# **GROW GREEN (FARMER BLOG & CHAT)**

# **A Project Report**

**Submitted in partial fulfilment of the**

**Requirements for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**BY:**

**PRABHAT PANDEY**

**B. TECH, CSE-B (8th Sem.)**

**Exam Roll No. 4076961**

**Roll No. 23671**

**Under the Supervision of**

**MR. DHEERAJ SAHNI**

**Assistant Professor**

**Department of Computer Science & Engineering**



**UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY,**

**MAHARSHI DAYANAND UNIVERSITY,**

**ROHTAK-124001**

**CANDIDATE DECLARATION CERTIFICATE**

I hereby certify that the work which is being presented in the project report entitled Grow Green (Chat & Blog) by Prabhat Pandey in Complete fulfillment of requirements for the award of the degree of B. Tech submitted to the Department of CSE, U.I.E.T. at M.D.U Rohtak is an authentic record of my own work carried out during a period from Jan to June, 2023 under the supervision of Asst. professor Dheeraj Sahni.

This is to certify that the above statement by the candidate is correct to the best of my knowledge.

Signature of the Student

**Acknowledgment**

Prabhat Pandey is highly grateful to Dr. Yudhvir Singh, Director, University Institute of Engineering and Technology (UIET) for providing this opportunity to carry out the major project work at college.

The constant guidance and encouragement received from Dheeraj sahni, CSE Department, UIET, MDU, Rohtak has been of great help in carrying out the project work and is acknowledged with reverential thanks.

I would like to express a deep sense of gratitude and thanks profusely to Project Guide, without his wise counsel and able guidance, it would have been impossible to complete the project in this manner.

I expressed gratitude to other faculty members of the Computer Science and Engineering Department of UIET, MDU, Rohtak for their intellectual support throughout the course of this work.

Finally, I am indebted to all whosoever have contributed in this report work.

Signature of the Supervisor

**TABLE OF CONTENT**

**Contents Page No.**

**CANDIDATE DECLARATION CERTIFICATES I**

**ACKNOWLEDGEMENT II**

**LIST OF FIGURES III**

**LIST OF SYSMBOLS AND ABBREVIATIONS 1**

**CHAPTER 1: INTRODUCTION 2**

* 1. INTRODUCTION.
  2. OBJECTIVE.
  3. SCOPE.

**CHAPTER 2: TECHNOLOGY DESCRIPTION 4**

2.1 HARDWARE REQUIREMENTS

2.2 SOFTWARE REQUIREMNTS

**CHAPTER 3: CONTROL FLOW OF PROJECT 7**

3.1 WATERFALL MODEL

3**.2** CONTROL FLOW OF PROJECT

**CHAPTER 4: IMPLEMENTATIONS 10**

4.1 FRONTEND DESCRIPTION

4.1.1 HTML, CSS

4.1.2 BOOTSTRAPS

4.2 BACKEND DESCRIPTION

4.2.1 DJANGO, PYTHON

4.2.2 SQLite

**CHAPTER 5: RESULTS AND DISCUSSION 20**

5.1 DESIGN SCREENSHOTS.

### 5.2 LIMITATIONS.

### 5.3 CHALLENGE’S

### 5.4 FUTURE PLAN

### CHAPTER 6: CONCLUSION AND RECOMMENDATIONS 32

**CHAPTER 7: FUTURE SCOPE 33**

**BIBLIOGRAPHY 34**

## LIST OF FIGURE

**Figure 1:** Data Flow Diagram 9

**Figure 2:** Homepage of Grow Green (Chat & Blog) 20

**Figure 3:** Navigation Bar of website 20

**Figure 4**: Footer of Website 21

**Figure 5**: Blog Posts 21

**Figure 6:** Sign-Up Page 22

**Figure 7:** Sign-in Page 22

**Figure** 8: About Page 23

**Figure 9**: Read Post 24

**Figure 10**: Blog Page

**Figure 11:** See Recent Posts 25

**Figure** 12: Forum List Page 27

**Figure 13:** Search Result Page 29

**Figure 14:** Chat End To End Page 29

**Figure 15:** Contact Page Administrator 30

**Figure 16:** Admin/User Pane 30

#### **LIST OF SYMBOLS AND ABBREVIATIONS**

1. SSP - Student Study Portal
2. SSM - Student Study Management
3. ELP - E-Learning Platform
4. LMS - Learning Management System
5. SIS - Student Information System
6. CMS - Content Management System
7. API - Application Programming Interface
8. DBMS - Database Management System
9. HTML - Hypertext Markup Language
10. CSS - Cascading Style Sheets
11. JS - JavaScript
12. SQL - Structured Query Language
13. MVC - Model-View-Controller
14. CRUD - Create, Read, Update, Delete
15. LDAP - Lightweight Directory Access Protocol
16. OAuth - Open Authorization
17. SSL - Secure Socket Layer
18. HTTP - Hypertext Transfer Protocol

# **Chapter 1 Introduction**

# **Introduction**

This project is aimed to provide farmers a platform where they can interact with the expert and the peers. It provide them the facility of communicating with the peers and the expert where they can gain knowledge and new skills for a particular field which in this case is farming. Any suggestion and doubt is very much appreciated as well as can be asked respectively by using the features such as comments and by direct chat. Comments can be used on both the forum as well as the on the blog posts. Grow Green provide seamless and secure integration of comments where user cannot provide the comment without getting authenticated. Talking about authentication, then there’s no need to be getting worried about security as it implemented using **Django** Framework and so it provides the greatest security by default, weather it is in terms of authentication and authorization or providing safety of your data from the malicious user. Django is already proven by the other companies as well such as Redit and Instagram.

* 1. **Objective**

The main objective of this project is to provide Farmers an easy friendly web app where they can interact and seek help regarding farming and growing.

Reduce the time and lack of knowledge farmers face by providing 24\*7 accessible website with user -friendly interface.

Grow green is introduced to solve the complications faced by the farmers on the field. Also, it aims to provide the knowledge which is necessary in order to yield a fine crop while ensuring the maximum profitability and secondly trying to maximize production to its peak.

### Grow Green does many works such as:

* Providing Farmers, a platform where they can communicate directly to the experts and the peers.
* Provide the way to share the ideas among the peers which can in turn help each other.
* Help in solving the personalized problem.

### The advantages of the Grow Green Web application platform:

* Time saving Technology.
* Improved Efficiency.
* Help in reduction of cost.
* Improved communication.
* Can provide potential help in economic boost.
* Reduce the extra work.
* Easy access to information
* Cost effective.

# **Scope**

The proposed software product is useful for the farmers and the farming related technical enthusiasts. As we know our most of the people of India doing the farming and also India is the great producer of different crops and

the spices and also unfortunately many of them are not that much literate to grow crops efficiently and reach out the full potential.

# **Chapter 2**

# **Technical Description**

* 1. **Hardware Requirements**
     + CPU and RAM

Central Processing Unit: Greater or equivalent to Pentium 4 processors for the client to use

RAM: Less than 2 GB

* + - Graphics Card

Do you need a graphics card to code and learn Python? Well, the one integrated with the CPU is enough. However, if you want to use the laptop also for games and photo processing, you should choose dedicated graphics cards.

I have been using NVIDIA cards in my computers for years – GeForce and Quadro (for video processing). There are great AMD cards on the market. These are all very similar to each other when it comes to performance.

* + - **Storage**

SSD drives have been the standard for data storage for several years. However, there area few options here.

Older laptops may have SSD drives connected via the SATA interface. Newer ones are usually equipped with modern M.2 disks with an NVMe interface. There are no major differences in basic applications, but with

large files, you do experience a significant improvement in data transfer.

An M.2 SSD (connector on the motherboard) with a capacity of 240 or 512GB is enough for learning Python. You can easily fit the operating system and all the necessary programs in it.

**Software Requirements**

* **Operating System**

This depends on what you want to do. Some people cannot imagine working on anything other than Windows. Many professional developers, however, choose Linux distributions. They areopen-sourcee and easy to modify.

If you want to start working with Linux, try the Ubuntu distribution – it’s the most popular. You can download and [RUN IT IN LIVE MODE](https://ubuntu.com/tutorials/try-ubuntu-before-you-install) if you’re not ready to make a full switch; you won't need to install anything. See if you can leave Microsoft and join the rebellion.

If you choose a Mac, you'll work on OS X. More and more people are using the OS from Google, Chrome OS. You'll find it in one of the laptops on my list. It's very simple, and you don't think of it for professional programming. However, it is good enough for learning Python.

* + - Python 3.10

Python is an easy-to-learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming.

Python’s elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source

or binary form for all major platforms from the Python web site, <https://www.python.org/>, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.The Python interpreter is easily extended with new functions and data types

Implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

* + - **Visual Studio Code**

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS, and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as **C++, C#, Java, Python, PHP, Go, .NET**).

* + - **Web Browsers**

Any web browser is acceptable to run this application whether it is on phone or be it on pc or any other digital device supporting the **JavaScript**, **jQuery.**

### Some of the recommended browsers are:

* + - * Chrome
      * Microsoft Edge
      * Mozilla Firefox
      * Apple Safari

# **Chapter 3**

**Control Flow Of Project**

**DESCRIPTION OF USED MODEL**

**3.1 The Waterfall Model**

The Waterfall model is a sequential software development process, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design (validation), Construction, Testing and Maintenance.

To follow the waterfall model, one proceeds from one phase to the next in a sequential manner. For example, one first completes requirements specification, which after sign-offare considered "set in stone." When the requirements are fully completed, one proceedsto design. The software in question is designed and a blueprint is drawn for implementers(coders) to follow — this design should be a plan for implementing the requirements given. When the design is fully completed, an implementation of that design is made by coders. Towards the later stages of this implementation phase, separate software components produced are combined to introduce new functionality and reduced risk through the removal of errors.

Thus the waterfall model maintains that one should move to a phase only when its preceding phase is completed and perfected. However, there are various modified waterfall models (including Royce's final model) that may include slight or major variations upon this process. Time spent early in the software production cycle can lead to greater economy at later stages. It has been shown that a bug found in the early stages (such as requirements specification or design) is cheaper in terms of money, effort and time, to fix than the same bug found later on in the process.

This is the central idea behind the waterfall model - time spent early on making sure that requirements and design are absolutely correct will save you much time and effort later.

complete and absolutely correct before proceeding to the next phase of program creation. Program requirements should be set in stone before design is started (otherwise work put into a design based on incorrect requirements is wasted); the program's design should be perfect before people begin work on implementing the design(otherwise they are implementing the wrong design and their work is wasted), etc.

Basic principles of the waterfall model are:

Project is divided into sequential phases, with some overlap and splash back acceptable between phases.

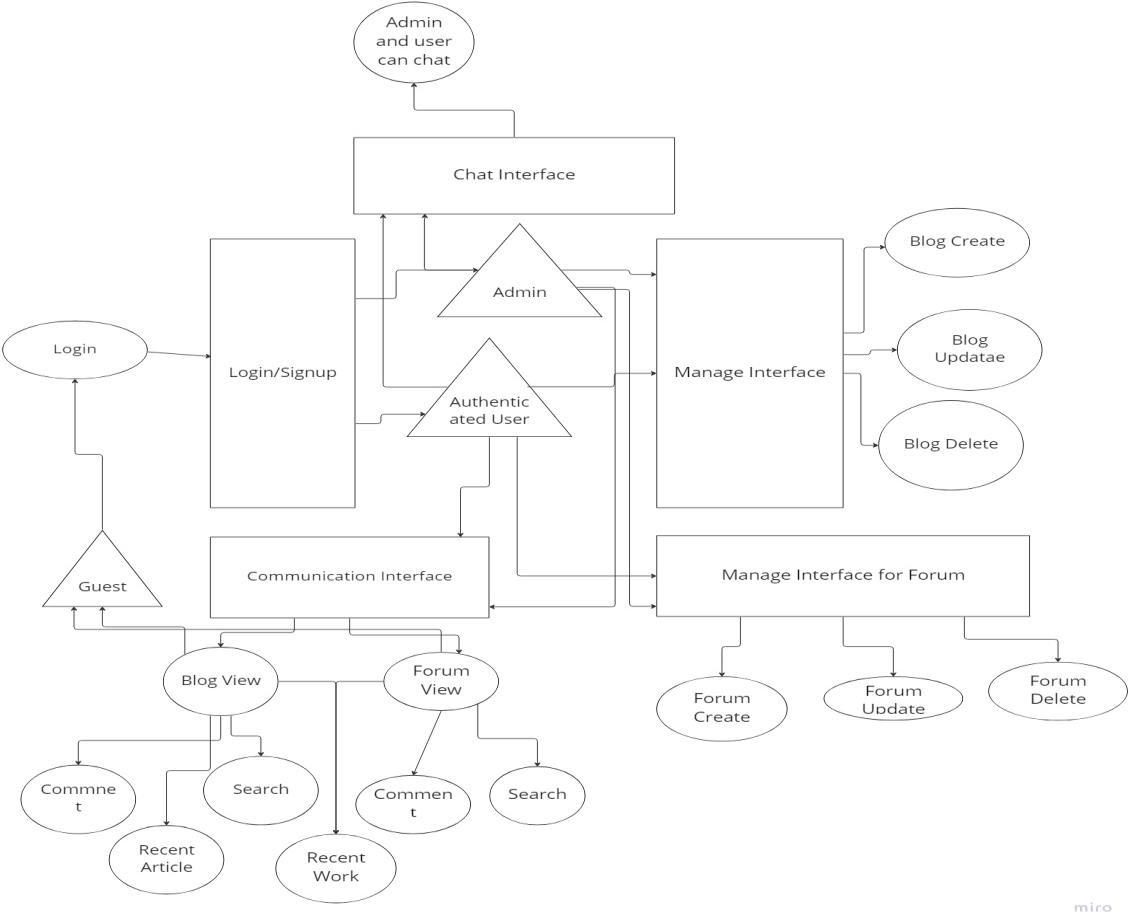
Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.

Tight control is maintained over the life of the project through the use of extensive written documentation, as well as through formal reviews and approval/signoff by the user and information technology management occurring at the end of most phases before beginningthe next phase.

**3.2 Data Flow Diagram**

The DFD takes an input-process-output view of a system. That is, data objects flow into the software, are transformed by the processing elements, and resultant data objects flow out of the software. Data objects are represented by labeled arrows and transformations are represented by circles. The DFD is represented in a hierarchical fashion. The first DFD represents the system as a whole. Subsequent data flow diagrams provide increasing detail with each subsequent level.

The data flow diagram enables the software engineer to develop models of the information domain and functional domain at the same time. As the DFD is refined into levels of greater detail, the analysts perform an implicit functional decomposition of the system.



# **FIG 1. DATA FLOW DIAGRAM**

# **Chapter 4**

# **Implementation**

* 1. **Frontend Description**

## HTML

HTML (Hypertext Markup Language) is the set of markup symbols or codes inserted in a file intended for display on a World Wide Web browser page. The markup tells the Web browser how to display a Web page's words and images for the user. Each individual markup code is referred to as an element (but many people also refer to it as a tag). Some elements come in pairs that indicate when some display effect is to begin and when it isto end.

## CASCADING STYLE SHEET (CSS)

Cascading Style Sheets (CSS) are a collection of rules we use to define and modify web pages. CSS are similar to styles in Word. CSS allow Web designers to have much more control over their pages look and layout. For instance, you could create a style that defines the body text to be Verdana, 10 point. Later on, you may easily change the body text to Times New Roman, 12 point by just changing the rule in the CSS. Instead of having to change the font on each page of your website, all you need to do is redefine the style on the style sheet, and it will instantly change on all of the pages that the style sheet has been applied to. With HTML styles, the font change would be applied to each instance of that font and have to be changed in each spot.

CSS can control the placement of text and objects on your pages as well as the look of those objects.

HTML information creates the objects (or gives objects meaning), but styles describe how the objects should appear. The HTML gives your page structure,

while the CSS creates the “presentation”. An external CSS is really just a text file with a css extension. These files can be created with Dreamweaver, a CSS editor, or even Notepad.

**BOOTSTRAP**

## Using Bootstrap as a module

We provide a version of Bootstrap built as ESM (bootstrap.esm.js and bootstrap.esm.min.js) which allows you to use Bootstrap as a module in your browser, if your [targeted browsers support it](https://caniuse.com/es6-module).

Copy

<script type="module">

import { Toast } from 'bootstrap.esm.min.js'

Array.from(document.querySelectorAll('.toast'))

.forEach(toastNode => new Toast(toastNode))

</script>

## Dependencies

Some plugins and CSS components depend on other plugins. If you include plugins individually, make sure to check for these dependencies in the docs.

Our dropdowns, popovers and tooltips also depend on [Popper](https://popper.js.org/).

## Events

Bootstrap provides custom events for most plugins' unique actions. Generally, these come in an infinitive and past participle form - where the infinitive (ex. show) is triggered at the start of an event, and its past participle form (ex. shown) is triggered on the completion of an action.

All infinitive events provide [preventDefault()](https://developer.mozilla.org/en-US/docs/Web/API/Event/preventDefault) functionality. This provides the ability to stop the execution of an action before it starts. Returning false from an event handler will also automatically call preventDefault().

Copy

var myModal = document.getElementById('myModal')

myModal.addEventListener('show.bs.modal', function (event) {

if (!data) {

return event.preventDefault() // stops modal from being shown

}

})

# **Display property**

Quickly and responsively toggle the display value of components and more with our display utilities. Includes support for some of the more common values, as well as some extras for controlling display when printing.

* [How it works](https://getbootstrap.com/docs/5.0/utilities/display/#how-it-works)
* [Notation](https://getbootstrap.com/docs/5.0/utilities/display/#notation)
* [Examples](https://getbootstrap.com/docs/5.0/utilities/display/#examples)
* [Hiding elements](https://getbootstrap.com/docs/5.0/utilities/display/#hiding-elements)
* [Display in print](https://getbootstrap.com/docs/5.0/utilities/display/#display-in-print)
* [Sass](https://getbootstrap.com/docs/5.0/utilities/display/#sass)
  + [Utilities API](https://getbootstrap.com/docs/5.0/utilities/display/#utilities-api)

## How it works

Change the value of the [display property](https://developer.mozilla.org/en-US/docs/Web/CSS/display) with our responsive display utility classes. We purposely support only a subset of all possible values for display. Classes can be combined for various effects as you need.

## Notation

Display utility classes that apply to all [breakpoints](https://getbootstrap.com/docs/5.0/layout/breakpoints/), from xs to xxl, have no breakpoint abbreviation in them. This is because those classes are applied from min-width: 0; and up, and thus are not bound by a media query. The remaining breakpoints, however, do include a breakpoint abbreviation.

As such, the classes are named using the format:

* .d-{value} for xs
* .d-{breakpoint}-{value} for sm, md, lg, xl, and xxl.

Where value is one of:

* none
* inline
* inline-block
* block
* grid
* table
* table-cell
* table-row
* flex
* inline-flex

The display values can be altered by changing the $displays variable and recompiling the SCSS.

The media queries affect screen widths with the given breakpoint or larger. For example, .d-lg-none sets display: none; on lg, xl, and xxl screens.

## Hiding elements

For faster mobile-friendly development, use responsive display classes for showing and hiding elements by device. Avoid creating entirely different versions of the same site, instead hide elements responsively for each screen size.

To hide elements simply use the .d-none class or one of the .d-{sm,md,lg,xl,xxl}-none classes for any responsive screen variation.

To show an element only on a given interval of screen sizes you can combine one .d-\*-none class with a .d-\*-\* class, for example .d-none .d-md-block .d-xl-none .d-xxl-none will hide the element for all screen sizes except on medium and large devices.

## Background gradient

By adding a .bg-gradient class, a linear gradient is added as background image to the backgrounds. This gradient starts with a semi-transparent white which fades out to the bottom.

Do you need a gradient in your custom CSS? Just add background-image: var(--bs-gradient);.

.bg-primary.bg-gradient

# **4.2Backend Development**

## Language Used: - Python

Python is a widely used general-purpose, high level programming language. It wasinitially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems moreefficiently.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

## Django

Django is a high-level Python web framework that encourages rapid development and clean,pragmatic design. Built by experienced developers, it takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel.It is free and open source, has a thriving

and active community, great documentation,and many options for free and paid- for support.

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build an application after collecting client requirement.

# **How to use sessions**

Django provides full support for anonymous sessions. The session framework lets you store and retrieve arbitrary data on a per-site-visitor basis. It stores data on the server side and abstracts the sending and receiving of cookies. Cookies contain a session ID – not the data itself (unless you’re using the [cookie based backend](https://docs.djangoproject.com/en/4.2/topics/http/sessions/#cookie-session-backend)).

## Enabling sessions

Sessions are implemented via a piece of [middleware](https://docs.djangoproject.com/en/4.2/ref/middleware/).

To enable session functionality, do the following:

* Edit the [MIDDLEWARE](https://docs.djangoproject.com/en/4.2/ref/settings/#std-setting-MIDDLEWARE) setting and make sure it contains 'django.contrib.sessions.middleware.SessionMiddleware'. The default settings.py created by django-admin startproject has SessionMiddleware activated.

If you don’t want to use sessions, you might as well remove the SessionMiddleware line from [MIDDLEWARE](https://docs.djangoproject.com/en/4.2/ref/settings/#std-setting-MIDDLEWARE) and 'django.contrib.sessions' from your [INSTALLED\_APPS](https://docs.djangoproject.com/en/4.2/ref/settings/#std-setting-INSTALLED_APPS). It’ll save you a small bit of overhead.

## Configuring the session engine

By default, Django stores sessions in your database (using the model django.contrib.sessions.models.Session). Though this is convenient, in some setups it’s faster to store session data elsewhere, so Django can be configured to store session data on your filesystem or in your cache.

### Using database-backed sessions

If you want to use a database-backed session, you need to add 'django.contrib.sessions' to your [INSTALLED\_APPS](https://docs.djangoproject.com/en/4.2/ref/settings/#std-setting-INSTALLED_APPS) setting.

Once you have configured your installation, run manage.py migrate to install the single database table that stores session data.

### Using cached sessions

For better performance, you may want to use a cache-based session backend.

To store session data using Django’s cache system, you’ll first need to make sure you’ve configured your cache; see the [cache documentation](https://docs.djangoproject.com/en/4.2/topics/cache/) for details.

### Django helps you write software that is:

**Complete**

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything you need is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and [up-to-date documentation](https://docs.djangoproject.com/en/stable/).

### Versatile

Django can be (and has been) used to build almost any type of website — from content management systems and wikis, to social networks and news sites. It can work with any client-side framework and can deliver content in almost any format (including HTML, RSS feeds, JSON, XML, etc).

Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

### Secure

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the website automatically. For example, Django provides a secure way to manage

user accounts and passwords, avoiding common mistakes like putting session information in cookies where it is vulnerable (instead cookies just contain a key, and the actual data is stored in the database) or directly storing passwords rather than a password hash.

A password hash is a fixed-length value created by sending the password through a [cryptographic hash function.](https://en.wikipedia.org/wiki/Cryptographic_hash_function) Django can check if an entered password is correct by running it through the hash function and comparing the output to the stored hash value. However due to the "one-way" nature of the function, even if a stored hash value is compromised it is hard for an attacker to work out the original password.

Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and [clickjacking](https://developer.mozilla.org/en-US/docs/Glossary/Clickjacking) (see [Website](https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Website_security) security for more details of such attacks).

### Scalable

Django uses a component-based "[shared-nothing](https://en.wikipedia.org/wiki/Shared_nothing_architecture)" architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed).Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two).

### Maintainable

Django code is written using design principles and patterns that encourage the creation ofmaintainable and reusable code. In particular, it makes use of the Don't Repeat Yourself (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related

functionality into reusable "applications" and, at a lower level, groups related code into modules (along the lines of the [Model](https://developer.mozilla.org/en-US/docs/Glossary/MVC)View Controller (MVC) pattern).

### Portable

Django is written in Python, which runs on many platforms. That means that you are not tied to any particular server platform, and can run your applications on many flavors of Linux, Windows, and macOS. Furthermore, Django is well- supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django sites.

## SQLite

SQLite3 can be integrated with Python using sqlite3 module, which was written by Gerhard Haring. It provides an SQL interface compliant with the DB-API

* 1. specification described by PEP 249. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards.

To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements.

## 1. Initial Setup

There aren’t any prerequisites to getting started (other than a Mac with iMessage). However, your mileage may vary depending on whether you are using a SQL client (I use [TablePlus](https://tableplus.com/)) or [sqlite3](https://sqlite.org/cli.html) on the command line. The setup for both is similar.

The database is located at ~/Library/Messages/chat.db. If you open a terminal and run ls ~/Library/Messages/chat.db, you’ll get a message like: ls: Messages: Operation not permitted. Similarly, if you try and open the database in a SQL client, you’ll likely see an error. We need to grant access first.

I’m doing this on macOS Catalina, which has added some additional security features (i.e., restrictions). Even so, it’s actually quite easy to get access to the database, as long as you don’t mind granting full disk access to some applications. Let’s get that out of the way right away.

Open System Preferences, find the “Security & Privacy” pane, click on the “Privacy” tab, and find the “Full Disk Access” item. Make sure your SQL client and/or terminal are selected.

That should be it! You’ll need to restart any applications that were open while you granted them access. Otherwise, you should be able to access the message database:

In a terminal, run sqlite3 ~/Library/Messages/chat.db

In a SQL client, connect to the database at ~/Library/Messages/chat.db. If you’re having trouble navigating to it, you can press CMD+SHIFT+G in a finder window and type that in the dropdown.

# **Chapter 5**

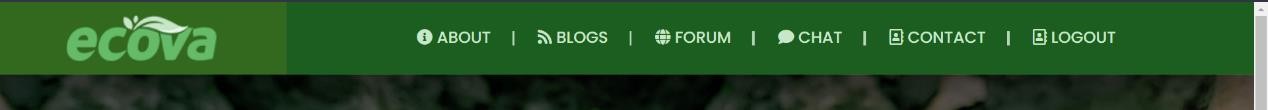
# **Results & Discussion**

# **Snapshots of the Project:**



* + 1. The homepage of grow green, the user home page includes a navigation bar at the top for easy access to different sections. The home section welcomes the user and sets the tone for the website. The recent posts section displays a collection of recent blog posts using Bootstrap cards or any desired layout. Finally, the footer provides basic copyright information.

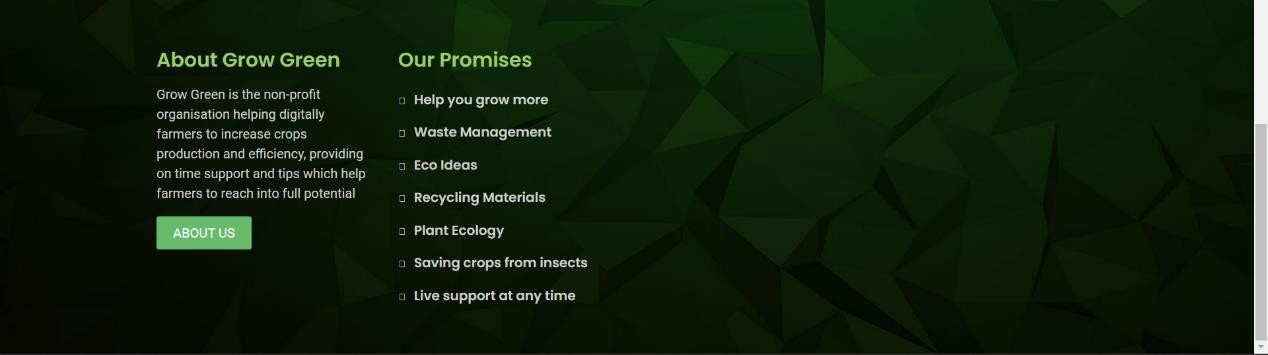
**FIG 2. User Home Page**



* + 1. The navigation bar is created using the Bootstrap `navbar` class. The `navbar-brand` represents the website's brand or logo. The navigation links are structured using `ul` and `li` elements within the `navbar-nav` class. You can customize the links and add additional ones as per your website's requirements.

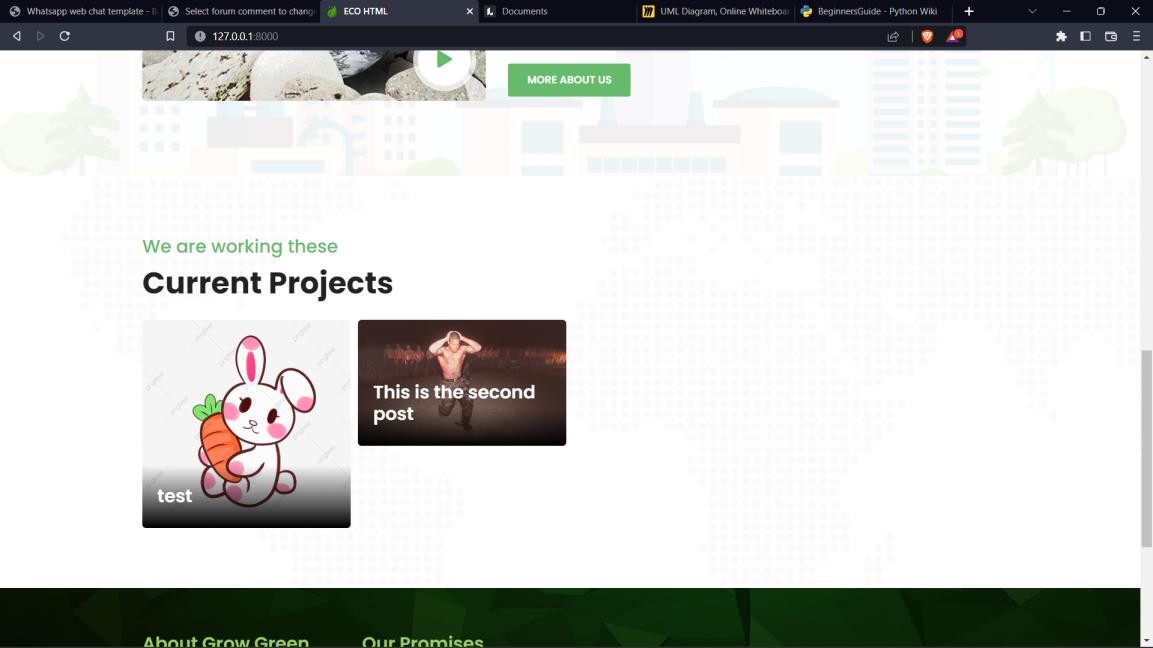
**FIG 2. Navigation Bar**

**3.Footer**



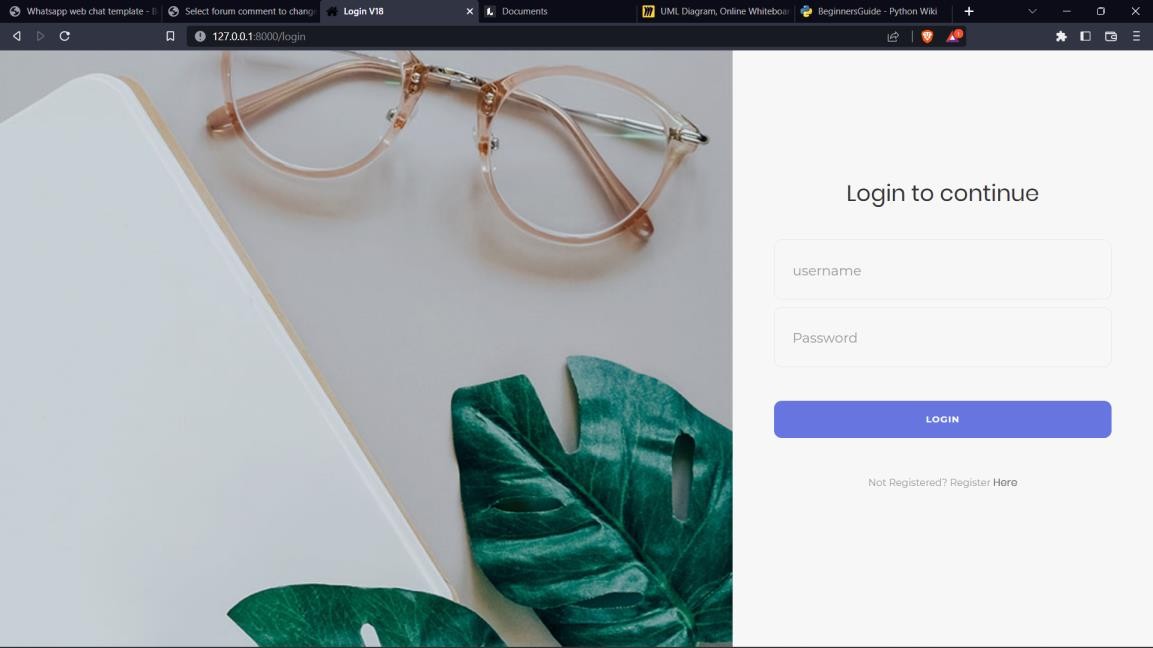
Footer: The footer is divided into two columns using the Bootstrap `col-md-6` class within a `row`. The first column contains information about the website or company, including a brief description and a link to read more about it. The second column provides contact details, such as the address, email, and phone number, along with a link to get in touch.

**FIG 2. Footer**



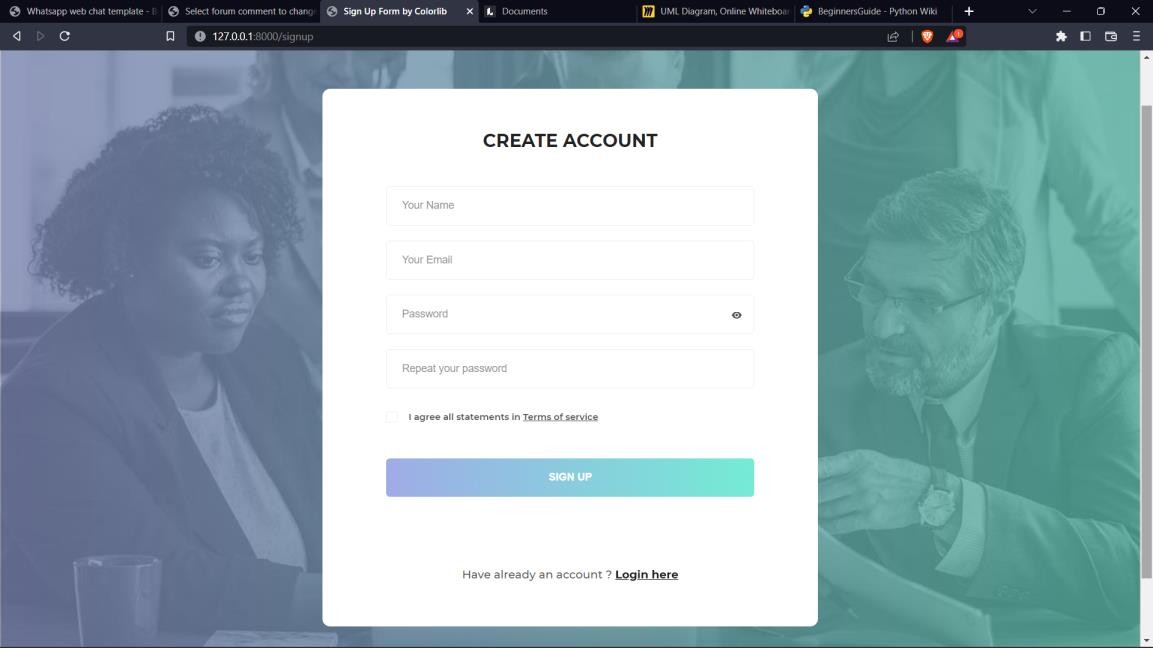
Moving down you can find out the recent blog posts on which you can click and get into detail page of the blog post.

**FIG 4. Blog Post**



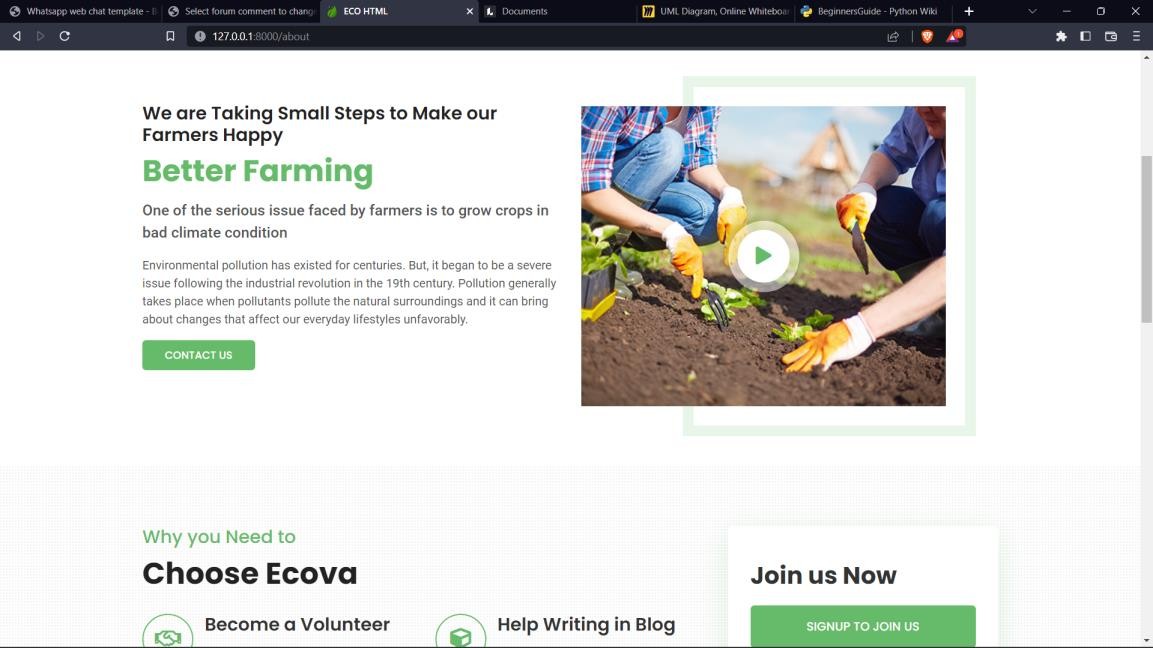
Login Page: The login form includes fields for the email address and password, with corresponding `input` elements of type `email` and `password`. The form is wrapped in a `form` element, and the login button has the class `btn btn-primary`.

**FIG 5. Sign In Page**

6. Sign Up Page

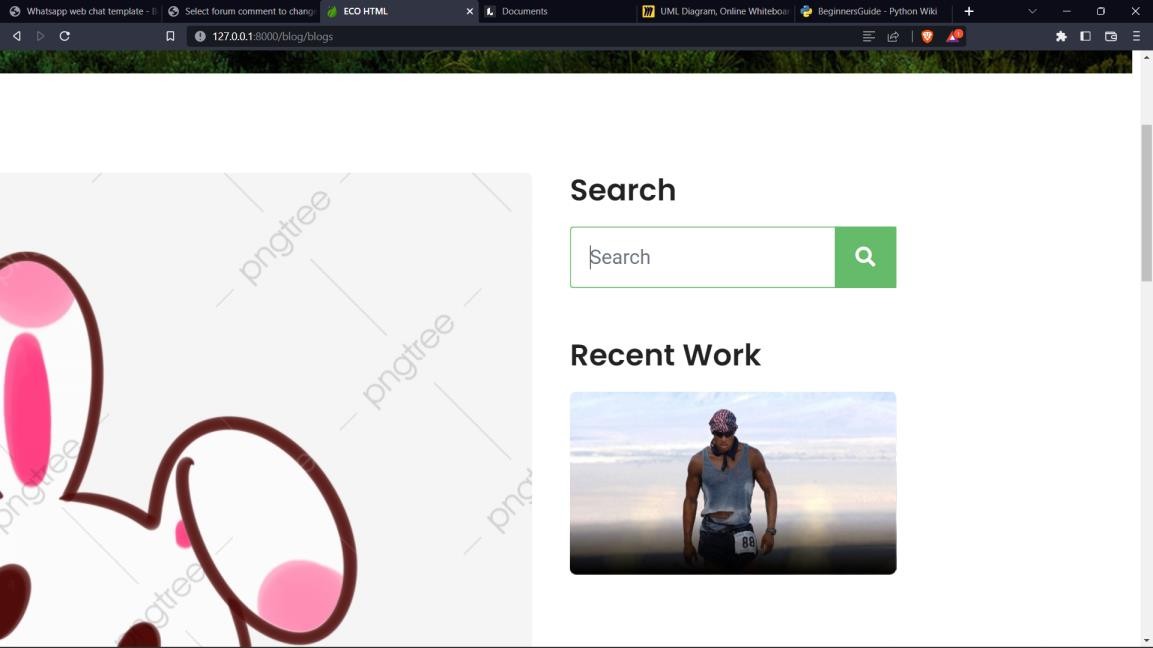
Sign up: If user is not already registered than he/she can create new account from the sign up page.

**FIG 6. Sign Up Page**



About Page

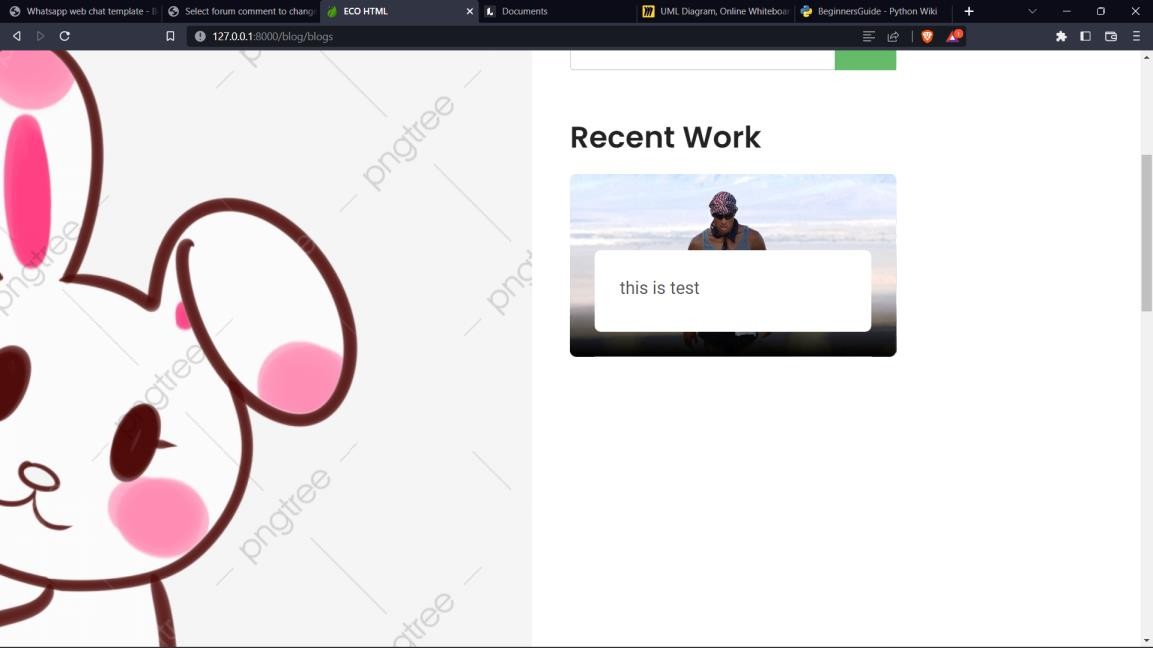
**FIG 7. About Page**



There’s also one option of search where you can search for the other blogs.

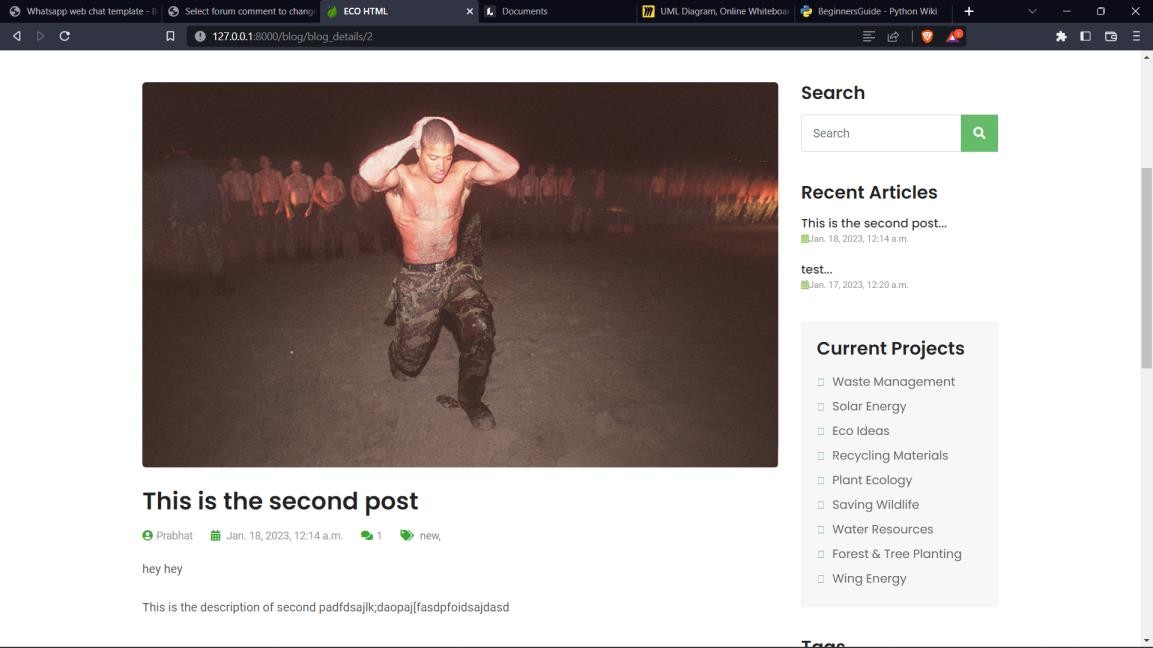
**FIG 8. Blog Page**

Blog Page: It shows the list of blog post.



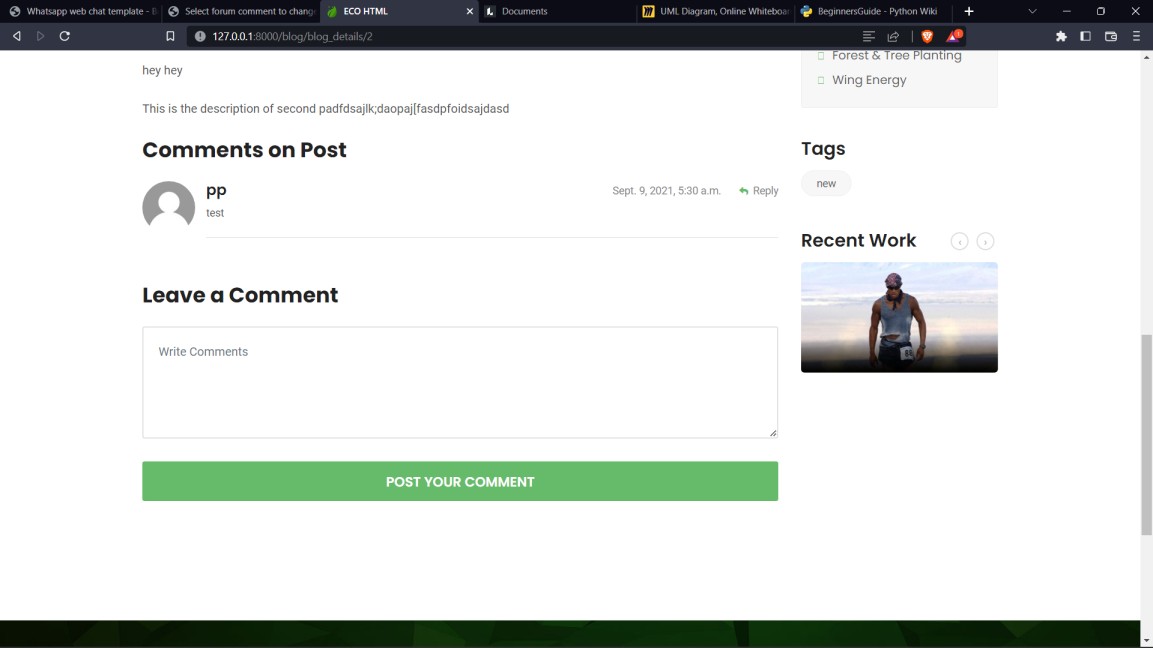
**FIG 9. Search Bar**

Create and Edit Blog Posts: Admins and authorized users can create new blog posts or edit existing ones. This feature usually includes a text editor or content management system to write and format blog content, add images, and set categories and tags.

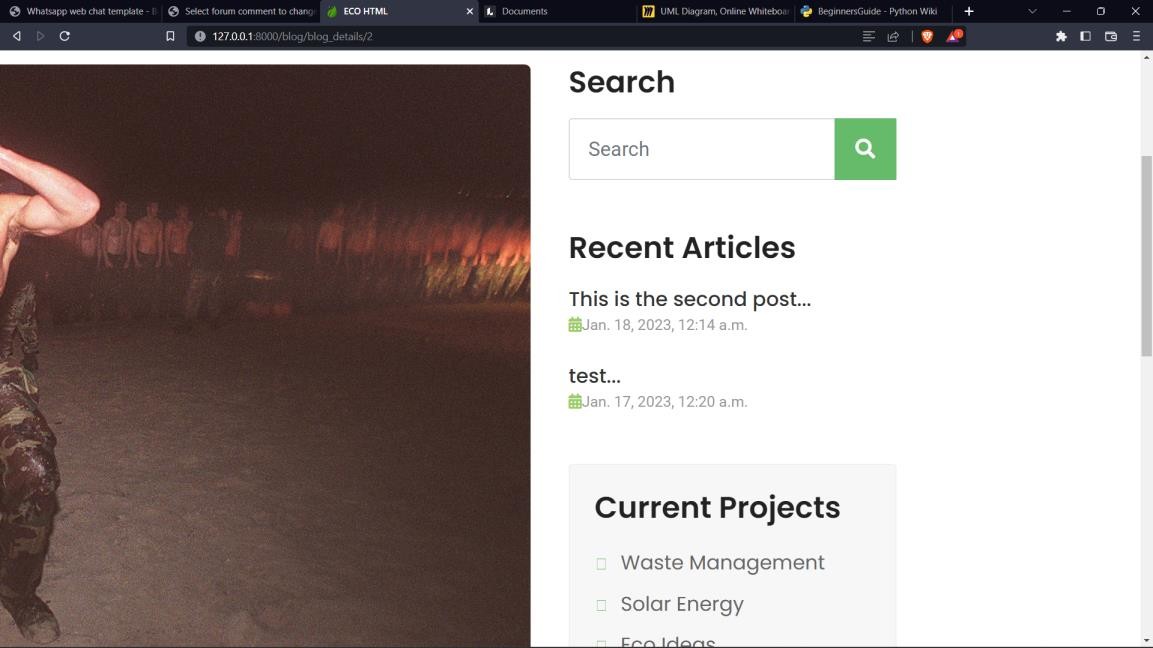


Blog detailed page: Here you can see a blog post in detail by selecting the read post button on one the blog from blog list page.

**FIG 10. Blog detail Page**

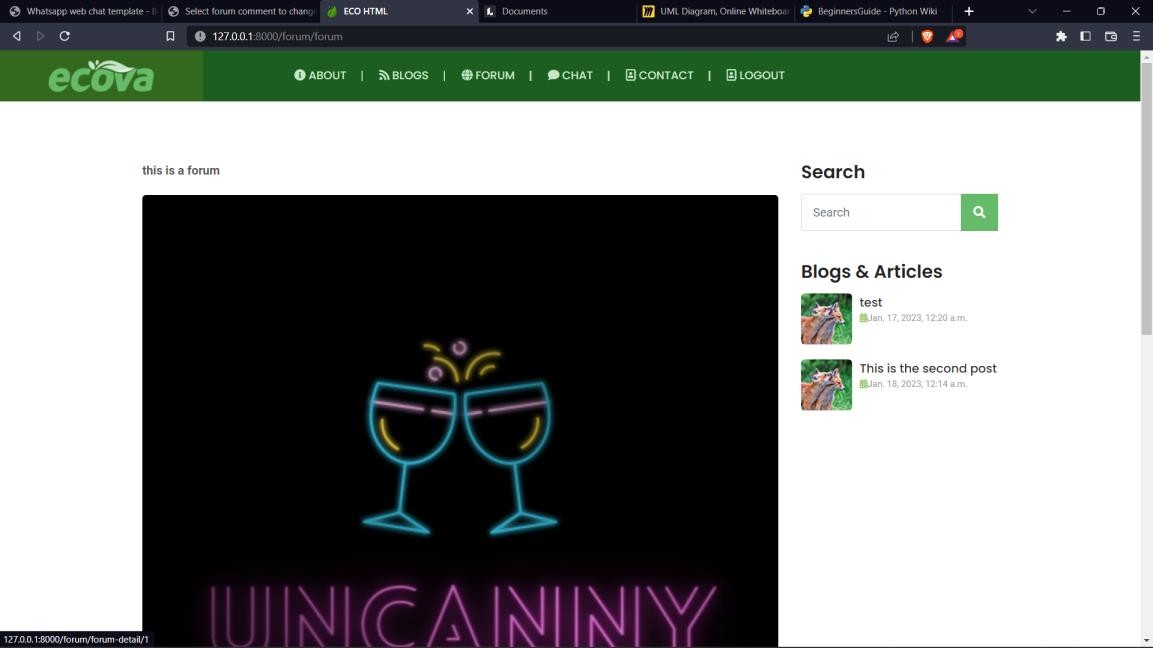


After loggin in you will get the option of commenting, One of the quality of this project which I would like to mention is this that the process of commenting is very smooth and seamless as it uses the **AJAX** technology.



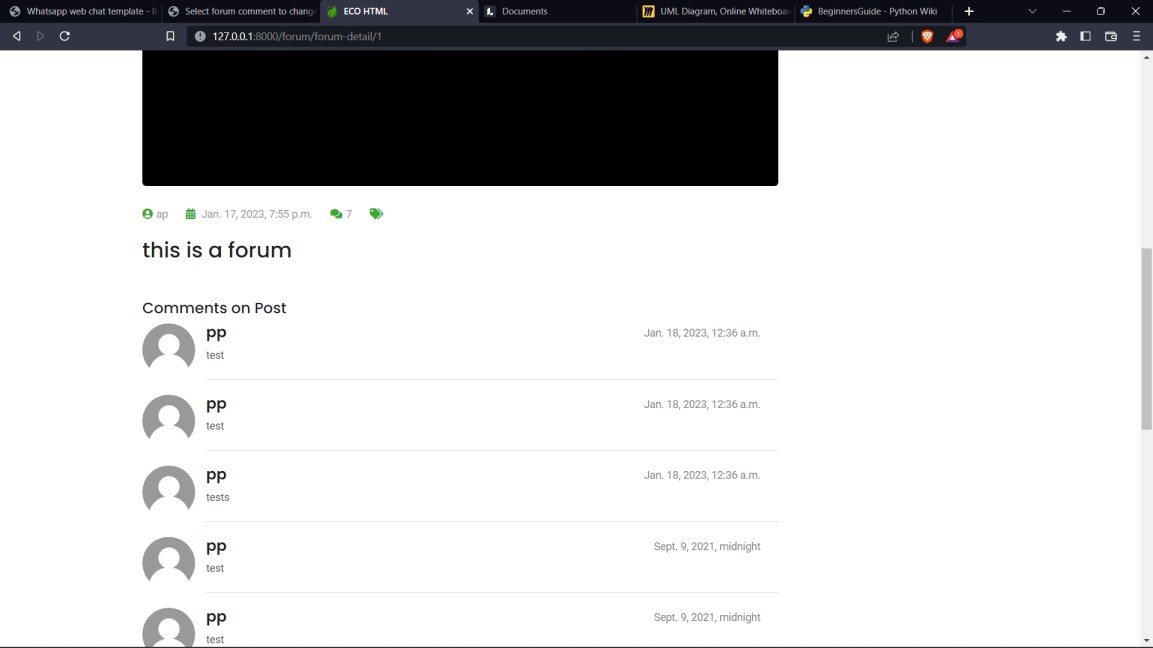
Similar to the blog list page you can also search here. Also, you can look through the recent articl

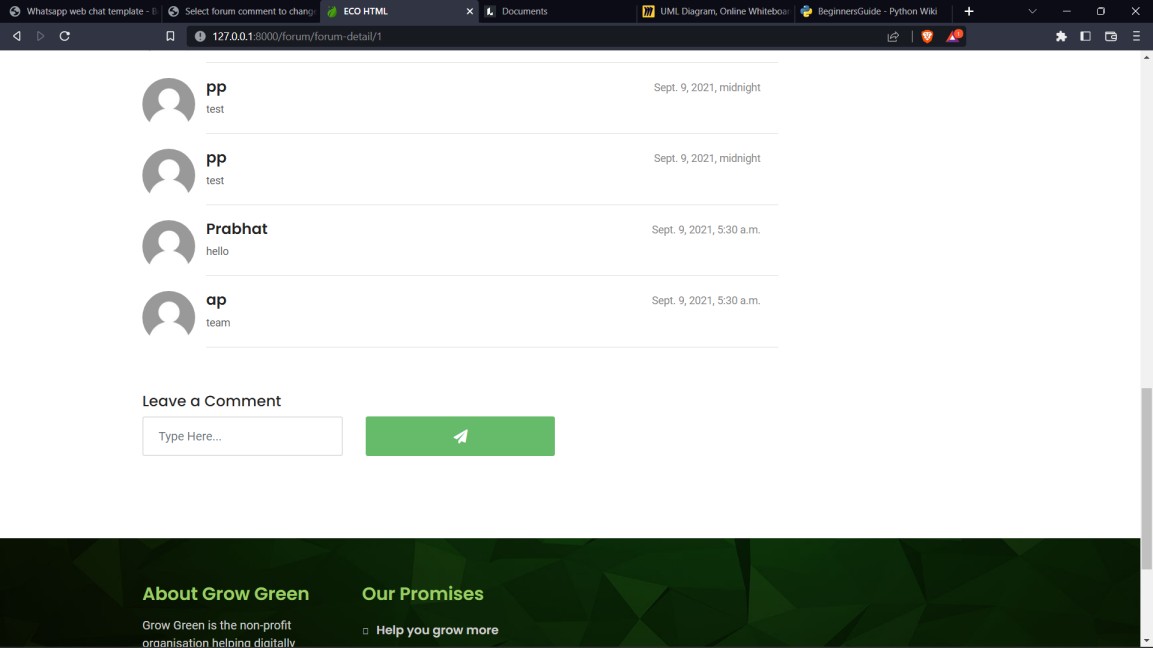
Comment Management: Admins can moderate and manage comments posted on blog posts, including approving, deleting, or flagging comments for review.



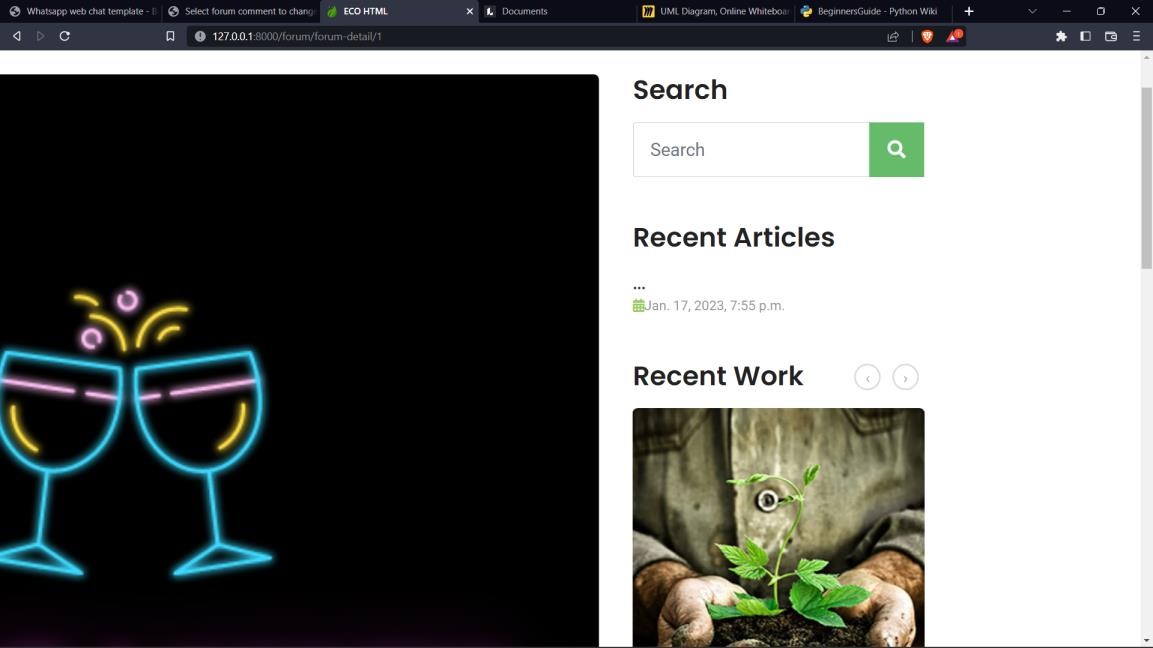
**FIG 11.Forum list page.**

The forum discussion titles are clickable links (`<a href="forum-detail.html">`) that direct users to the detail page of each specific forum discussion. You can replace `"forum-detail.html"` with the appropriate URL or file path of the detail page for each discussion.



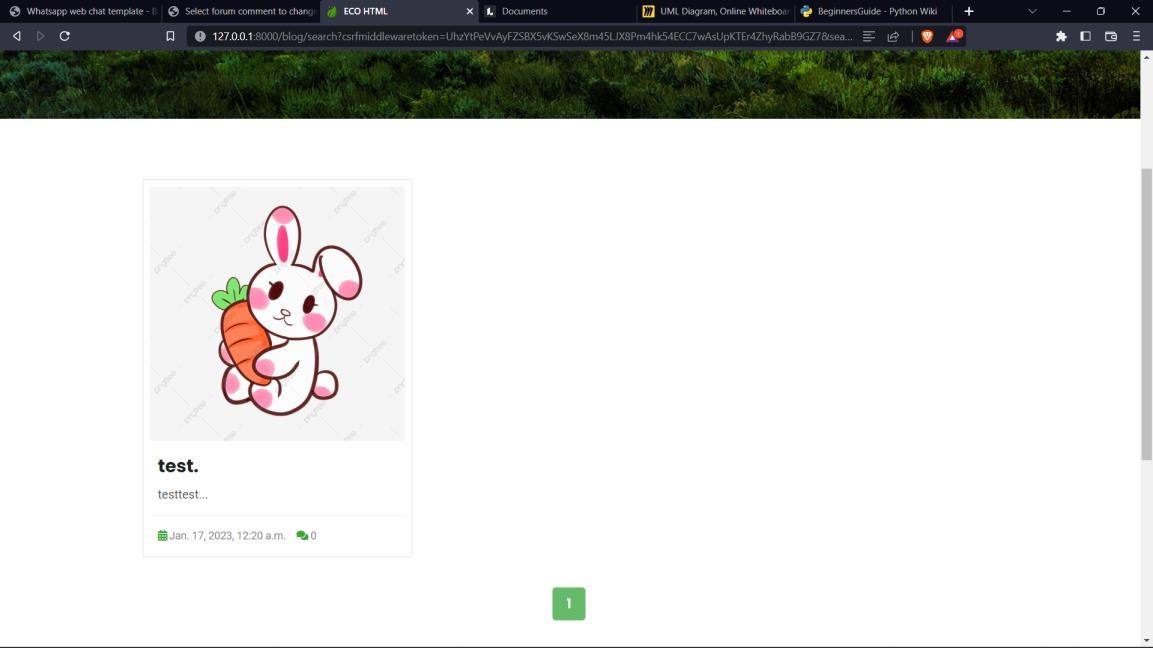


Forum detail post page along with comments and option of sending a comment. Again it is necessary to be an authenticated user inorder to comment.

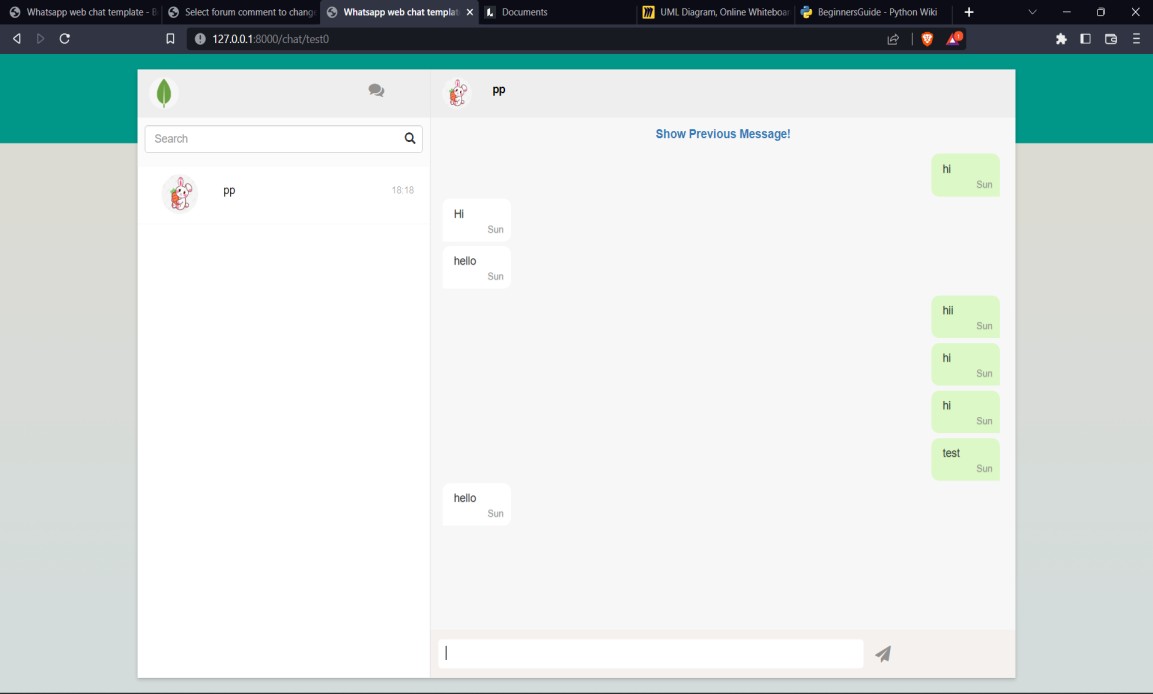


Create and Edit Blog Posts: Admins and authorized users can create new blog posts or edit existing ones. This feature usually includes a text editor or content management system to write and format blog content, add images, and set categories and tags.

Manage Categories and Tags: Admins can create, edit, and delete blog categories and tags to organize and classify blog posts for easy navigation and search.



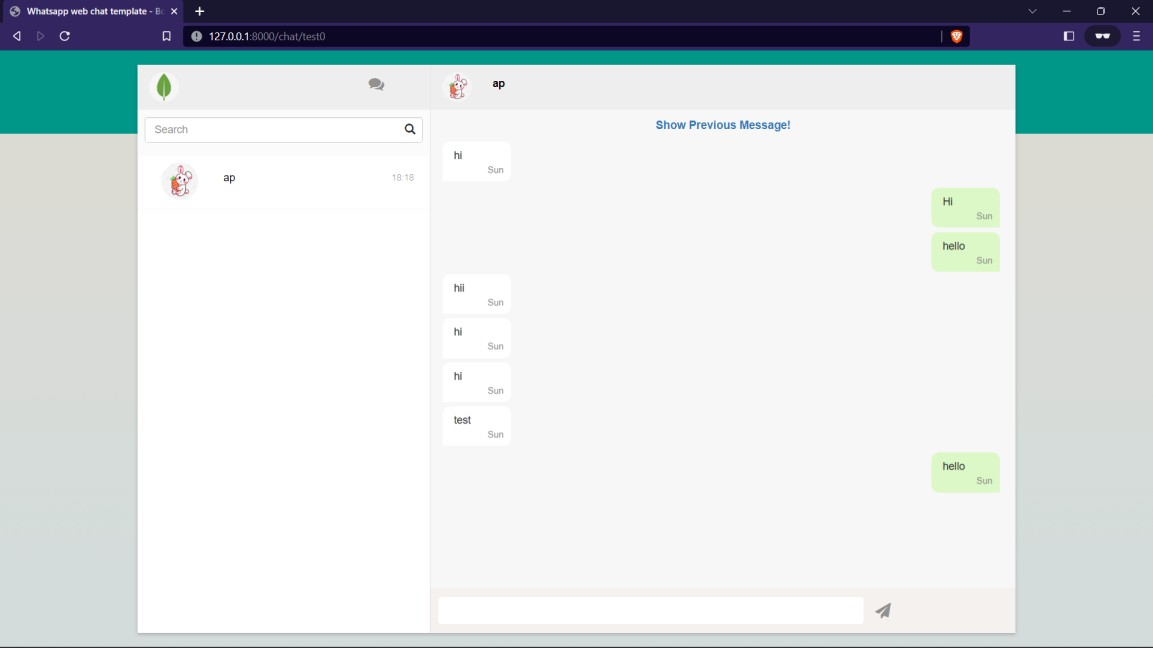
**FIG 12. Search result page.**



User Chat: Users can access and participate in chat discussions with other users. This feature allows real-time messaging, private conversations, and group chats.

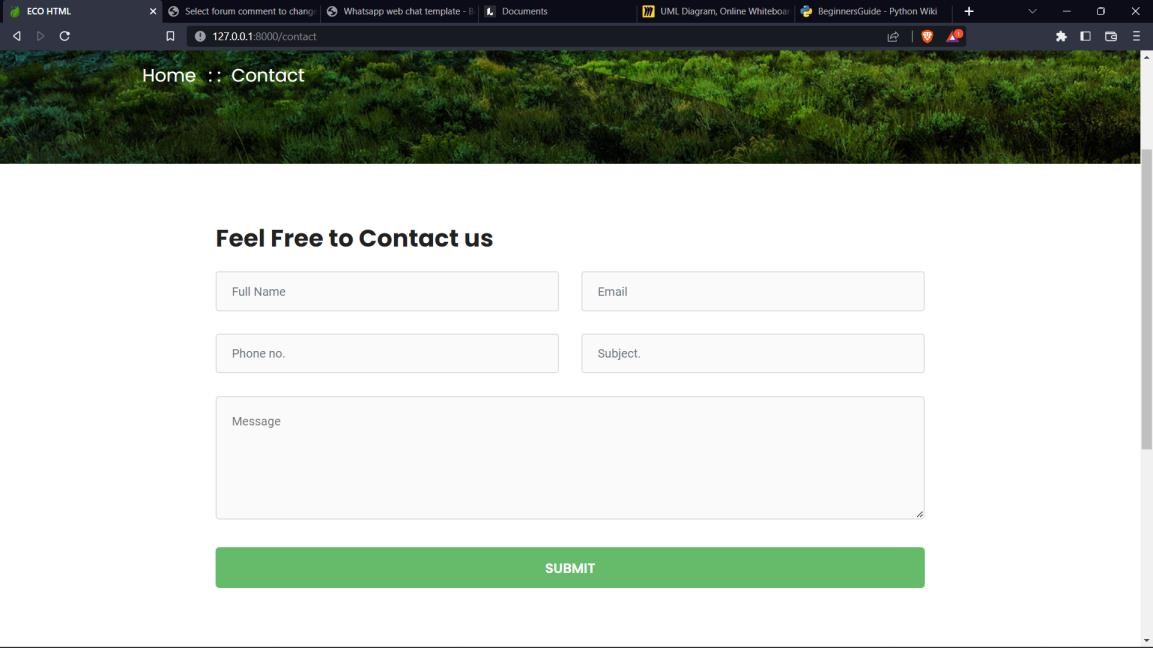
Chat Moderation: Admins have the ability to monitor and moderate chat activities, ensuring compliance with community guidelines and taking action against any inappropriate behavior or content.

**FIG 13.USER A CHAT**



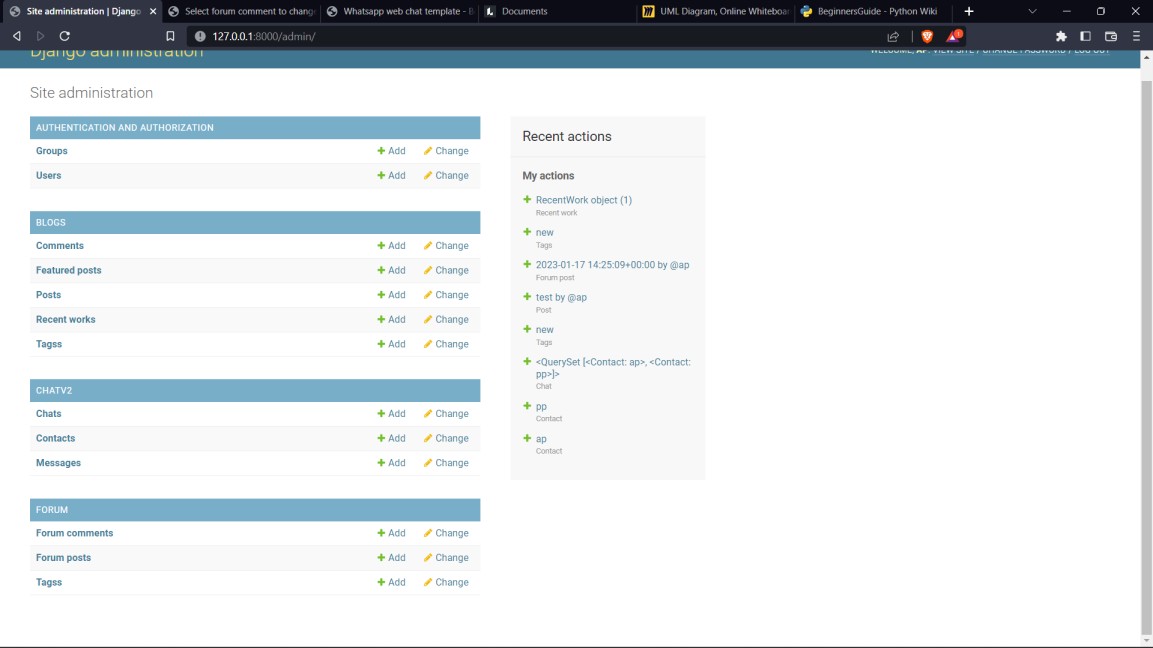
F**IG 14. User B CHAT**

Chat page where you can chat end to end with the users.



**FIG 15. Contact page**

Contact Page: Whenever a user contact a mail sent to the administrator.



**FIG 16. Admin/User Panel**

The Admin/User Panel of a chat and blog website provides an interface for administrators and users to manage and interact with various features and settings of the website. Here's an example of how the Admin/User Panel can be structured and what functionalities it can offer:

# **Limitation**

**Current Limitations of Grow Green Website:**

Besides the above achievements and the successful completion of the project, we still feel the project has some limitations, listed below:

1. It is not a large-scale system.
2. Only limited information is provided by this system.
3. Since it is an online project, customers need an internet connection to buy products.
4. The size of the database increases day by day, increasing the load on the database backup and data maintenance activity.
5. Training for simple computer operations is necessary for the users working on the system.

**CHALLENGES**

1. User Engagement: Encouraging active participation and engagement from users can be a challenge. Building a community and fostering discussions require effective moderation and incentives to keep users interested and motivated to contribute.
2. Content Quality and Moderation: Maintaining content quality and preventing spam, trolling, or offensive material is crucial for the reputation and user experience of the website. Implementing robust moderation tools and techniques to filter out inappropriate content can be challenging, especially as the volume of user-generated content increases.
3. Security and Privacy: Websites need to ensure the security of user data and protect against hacking attempts, data breaches, or unauthorized access. Implementing secure authentication mechanisms, encryption protocols, and regular security audits are essential.

**FUTURE PLAN**

The proposed software product is useful for farmers and farming-related technical enthusiasts. As we know most of the people of India do farming and also India is a great producer of different crops and spices and also unfortunately many of them are not that literate to grow crops efficiently and reach their full potential.

# Chapter 6

**CONCLUSION AND RECOMMENDATIONS**

The project is created for our Farmer brothers and we expect and believe that It will deliver the service. This is the base version or the very first version of the Grow Green. We are looking forward to the user feedback in order to improve this website more and will release the new versions.

It is a very reliable website with a low down rate and high security. We implemented the industry-level standards while developing this website and the framework that we have used also uses the industry-level standards as well as it is widely used in giant industries such as Meta, and Reddit.

It ensures a great level of security and reliability along with the best practices and scalability. While developing this application we have scalability in our minds and hence this application ensures the ease of upgradability and scalability.

# **Chapter 7**

**Future Scope**

There’s a variety of things which we can enhance in future like according to user’s usage and load we can scale our application.

Also, there’s always this flexibility available with Django to quickly put new features or updates for the software, because of relying on Django.

The proposed software product is useful for farmers and farming-related technical enthusiasts. As we know most of the people of India do farming and also India is a great producer of different crops and spices and also unfortunately many of them are not that literate to grow crops efficiently and reach their full potential.

This product aims to fill this gape of knowledge by providing the right knowledge to the right person by using personalized results. It gives the farmers ability to communicate which help them sort out their problem-related crops and production and also provides them tips and knowledge to produce more effectively/efficiently.

Also, these quality in turn provides a reduction in cost and increase in the output of production.

**BIBLIOGRAPHY**

* https://github.com/Prax583/GrowGreen.git
* <https://www.udemy.com/course/the-complete-web-development-bootcamp/learn/lecture/12638830?start=0#overview>
* <https://www.youtube.com/watch?v=mrEcfu-ByDw&list=PL-Jc9J83PIiEeD3I4VXETPDmzJ3Z1rWb4>
* <https://en.wikiversity.org/wiki/Web_Design%2FAn_Introduction_to_Programming_with_JavaScript>
* <https://www.javatpoint.com/python-mongodb>
* <https://tailwindcss.com/docs/responsive-design>
* <https://getbootstrap.com/docs/5.0/getting-started/contents/>