1 Internal CSS

The Internal CSS has <style> tag in the <head> section of the HTML document. This CSS style is an effective way to style single pages. Using the CSS style for multiple web pages is time-consuming because we require placing the style on each web page.

2.6.2 External CSS

In external CSS, we link the web pages to the external .css file. It is created by text editor.

The CSS is more efficient method for styling a website. By editing the .css file, we can change the whole site at once.

2.6.3 Inline CSS

Inline CSS is used to style a specific HTML element. Add a style attribute to each HTML tag without using the selectors. Managing a website may difficult if we use only inline CSS. However, Inline CSS in HTML is useful in some situations. We have not access the CSS files or to apply styles to element.

2.7 Framework

A web framework (WF) or web application framework (WAF) is a software framework that is designed to support the development of web applications including the web services, web resources, and the web APIs. Web frameworks provide a standard way to build and deploy web applications on the World Wide Web. Web framework's aim to automate the overhead associated with common activities performed in web development. For example, many web frameworks provide libraries for database access, templating frameworks, and session

management, and they often promote code reuse although they often target development of dynamic web sites, they are also applicable to static websites.

Types of framework architectures:

Most web frameworks are based on the model–view–controller (MVC) pattern. [citation needed]

Many frameworks follow the MVC architectural pattern to separate the data model with business rules from the user interface. This is generally considered a good practice as it modularizes code, promotes code reuse, and allows multiple interfaces to be applied. In web applications, this permits different views to be presented, such as web pages for humans and web service interfaces for remote applications.

2.7.1 Push-based vs. pull-based

Most MVC frameworks follow a push-based architecture also called "action-based". These Frameworks use actions that do the required processing, and then "push" the data to the view layer to render the results. Django, Ruby on Rails, Symphony, Spring MVC, Stripes, Sails.js, and Code Igniter are good examples of this architecture. An alternative to this is pull-based architecture, sometimes also called "component-based". architecture, multiple controllers can be involved with a single view. Lift, Tapestry, JBoss Seam, Java Server Faces, and Wicket are examples of pull-based architectures. Play, Struts, RIFE, and ZK have support for both push- and pull-based application controller calls.

2.7.2 Three-tier organization

In three-tier organization, applications are structured around three physical tiers: client, application, and database. The database is normally an RDBMS. The application contains the business logic, running on a server and communicates with the client using HTTP. The client on web applications is a web browser that runs HTML generated by the application layer. The term should not be confused with MVC, where, unlike in three-tier architecture, it is

considered a good practice to keep business logic away from the controller, the "middle layer".

2.7.3 Model View Controller architecture

More than 80% of all web app frameworks rely on the Model View Controller architecture. The secret of this pattern's popularity is in how rationally it separates the app logic from the interface forming the 3 components reflected in the architecture's name.

Model

The Model knows all about the content and the structure of an app. Upon receiving user input data from the Controller, it communicates the way an updated interface should look directly to the View.

View

This is the app's frontend. It knows the layout and the way a user can interact with any of its parts. The View receives user input, communicates it to the Controller for analysis and updates or reassembles itself according to the Model's instructions (or the Controller's, if a change is minor).

Controller

The Controller is an intermediary between the Model and the View. It receives user input from the View, processes it and informs the Model (or the View) what changes should be made.

Some people advocate that the Controller isn't always necessary and what matters most is to separate the logic from the interface, that is, the Model and the View. Yet, assigning input processing to either Model or View disrupts the pattern's initial ideology of Separated.

When each component in an architecture is responsible for a single line of tasks, the projects transparent, flexible and easy to maintain. Besides, the MVC architecture allows:

- Parallel development (less time to deliver)

- Code reuse
- Fixing or modifying one of the components without having to update the others
- Setting SEO - friendly URLs.

Many web frameworks have incorporated the MVC pattern, so if you're interested in it, make sure the framework of your choice relies on this architecture.

2.7.4 Types of web application frameworks

In the Web 1.0 era, all web apps were mainly built around servers. Such apps still exist and are highly secure, since their entire app logic is stored on the back end.

But as web standards began to change, app logic started to shift toward the client, which helped to ensure a smarter interaction between a user and a web app. With the logic on its side, a client can instantly react to user input. What's more, client-side logic makes apps responsive, so they are easy to navigate on any device. This way, we now have two groups of web application frameworks: one helps to set up app logic on the server, the other – on the client. To create a powerful web app, you can use both of them simultaneously.

Server-side web application framework

Although front end has evolved, its first and foremost job is to display an interface, and without app logic any UI/UX is irrelevant. That's why server-side frameworks are important. Among the most popular MVC-based server-side web frameworks are:

- Symfony (PHP)
- Django (Python)
- Express (Node.js/JavaScript)
- Ruby on Rails (Ruby)
- ASP.NET (C#)

By using either of these server-side web application frameworks, you let it handle HTTP requests, database control and management, as well as URL mapping. You can also render

view data with a server, like in the Web 1.0 era, but consider using client-side frameworks instead to introduce more user-engaging features and responsiveness.

Client-side web application framework

One of the main components of the architecture is a JavaScript client layer. To properly set it up, you need client-side frameworks, such as:

- Bootstrap
- React.js
- Angular.js
- Backbone
- Semantic - UI.

While with server-side frameworks your choice mainly depends on the language you feel comfortable to develop in, here you should mind specific capabilities of different client-side frameworks. Since they vary in the scope of functionality supported, look for the one that fits the needs of your future web app.

2.8 Express JS

Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications. Following are some of the core features of Express framework –

- Allows to set up middlewares to respond to HTTP Requests.
- Defines a routing table which is used to perform different actions based on HTTP Method and URL.
- Allows to dynamically render HTML Pages based on passing arguments to templates.

Express application uses a callback function whose parameters are **request** and **response** objects.

```
app.get('/', function (req, res) {
```

```
// --  
})
```

Request Object – The request object represents the HTTP request and has properties for the request query string, parameters, body, HTTP headers, and so on.

Response Object – The response object represents the HTTP response that an Express app sends when it gets an HTTP request.

Features:-

- Robust routing
- Focus on high performance
- Super-high test coverage
- HTTP helpers (redirection, caching, etc)
- View system supporting 14+ template engines
- Content negotiation
- Executable for generating applications quickly

2.9 JavaScript

JavaScript which also calls in short JS is a programming language that conforms to ECMA script specification. JS is high-level, often just-in-time compiled and multi-paradigm. Along with HTML and CSS JavaScript is one of the core technologies of the World Wide Web. Almost all websites use it client-side for web page behaviour.

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

JavaScript is the most popular programming language in the world and that makes it a programmer's great choice. Once you learnt JavaScript, it helps you developing great front-end as well as back-end software's using different JavaScript based frameworks like jQuery, Node.JS etc.

JavaScript is everywhere, it comes installed on every modern web browser and so to learn JavaScript you really do not need any special environment setup. For example, Chrome, Mozilla Firefox, Safari and every browser you know as of today, supports JavaScript.

JavaScript helps you create really beautiful and crazy fast websites. You can develop your website with a console like look and feel and give your users the best Graphical User Experience.

2.9.1 Application of JavaScript:

- ❖ **Client side validation** - This is really important to verify any user input before submitting it to the server and JavaScript plays an important role in validating those inputs at front-end itself.
- ❖ **Manipulating HTML Pages** - JavaScript helps in manipulating HTML page on the fly. This helps in adding and deleting any HTML tag very easily using JavaScript and modify your HTML to change its look and feel based on different devices and requirements.
- ❖ **User Notifications** - You can use JavaScript to raise dynamic pop-ups on the webpages to give different types of notifications to your website visitors.
- ❖ **Back-end Data Loading** - JavaScript provides Ajax library which helps in loading back-end data while you are doing some other processing. This really gives an amazing experience to your website visitors.
- ❖ **Presentations** - JavaScript also provides the facility of creating presentations which gives website look and feel. JavaScript provides RevealJS and BespokeJS libraries to build a web-based slide presentation.

- ❖ **Server Applications** - Node JS is built on Chrome's JavaScript runtime for building fast and scalable network applications. This is an event based library which helps in developing very sophisticated server applications including Web Servers.

2.9.2 JavaScript Development Tools:

- ❖ **Microsoft FrontPage:** – Microsoft has developed a popular HTML editor called FrontPage. FrontPage also provides web developers with a number of JavaScript tools to assist in the creation of interactive websites.
- ❖ **Macromedia Dreamweaver MX:** – Macromedia Dreamweaver MX is a very popular HTML and JavaScript editor in the professional web development crowd. It provides several handy prebuilt JavaScript components, integrates well with databases, and conforms to new standards such as XHTML and XML.
- ❖ **Macromedia Home Site 5:** – Home Site 5 is a well-liked HTML and JavaScript editor from Macromedia that can be used to manage personal websites effectively.

2.9.3 JavaScript as OOP's:

JavaScript is an Object Oriented Programming (OOP) language. A programming language can be called object-oriented if it provides four basic capabilities to developers –

- **Encapsulation** – the capability to store related information, whether data or methods, together in an object.
- **Aggregation** – the capability to store one object inside another object.

- **Inheritance** – the capability of a class to rely upon another class (or number of classes) for some of its properties and methods.
- **Polymorphism** – the capability to write one function or method that works in a variety of different ways.

Objects are composed of attributes. If an attribute contains a function, it is considered to be a method of the object; otherwise the attribute is considered a property.

2.10 JSP

Java Server Pages (JSP) is a server-side programming technology that enables for creating of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java API s, including the JDBC API to access enterprise databases. This will help us how to use Java Server Pages to develop your web applications in simple and easy steps.

Why to Learn JSP?

Java Server Pages often serve the same purpose as programs implemented using the **Common Gateway Interface (CGI)**. But JSP offers several advantages in comparison with the CGI.

Performance is significantly better because JSP allows embedding Dynamic Elements. JSP are always compiled before they are processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.

Java Server Pages are built on top of the Java Servlets API, so like Servlets, JSP also has access to all the powerful Enterprise Java APIs, including **JDBC, JNDI, EJB, JAXP**, etc.

JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

Finally, JSP is an integral part of Java EE. This means that JSP can play a part in the simplest applications to the most complex and demanding.

2.10.1 Applications of JSP:

As mentioned before, JSP is one of the most widely used language over the web. I'm going to list few of them here:

JSP vs. Active Server Pages (ASP)

The advantages of JSP are twofold. the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.

JSP vs. Pure Servlets

It is more convenient to write regular HTML than to have many `println` statements that generate the HTML.

JSP vs. Server-Side Includes (SSI)

SSI is only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.

JSP vs. JavaScript

JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.

CHAPTER 3

DESIGN

3.1 DESIGN GOALS

- Allow users to easily see and make it user friendly.
- Give users an easy way to back out (i.e. change & error recovery).
- Allow user to complete their task without being distracted by software or losing train of thought, which is while they are reading and typing.
- Everything will be made it by menu selection for easy understand.
- Give users access to information they need to complete their task.
- Wishing the user whenever they open the screen.
- Collects the details one by one without overloading.



Fig 3.1.1 Designing goals of project.

CHAPTER 4

IMPLEMENTATION

4.1 MODULE 1 FUNCTIONALITY

➤ Home page with css.

```
<!DOCTYPE html>
<html>
<head>
  <title>Agribuzz</title>
  <link rel="stylesheet" type="text/css" href="style_index.css">
</head>
<body>
  <header>
    <div class="main">
      <ul>
        <li><a href="#"><strong>Home</strong></a></li>
        <li><a href="aboutus.html"><strong>About Us</strong></a></li>
        <li><a href="contactform.html"><strong>Contact Us</strong></a></li>
        <li><a href="feedback.html"><strong>Feedback</strong></a></li>
      </ul>
    </div>
    <div class="title">
      <h1>Agribuzz</h1>
    </div>
    <div class="button">
      <a href="login.html" class="btn"><strong>Login</strong></a>
      <a href="registration.html" class="btn"><strong>Register</strong></a></div>
    </header>
  </body>
</html>
```

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="utf-8">
  <title></title>
  <link rel="stylesheet" href="style_contactform.css">
</head>
<body>
  <div class="contact-form">
    <h1>Contact Us</h1>
    <div class="txtb">
      <label>Full Name :</label>
      <input type="text" name="" value="" placeholder="Enter Your Name">
    </div>

    <div class="txtb">
      <label>Email :</label>
      <input type="email" name="" value="" placeholder="Enter Your Email">
    </div>

    <div class="txtb">
      <label>Phone Number :</label>
      <input type="text" name="" value="" placeholder="Enter Your Phone Number">
    </div>

    <div class="txtb">
      <label>Message :</label>
      <textarea></textarea>
    </div>
    <a class="btn">Send</a>
  </div>
</body>
</html>
```

```
<link rel="stylesheet" href="style_aboutus.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

<div class="wrapper">
  <h1>About Us</h1>
  <div class="our_team">
    <div class="team_member">
      <div class="member_img">
        
    <div>
      <a href="#" class="fa fa-facebook"></a>
      <a href="#" class="fa fa-twitter"></a>
      <a href="#" class="fa fa-linkedin"></a>
      <a href="#" class="fa fa-instagram"></a>
    </div>
    <div class="team_member">
      <div class="member_img">
        
      </div>
      <h3>Connect With Experts</h3>
      <span></span>
      <p>Any questions ? Feel free to ask us . Agribuzz provide farmers agricultural experts and advisers to guide the farmers and

<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title></title>
    <link rel="stylesheet" href="style_feedback.css">
  </head>
  <body>
    <div class="contact-form">
      <h1>Feedback</h1>
      <div class="txtb">
        <label>Full Name :</label>
        <input type="text" name="" value="" placeholder="Enter Your Name">
      </div>
      <div class="txtb">
        <label>Email :</label>
        <input type="email" name="" value="" placeholder="Enter Your Email">
      </div>
      <div class="txtb">
        <label>Phone Number :</label>
        <input type="text" name="" value="" placeholder="Enter Your Phone Number">
      </div>
      <div class="txtb">
        <label>Message :</label>
        <textarea></textarea>
      </div>
      <a class="btn">Send</a>
    </div>
  </body>
</html>
```

4.2 MODULE 2 FUNCTIONALITY

➤ Login and Registration page.

```

<html>
<head>

<title>Login Form</title>
  <link rel="stylesheet" type="text/css" href="style_login.css">
</head>
<body>

  <div class="loginbox">

    
    <h1>Login Here</h1>
    <p>Username</p>
    <input type="text" name="username" id = "username" placeholder="Enter Username" required>
    <p>Password</p>
    <input type="password" name="password" id = "password" placeholder="Enter Password" required>
    <!-- <input type="submit" id="sub" value="Login" > -->
    <input type="button" value="Login" id="log">

    <a href="#">Lost your password?</a><br>
    <a href="registration.html">Don't have an account?</a>
    <marquee><h4 style="color: red;" id="data"></h4></marquee>

  </div>
  <script type="module" src="login.js"></script>
</body>

</html>

<html>
<head>
<title>Register Yourself</title>
  <link rel="stylesheet" type="text/css" href="style_register.css"/>
</head>
<body>

  <div class="register">
    <form id="register">

      
      <h1>Register Yourself</h1>
      <center><h4 id="result" style="color: red;"></h4></center>
      <label>First Name</label>
      <input type="text" name="fname" id="fname" placeholder="Enter First Name" required>

    <br>
    <div>
      <label>Last Name</label>
      <input type="text" name="lname" id="lname" placeholder="Enter Last Name" required>
    </div>
    <br>
    <div>
      <label>Contact Number</label>
      <input type="number" name="number" id="number" value="" placeholder="Enter your Contact Number" minlength="10" required>
    </div>
    <br>
    <div>
      <label>Email</label>
      <input type="email" name="email" id="email" placeholder="Enter your Email" required>
    </div>
    <br>
    <div>

```


4.3 MODULE 3 FUNCTIONALITY

➤ Main page of agribuzz.

```

<body>
<section id="nav-bar">
  <nav class="navbar navbar-expand-lg navbar-light">

    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="nav
    |   <span class="navbar-toggler-icon"></span>
    </button>
    <div class="topnav">
      <a style="color:■ black;" class="active" href="index1.html"> <b>AGRI</b>BUZZ </a>
    </div>

    <div class="collapse navbar-collapse" id="navbarNav">

      <ul class="navbar-nav ml-auto">
        <li class="search">
          <i class="fas fa-search search"></i>
          <input type="text" id="input" name="searchBox" placeholder="Search for Agri products....." style="w
        </li>
        <li class="nav-item">
          <a style="color:■ black;" class="nav-link" href="#featured-categories">Home</a>
        </li>
        <li class="nav-item">
          <a style="color:■ black;" class="nav-link" href="cart.html">Cart</a>
        </li>

        <li class="nav-item">
          <a style="color:■ black;" class="nav-link" href="#services">Our Services</a>
        </li>
        <li class="nav-item">
          <a style="color:■ black;" class="nav-link" href="#contact">Contact</a>
        </li>
        <li class="nav-item">
          <a style="color:■ black;" class="nav-link" href="index.html">Logout</a>
        </li>
      </ul>
    </div>
  </nav>
</section>

.title-box
{
  background: ■ #00CDCE;
  color: □ #fff;
  width: 180px;
  padding: 4px 10px;
  height: 40px;
  margin-bottom: 30px;
  display: flex;
  margin-top: 70px;
}

.title-box h2
{
  font-size: 24px;
}

.title-box::after
{
  content: '';
  border-top: 40px ■ #00cdce;
  border-right: 50px solid transparent;
  position: absolute;
  display: flex;
  margin-top: -4px;
  margin-left: 170px;
}

h1, h2, h3, h4, h5, h6 {
  font-weight: normal;
  font-size: 50%;
}

#site {
  max-width: 960px;
  margin: 0 auto;
  padding: 0 1em;
}

```

4.4 MODULE 4 FUNCTIONALITY

➤ Cart Design

```

<!DOCTYPE html>
<html>
<head>
<title>Your Shopping Cart</title>
<meta charset="utf-8" />
<link rel="stylesheet" href="style.css" media="screen" type="text/css" />
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/2.0.3/jquery.min.js"></script>
<script type="text/javascript" src="jquery.store.js"></script>
<style>
  body {
    background-image: url('images/cart.jpg');
    background-repeat: no-repeat;
    background-attachment: fixed;
    background-size: cover;
  }
  h1 {
    padding-top: 100px;
  }
</style>
</head>

<body>

<div id="site">

  <div id="content">
    <h1>Your Shopping Cart</h1>
    <form id="shopping-cart" action="cart.html" method="post">
      <table class="shopping-cart">
        <thead>
          <tr>
            <th scope="col">Item</th>
            <th scope="col">Qty</th>
            <th scope="col" colspan="2">Price</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td><div id="item"></div>
            <td><div id="qty"></div>
            <td colspan="2"><div id="price"></div>
          </tr>
        </tbody>
      </table>
      <p id="sub-total">
        <strong><u>Sub Total:<span id="stotal"></span></u></strong>
      </p>
      <ul id="shopping-cart-actions">
        <li>
          <input type="submit" name="update" id="update-cart" class="btn" value="Update Cart" />
        </li>
        <li>
          <input type="submit" name="delete" id="empty-cart" class="btn" value="Empty Cart" />
        </li>
        <li>
          <a href="index1.html" class="btn">Continue Shopping</a>
        </li>
        <li>
          <a href="checkout.html" class="btn">Go To Checkout</a>
        </li>
      </ul>
    </form>
  </div>

</div>

<footer id="site-info">
  Copyright &copy; AgriBuzz
</footer>

</body>
</html>

```

4.5 MODULE 5 FUNCTIONALITY

➤ Order and Checkout page

```

<body id="checkout-page">

<div id="site">
  <div id="content">
    <h1>Your Order</h1>
    <table id="checkout-cart" class="shopping-cart">
      <thead>
        <tr>
          <th scope="col">Item</th>
          <th scope="col">Qty</th>
          <th scope="col">Price</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td>
            <div id="pricing">
              <p id="shipping">
                <strong>Shipping</strong>: <span id="sshipping"></span>
              </p>
              <p id="sub-total">
                <strong>Total</strong>: <span id="stotal"></span>
              </p>
            </div>
          </td>
        </tr>
      </tbody>
    </table>
    <div id="user-details">
      <h2>Your Data</h2>
      <div id="user-details-content"></div>
    </div>
  </div>
</div>

<form action="order.html" method="post" id="checkout-order-form" style="text-align: center; ">
  <h2>Your Details</h2>

  <fieldset id="fieldset-billing">
    <legend>Billing</legend>
    <div>
      <label for="name">Name</label>
      <input type="text" name="name" id="name" data-type="string" data-message="This field cannot be empty" />
    </div>
    <div>
      <label for="email">Email</label>
      <input type="text" name="email" id="email" data-type="expression" data-message="Not a valid email address" />
    </div>
    <div>
      <label for="city">City</label>
      <input type="text" name="city" id="city" data-type="string" data-message="This field cannot be empty" />
    </div>
    <div>
      <label for="address">Address</label>
      <input type="text" name="address" id="address" data-type="string" data-message="This field cannot be empty" />
    </div>
    <div>
      <label for="zip">ZIP Code</label>
      <input type="text" name="zip" id="zip" data-type="string" data-message="This field cannot be empty" />
    </div>
    <div>
      <label for="country">Country</label>
      <select name="country" id="country" data-type="string" data-message="This field cannot be empty">
        <option value="">Select</option>
        <option value="Denmark">Denmark</option>
        <option value="India">India</option>
      </select>
    </div>
  </fieldset>

```

4.6 MODULE 6 FUNCTIONALITY

➤ JQuery for main page

```

(function( $ ) {
    $.Shop = function( element ) {
        this.$element = $( element );
        this.init();
    };

    $.Shop.prototype = {
        init: function() {

            // Properties

            this.cartPrefix = "Furniture-"; // Prefix string to be prepended to the cart's name in the session storage
            this.cartName = this.cartPrefix + "cart"; // Cart name in the session storage
            this.shippingRates = this.cartPrefix + "shipping-rates"; // Shipping rates key in the session storage
            this.total = this.cartPrefix + "total"; // Total key in the session storage
            this.storage = sessionStorage; // shortcut to the sessionStorage object

            this.$formAddToCart = this.$element.find( "form.add-to-cart" ); // Forms for adding items to the cart
            this.$formCart = this.$element.find( "#shopping-cart" ); // Shopping cart form
            this.$checkoutCart = this.$element.find( "#checkout-cart" ); // Checkout form cart
            this.$checkoutOrderForm = this.$element.find( "#checkout-order-form" ); // Checkout user details form
            this.$shipping = this.$element.find( "#shipping" ); // Element that displays the shipping rates
            this.$subTotal = this.$element.find( "#stotal" ); // Element that displays the subtotal charges
            this.$shoppingCartActions = this.$element.find( "#shopping-cart-actions" ); // Cart actions links
            this.$updateCartBtn = this.$shoppingCartActions.find( "#update-cart" ); // Update cart button
            this.$emptyCartBtn = this.$shoppingCartActions.find( "#empty-cart" ); // Empty cart button
            this.$userDetails = this.$element.find( "#user-details-content" ); // Element that displays the user information
            this.$paypalForm = this.$element.find( "#paypal-form" ); // PayPal form

            this.currency = "&#8377;"; // HTML entity of the currency to be displayed in the layout
            this.currencyString = "₹"; // Currency symbol as textual string
            this.paypalCurrency = "RS"; // PayPal's currency code
            this.paypalBusinessEmail = "yourbusiness@email.com"; // Your Business PayPal's account email address
            this.paypalURL = "https://www.sandbox.paypal.com/cgi-bin/webscr"; // The URL of the PayPal's form

            handleCheckoutOrderForm: function() {
                var self = this;
                if( self.$checkoutOrderForm.length ) {
                    var $sameAsBilling = $( "#same-as-billing" );
                    $sameAsBilling.on( "change", function() {
                        var $check = $( this );
                        if( $check.prop( "checked" ) ) {
                            $( "#fieldset-shipping" ).slideUp( "normal" );
                        } else {
                            $( "#fieldset-shipping" ).slideDown( "normal" );
                        }
                    });

                    self.$checkoutOrderForm.on( "submit", function() {
                        var $form = $( this );
                        var valid = self._validateForm( $form );

                        if( !valid ) {
                            return valid;
                        } else {
                            self._saveFormData( $form );
                        }
                    });
                }
            },
        },
    };

```

CHAPTER 5

RESULTS

5.1 Home Page

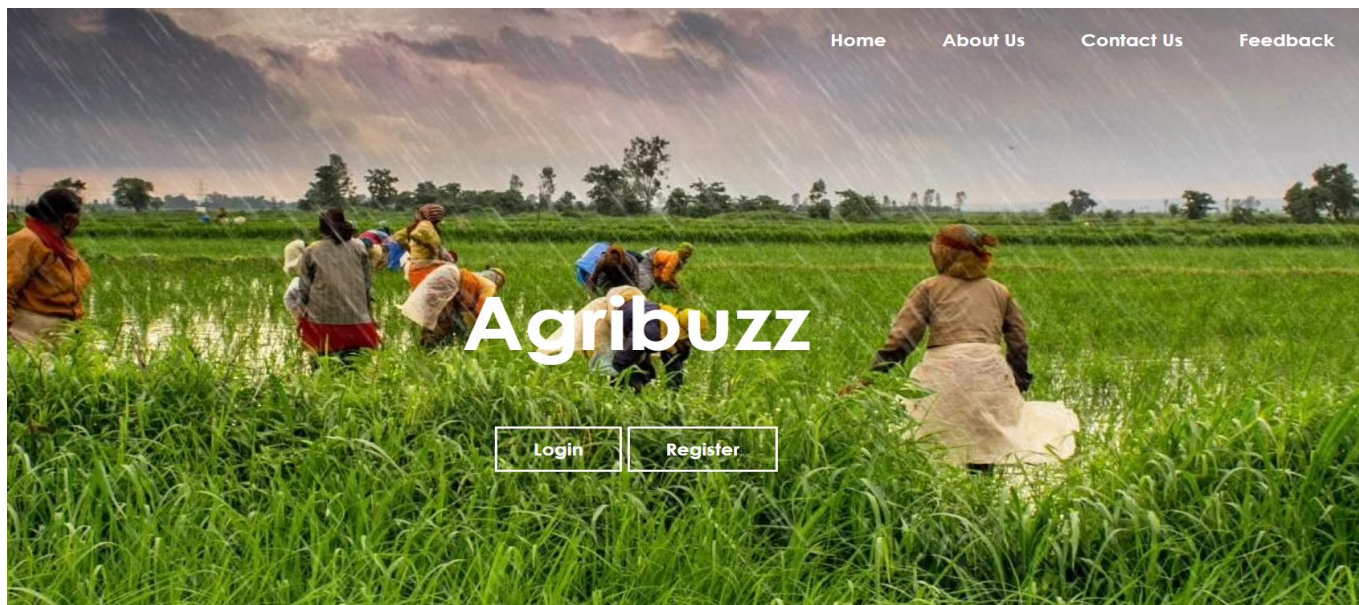


Fig 5.1.1 Home page.

5.2 Home Option

➤ About-Us

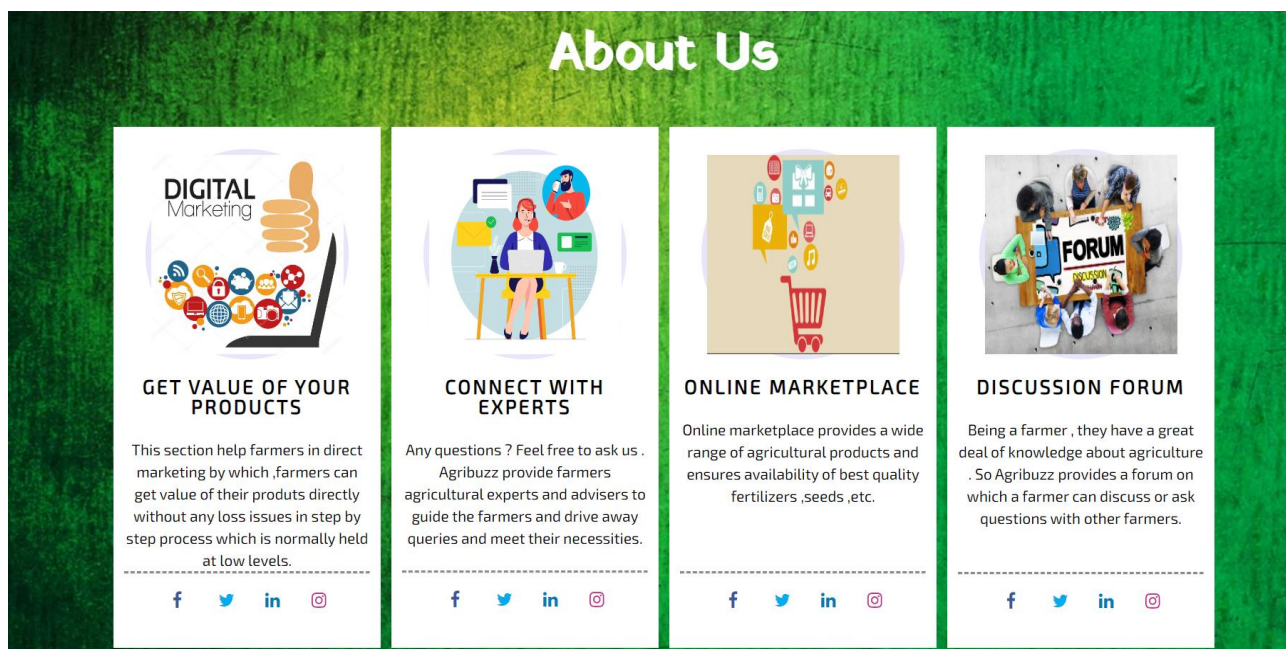
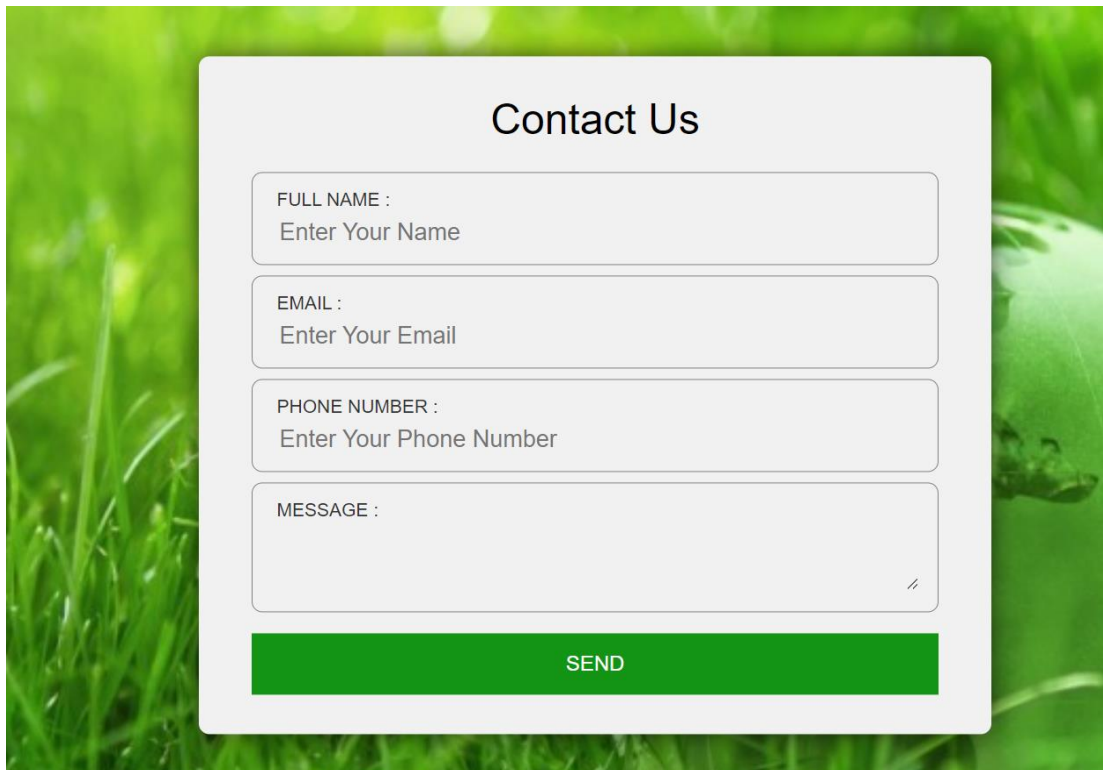


Fig 5.2.1 About-Us page.

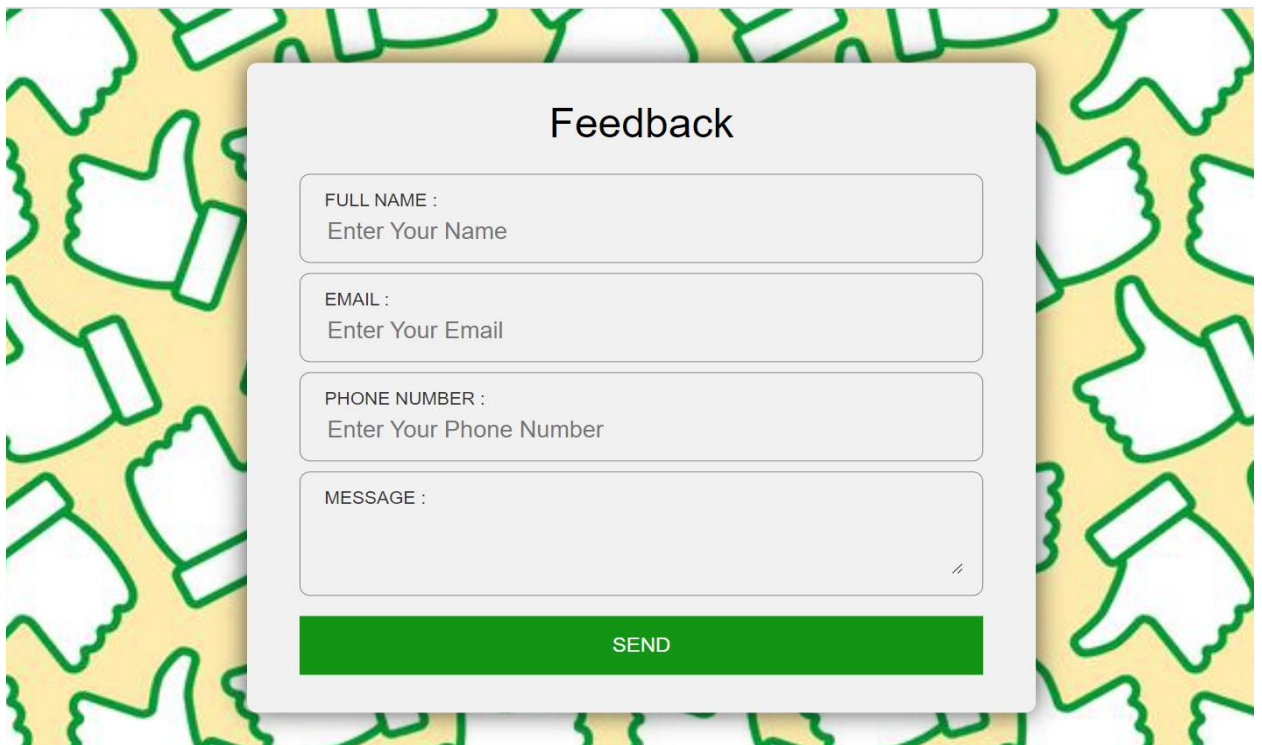
➤ **Contact-Us**



The image shows a 'Contact Us' form centered on a background of green grass. The form is a light gray rectangle with rounded corners. At the top, it has the title 'Contact Us' in a bold, black, sans-serif font. Below the title are four input fields, each with a label and a placeholder text: 'FULL NAME : Enter Your Name', 'EMAIL : Enter Your Email', 'PHONE NUMBER : Enter Your Phone Number', and 'MESSAGE :'. The 'MESSAGE' field is a larger text area with a small cursor icon at the bottom right. At the bottom of the form is a wide, green button with the word 'SEND' in white, uppercase letters.

Fig 5.2.2 Contact-Us page.

➤ **Feedback**



The image shows a 'Feedback' form centered on a background of yellow with green thumbs-up icons. The form is a light gray rectangle with rounded corners. At the top, it has the title 'Feedback' in a bold, black, sans-serif font. Below the title are four input fields, each with a label and a placeholder text: 'FULL NAME : Enter Your Name', 'EMAIL : Enter Your Email', 'PHONE NUMBER : Enter Your Phone Number', and 'MESSAGE :'. The 'MESSAGE' field is a larger text area with a small cursor icon at the bottom right. At the bottom of the form is a wide, green button with the word 'SEND' in white, uppercase letters.

Fig 5.2.3 Feedback Form.

5.3 Login and Register Page

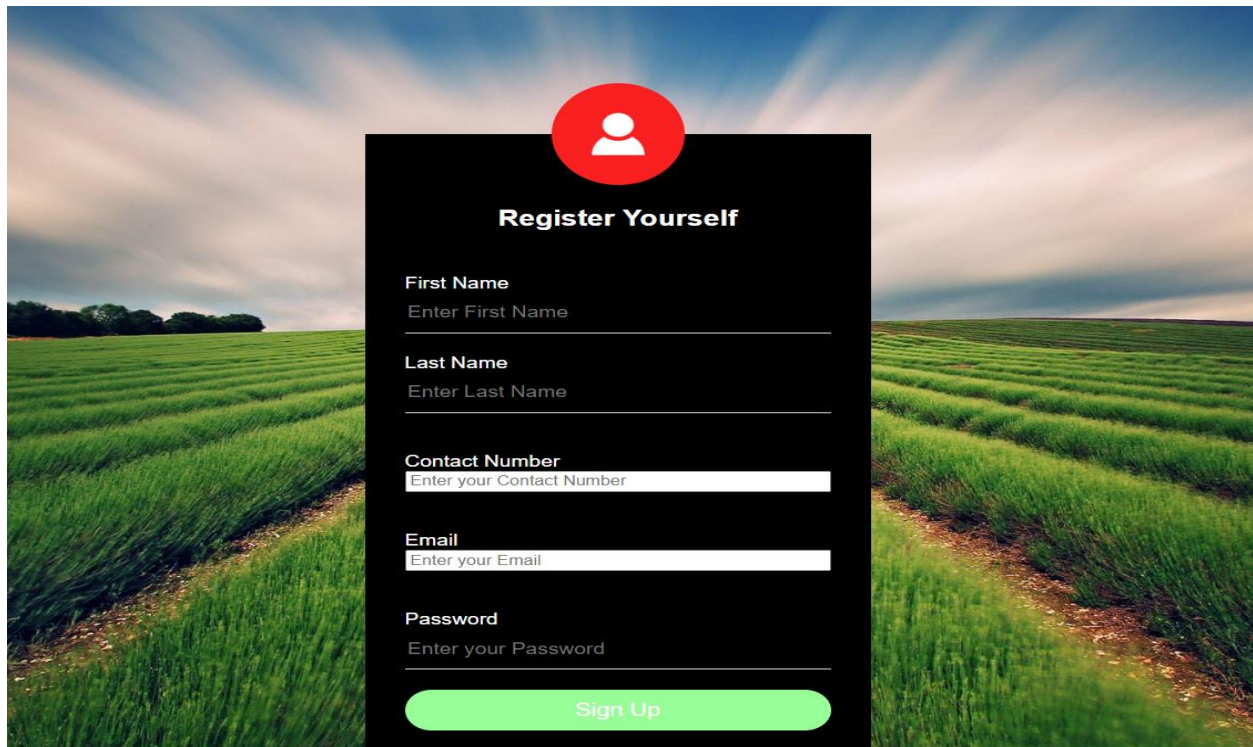
The image shows a registration form titled "Register Yourself" overlaid on a background of a green field under a cloudy sky. The form is a dark gray rectangle with a red circular icon containing a white person silhouette at the top. It includes input fields for "First Name", "Last Name", "Contact Number", "Email", and "Password", each with a placeholder text. A green "Sign Up" button is at the bottom.

Fig 5.3.1 Register Form.

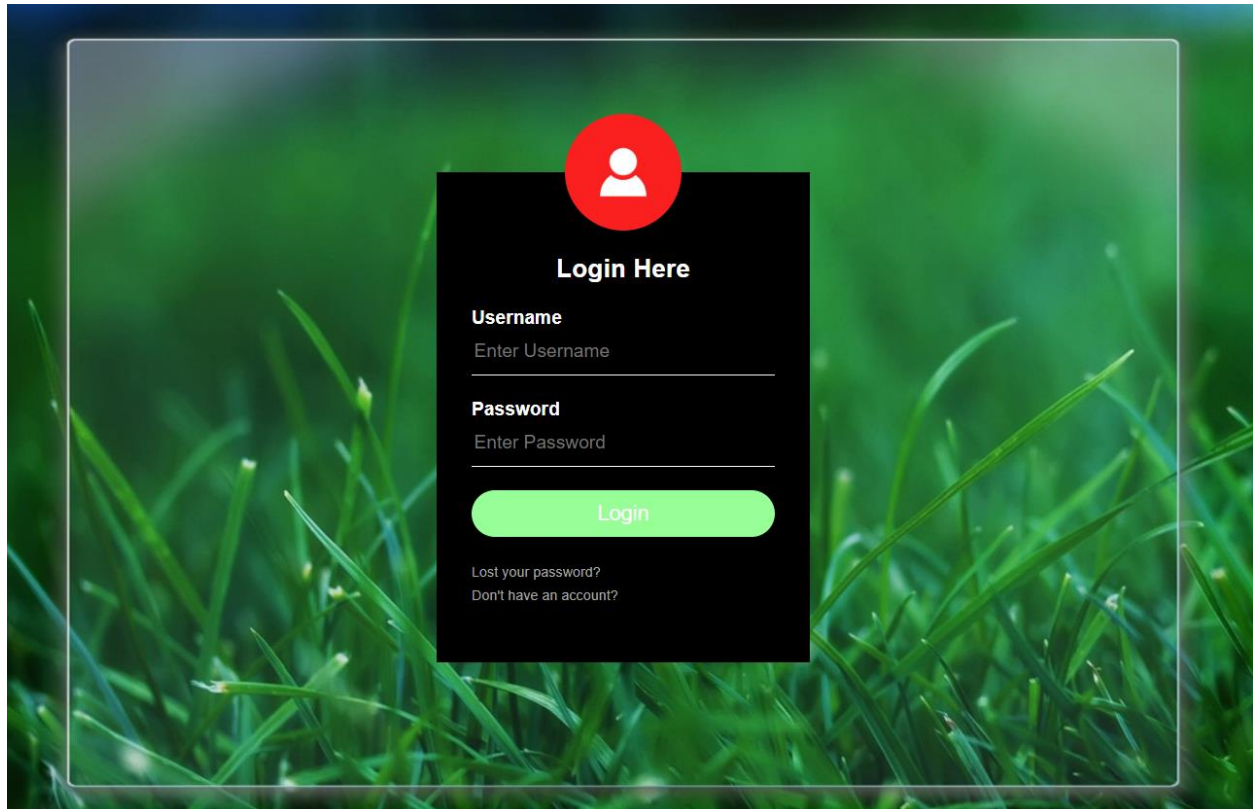
The image shows a login form titled "Login Here" overlaid on a background of green grass. The form is a dark gray rectangle with a red circular icon containing a white person silhouette at the top. It includes input fields for "Username" and "Password", each with a placeholder text. A green "Login" button is at the bottom. Below the button, there are two links: "Lost your password?" and "Don't have an account?".

Fig 5.3.2 Login Form.

5.4 Main Page

AGRIBUZZ

[Home](#)
[Cart](#)
[Our Services](#)
[Contact](#)
[Logout](#)

New Arrivals

★★★★★
Kisan Urea 46% 45 kg Bag
₹ 268

Quantity
[Add to cart](#)

★★★★★
ANNPRASH Premium Pearl Millet (1 kg)
₹ 44

Quantity
[Add to cart](#)

★★★★★
Falcon Garden Premium Shovel
₹ 873

Quantity
[Add to cart](#)

★★★★★
Nutraj Cucumber Seeds 200g
₹ 329

Quantity
[Add to cart](#)

★★★★★
Agrico Square Shovel with Wooden Handle
₹ 630

Quantity
[Add to cart](#)

★★★★★
NPK 00:52:34(mono potassium phosphate)
₹ 530

Quantity
[Add to cart](#)

★★★★★
Brown Raw Barley High in Protein
₹ 35

Quantity
[Add to cart](#)

★★★★★
Root Care Organic Plant Fertilizer(3kg)
₹ 973

Quantity
[Add to cart](#)

Our Services

24/7 Support
on order related queries

Return within 30 days
of receiving your order

Get free delivery
on orders above Rs 499

Get In Touch

Fig 5.4.1 Main page.

Department of CSE, NHCE

Page 32

5.5 Cart Page

Your Shopping Cart

ITEM	QTY	PRICE	
Kisan Urea 46% 45 kg Bag	<input type="text" value="1"/>	₹ 268	×
ANNPRASH Premium Pearl Millet (1 kg)	<input type="text" value="1"/>	₹ 44	×
Brown Raw Barley High in Protein	<input type="text" value="1"/>	₹ 35	×
Falcon Garden Premium Shovel	<input type="text" value="1"/>	₹ 873	×

Sub Total: ₹ 1220

[Update Cart](#) [Empty Cart](#) [Continue Shopping](#) [Go To Checkout](#)

Fig 5.5.1 Cart page.

5.6 Order Details Page

Checkout

ITEM	QTY	PRICE
Kisan Urea 46% 45 kg Bag	1	₹ 268
ANNPRASH Premium Pearl Millet (1 kg)	1	₹ 44
Brown Raw Barley High in Protein	1	₹ 35
Falcon Garden Premium Shovel	1	₹ 873

Shipping: ₹ 0
Total: ₹ 1220

Your Details

Billing

Name

Email

City

Address

ZIP Code

Country

Same as Billing ☒

[Submit](#)

Fig 5.6.1 Order Details page.

5.7 Check-out Page

Your Order

ITEM	QTY	PRICE
Kisan Urea 46% 45 kg Bag	1	₹ 268
ANNPRASH Premium Pearl Millet (1 kg)	1	₹ 44
Brown Raw Barley High in Protein	1	₹ 35
Falcon Garden Premium Shovel	1	₹ 873

Shipping: ₹ 0

Total: ₹ 1220

Your Data

Billing	Shipping
<input type="text" value="Ranganath k"/>	<input type="text" value="umesh"/>
<input type="text" value="ranganath.k.rmp@gmail.com"/>	<input type="text" value="abc@gmail.com"/>
<input type="text" value="Banglore"/>	<input type="text" value="Hubli"/>
<input type="text" value="new horizon college of engineering , outer ring road marathahalli"/>	<input type="text" value="3rd cross 4th main road ,Hubli, karnataka"/>
<input type="text" value="5774466"/>	<input type="text" value="512245"/>
<input type="text" value="India"/>	<input type="text" value="IN"/>

Pay Now

Fig 5.7.1 Check-out Page.

CHAPTER 6

CONCLUSION

This program makes the user simpler project to help the farmers in the digital way to increase their productivity and adopting them to use of the modernized techniques and methods in pricing the crops, purchasing the products and online query solving etc.

When login is successful we can see the four main section ecommerce, price, blog and articles and query solving and each section will function separately. In this project front end will be designed using the HTML,CSS, JavaScript with the JavaScript framework express and backend storage will be supported by the Google Firebase for data storage and user authentication.

REFERENCES

- [1] <https://www.programiz.com>
- [2] <https://www.geeksforgeeks.org>
- [3] <https://www.javatpoint.com>
- [4] <https://anzelg.github.io>
- [5] <https://www.tutorialspoint.com>
- [6] <https://www.w3schools.com>
- [7] <https://developer.mozilla.org>
- [8] <https://hackr.io/blog>