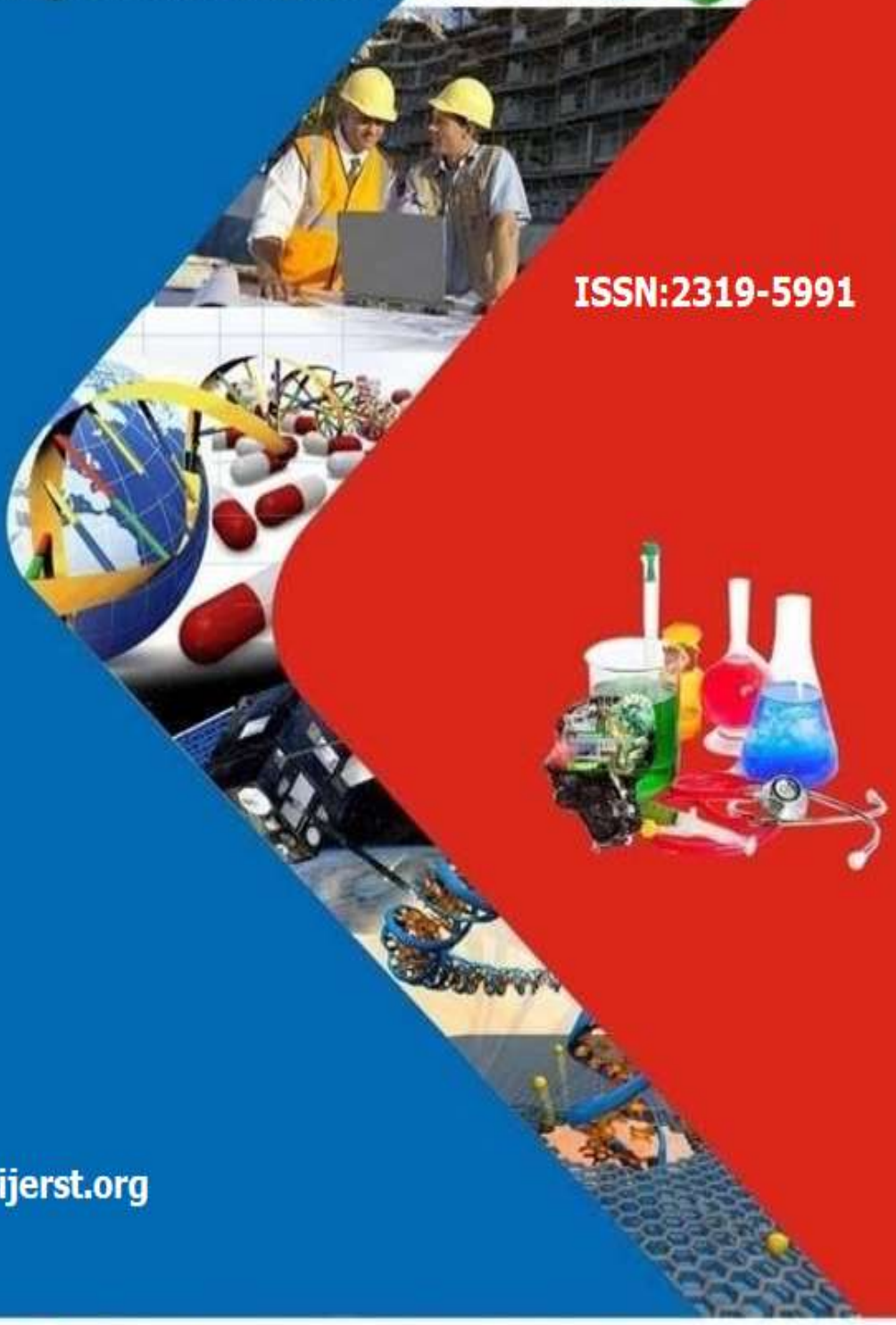
International Journal of Engineering Research and Science & Technology

IJERST



ISSN:2319-5991

www.ijerst.org

E-mail: editor@ijerst.org or ijerst.editor@gmail.com

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

CREATING A WEBSITE TO SHARE TIPS AND RESOURCES FOR SUSTAINABLE LIVING WITH SQL

Dr. G.N. V Vibhav Reddy[1*], M. Munilal[2], Gariganti Abhishek[2], K. Thirupathi[2], Shinde Lokesh[2]
[1,2]Department of Computer Science and Engineering, Sree Dattha Institute of Engineering and Science, Sheriguda, Ibrahimpatnam, 501510, Telangana
*Corresponding author: Dr. G.N.V Vibhav Reddy

ABSTRACT
Developing a website for sharing gardening tips and resources involves creating a platform where users can access gardening information, share tips, and participate in community discussions. Traditional methods for sharing gardening tips included printed publications, gardening forums, and local workshops, which were limited in scope and accessibility. These methods often lacked real-time updates, comprehensive gardening resources, and interactive features for community engagement. Limitations included delayed information dissemination, inconsistent content management, and challenges in fostering widespread participation and knowledge sharing among gardening enthusiasts. Thus, this work implements an integrated, user-friendly website that facilitates efficient sharing of gardening tips and resources. This application offers users the ability to access gardening articles, contribute tips and advice, participate in discussions, and access gardening tools and resources. In addition, integration with an SQL backend ensures secure data storage, enabling reliable management of user-generated content, community interactions, and administrative data. The significance of designing a website with SQL integration lies in its ability to provide a robust, scalable, and interactive solution for gardening enthusiasts. An SQL backend supports efficient data storage, streamlined content retrieval, and effective management of user interactions and contributions. This integration facilitates real-time updates, enhanced community engagement, and personalized gardening experiences through tailored content and collaborative features. By transitioning to a modern, integrated web-based system, gardening enthusiasts can benefit from enhanced access to gardening knowledge, increased community participation, and improved collaboration, ultimately fostering a vibrant and supportive online gardening community.

Keywords: Resource Sharing, Django Framework, CRUD Operations, SQL Integration, Web Application Development

1. INTRODUCTION

The global gardening market has seen significant growth, with a market value projected to reach over $100 billion by 2027. In the United States alone, more than 77% of households engage in gardening activities, reflecting a widespread interest in this hobby. Online gardening communities have become increasingly popular, with millions of users participating in forums and social media groups dedicated to sharing gardening tips and resources. This growing interest underscores the need for an integrated platform that can provide comprehensive gardening information, facilitate community engagement, and support real-time updates. Developing a website using the Django framework with SQL integration offers a robust solution to meet these needs, leveraging modern technology to enhance the gardening experience for users worldwide Traditional methods for sharing gardening tips and resources, such as printed publications, gardening forums, and local workshops, face several limitations. These methods often suffer from delayed information dissemination, inconsistent content management, and limited accessibility. Furthermore, they lack interactive features that can foster community engagement and real-time updates. The absence of a centralized, user-friendly platform hinders widespread participation and knowledge sharing among gardening enthusiasts. This creates a need for a modern solution that can address these limitations, providing a scalable and efficient way to share gardening information and foster community interaction.

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024



Fig 1: Django framework

The primary objective of the research is to develop a user-friendly website using the Django framework and SQL integration that facilitates the efficient sharing of gardening tips and resources. The website aims to provide users with access to a wealth of gardening articles, enable them to contribute tips and advice, and participate in community discussions. The platform will also offer gardening tools and resources to enhance the user experience. By integrating an SQL backend, the website will ensure secure data storage, reliable management of user-generated content, and real-time updates. The ultimate goal is to create a robust, scalable, and interactive platform that fosters a vibrant online gardening community.

2. LITERATURE SURVEY

Nishath Shireen and Banita Devi. [1] Smart garden management system. 2018.This paper discusses a smart garden management system, emphasizing automation and monitoring. Integrating HTML and CSS can enhance the user interface for managing the garden settings and monitoring data. Using Django for the backend provides robust support for managing data from sensors and user inputs, enabling efficient control and monitoring of garden parameters through a web interface. Norakmar binti Arbain Sulaiman and Muhamad Dan Darrawi bin Sadli [2] . An IoT-based smart garden with weather station system. In 2019 IEEE 9th Symposium on Computer Applications Industrial Electronics.The study presents an IoT-based smart garden system with a weather station. HTML and CSS can be used to create a user-friendly interface for displaying weather data and garden conditions. Django, with its powerful backend capabilities, can manage real-time data from IoT devices, providing users with accurate and timely information for garden management. J. Kumar, N. Gupta, A. Kumari, and S. Kumari [3] , "Automatic Plant Watering and Monitoring System Using Nodemcu," 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2019.This paper details an automatic plant watering and monitoring system. HTML and CSS can be utilized to develop an intuitive dashboard for monitoring plant health and watering schedules. Django can handle the backend processes, including data storage, user authentication, and interaction with the NodeMCU for automating watering tasks based on sensor data. G Kapinesh, Gayatri Konduru, V Thanikaiselvan [4], "A New Approach for Effective Garden Monitoring using IoT Applications", 2022 7th International Conference on Communication and Electronics Systems.The authors propose an effective garden monitoring system using IoT applications. HTML and CSS can

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

design interfaces for real-time monitoring and control of garden devices. Django provides a scalable and secure backend to manage IoT device data, user interactions, and control commands, ensuring efficient garden management. A. Sulaiman, M. Dan, S. Alam, and A. S. Garden [5] , "System", 2019 IEEE 9th Symp. Comput. Appl. Ind. Electron, pp. 38-43, 2019.This study describes a smart garden system integrating various technologies. HTML and CSS can create an engaging interface for users to interact with garden devices and view system status. Django can support the backend operations, including data handling from sensors, user management, and automation tasks, enhancing the overall functionality of the garden system.

M. Sheth and P. Rupani [6] , "Smart Gardening Automation using IoT with BLYNK App", Proc. Int. Conf. Trends Electron. Informatics ICOEI 2019, vol. 2019-April.The paper focuses on smart gardening automation using IoT and the BLYNK app. HTML and CSS can be used to develop a web-based interface for users who prefer desktop access. Django's backend capabilities can integrate with the BLYNK app, managing IoT data and providing additional functionalities such as data analytics and user-specific settings. S. N. Ishak et al., [7] "Smart home garden irrigation system using Raspberry Pi", 2017 IEEE 13th Malaysia Int. Conf. Commun. MICC 2017.This research presents a smart home garden irrigation system using Raspberry Pi. HTML and CSS can design a responsive web interface for system control and monitoring. Django, running on the Raspberry Pi, can handle backend tasks such as sensor data processing, user authentication, and irrigation control, providing a seamless user experience. S. Devika et al., [8] "Arduino based automatic plant watering system", International Journal of Advanced Research in Computer Science and Software Engineering.The paper discusses an Arduino-based automatic plant watering system. HTML and CSS can create a simple yet effective interface for monitoring and controlling the system. Django can manage the backend processes, including data collection from Arduino, scheduling watering tasks, and providing users with real-time updates and control options. S Sajithra Varun et al., [9] "An IOT Based Smart Monitoring and Controlling System for Garden", 2023 7th International Conference on Design Innovation for 3 Cs Compute Communicate Control.This study explores an IoT-based system for smart garden monitoring and control. HTML and CSS can be used to develop interactive dashboards for users to monitor garden conditions and control devices. Django offers a powerful backend to manage real-time data from IoT sensors, user authentication, and control commands, ensuring efficient garden management. Mahfuzulhoq Chowdhury and Alve Ahmad [10] , "Gardener: A Gardening Assistance Mobile Application With Plant Details, Plant Shopping, Expert Contact, Alert, and Plantation Tips Facilities", 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions.This paper presents Gardener, a mobile application providing various gardening assistance features. HTML and CSS can be used to create a web version of the application for wider accessibility. Django can handle the backend functionalities, including managing plant databases, user interactions, alerts, and expert consultations, providing a comprehensive gardening assistance platform.

3. PROPOSED SYSTEM

The project includes several core functionalities to manage user interaction and gardening content. The home function renders the main landing page of the site, providing users with an introduction to the platform. The registration function handles user sign-ups by processing POST request data, validating inputs, creating new accounts upon successful validation, and redirecting users to the login page. In case of errors like mismatched passwords or duplicate usernames, users are redirected back with appropriate error messages. The login_view function manages user login by authenticating credentials and logging users in if valid, otherwise returning them to the login page with an error notification. The logout_view allows users to securely log out and redirects them to the login screen. For administrators, the admin_login_view ensures only staff members can log in with admin rights,

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

validating their credentials and restricting access to non-admin users with redirection and error feedback. The garden_view provides a personalized dashboard where users can view gardening tips and their submitted gardening needs, as well as contribute new tips through a form interface. Lastly, the add_gardenplan function processes and stores new gardening plan submissions from users, saving the data to the database and updating the dashboard accordingly.

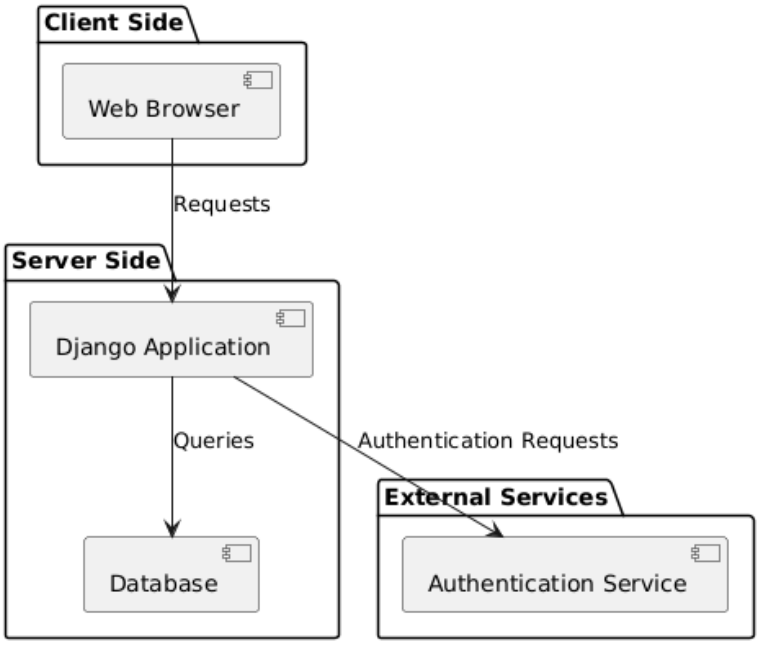


Figure 2: Architectural Block Diagram

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It follows the "batteries-included" philosophy, offering a comprehensive set of tools and features to build robust web applications. Django's primary aim is to simplify the development process by providing reusable components and a structured approach. It includes an ORM (Object-Relational Mapping) for database interactions, a built-in admin interface for managing application data, and support for user authentication, among other features. Django is known for its scalability, security, and the "don't repeat yourself" (DRY) principle, which helps developers avoid redundant code. It supports a wide range of databases and can be extended with numerous third-party packages, making it a versatile choice for building web applications of all sizes. The core components of Django include models, views, and templates, collectively known as the MTV (Model-Template-View) architecture. Models define the structure of data in the application, mapping to database tables, and are used for querying and manipulating data. Views handle the logic and control flow, receiving HTTP requests, processing data through models, and returning responses. Templates are responsible for rendering the user interface, separating the presentation layer from business logic. Additionally, Django includes middleware for processing requests and responses, a URL dispatcher for routing requests to appropriate views, and a robust admin interface for managing application data. These components work together to create a cohesive and manageable structure for web applications. HTML (HyperText Markup Language) and CSS (Cascading Style Sheets) are fundamental technologies for creating and styling web pages. HTML provides the structure of a web page by using elements like headings, paragraphs, links, images, and forms. It organizes content and allows browsers to interpret and display it correctly. CSS, on the other hand, is used to control the visual presentation of HTML elements. It defines styles for elements, including layout, colors, fonts, and spacing, allowing for a consistent and attractive design. Together, HTML and CSS enable developers to build structured, well-designed web pages that are responsive and visually appealing across various devices and screen

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

sizes. Integrating Django with HTML and CSS involves using Django's templating system to dynamically generate HTML content while applying CSS for styling. Django templates allow developers to insert Python variables and control structures into HTML files, making it possible to create dynamic web pages that adapt based on user input or database content. CSS can be linked to Django templates through static files, which are served by Django's static file handling system. By placing CSS files in the designated static directory and linking them in Django templates, developers can apply consistent styles across all web pages. This integration enables the creation of dynamic and visually cohesive web applications, combining Django's backend capabilities with HTML's structural elements and CSS's styling prowess.

4. RESULT ANALYSIS



Fig 3: Homepage for Sharing Gardening Tips and Resources

The home function in a Sharing Gardening Tips and Resources web application renders the home.html template when a request is made. It takes the request object as a parameter and returns the rendered template. This function serves to display the home page of the web application. Non-authenticated users would only see "Login" and "Register" links. This approach simplifies the menu by treating all logged-in users the same, with differentiating between regular users and staff members. It ensures that all authenticated users have access to the same features, streamlining the user interface.

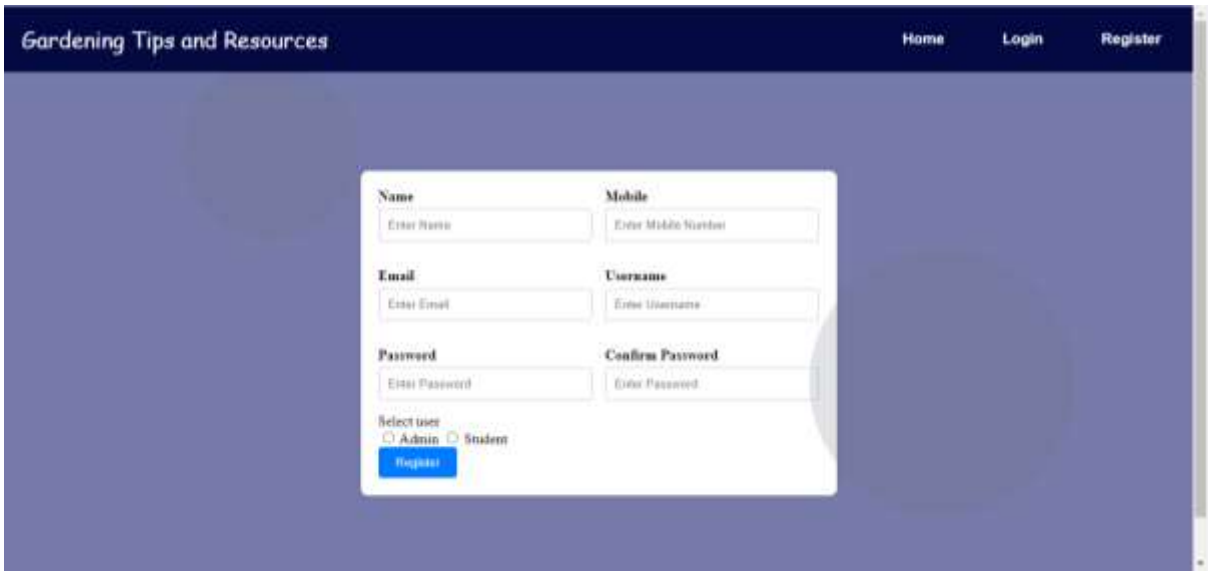


Fig 4: Registration Page for Sharing Gardening Tips and Resources

The register function handles user registration in a Sharing Gardening Tips and Resources web application. When a POST request is made, it retrieves user details from the form, including name, email, username, password, confirmation password, and user type (admin or regular). It checks if the

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

passwords match and whether the username already exists. If the username is unique and passwords match, a new user is created with the provided details, including setting the user as staff if selected. On success, it redirects to the login page with a success message. If there are errors, appropriate error messages are displayed, and the user is redirected back to the registration page. For GET requests, it renders the registration form



Fig 5: Login for Admin and User

The login function handles user authentication in a Sharing Gardening Tips and Resources web application. It processes POST requests by retrieving the username and password, authenticates the user, and logs them in if the credentials are correct. On successful login, it redirects to the home page and shows a success message. If authentication fails, it redirects back to the login page with an error message. For GET requests, it renders the login page.



Fig 6: Homepage After Login

The navigation menu for Sharing Gardening Tips and Resources would display the same options for all authenticated users. Logged-in users would see links to "Add Food," "Food list," and "Logout," regardless of their role or privileges. Non-authenticated users would only see "Login" and "Register" links. This approach simplifies the menu by treating all logged-in users the same, with differentiating between regular users and staff members. It ensures that all authenticated users have access to the same features, streamlining the user interface.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024



Fig 7: Admin Dashboard for checking Request Tips

The Dashboard page function retrieves all records from the NeedTips model, storing them in the allneeds variable. It then renders the Dashboard page template, passing the retrieved records to the template under the context variable allneeds. This setup allows users to view a list of all garden-related needs.



Fig 8: Adding Tips to Requests

The Dadhboard page function manages the submission of gardening tips via a form. If the request method is POST, it initializes a FormTips instance with the submitted data. If the form is valid, it saves the form data and redirects the user to the dashboard page. If the request method is not POST or if the form is invalid, it initializes an empty FormTips form. Finally, it renders the Dashboard page template, passing the form instance to the template under the context variable form. This setup allows users to submit gardening tips and see the form on the page.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024



Fig 9: User Dashboard For Requesting Tips

The Dashboard Page function handles the addition of new garden Tip Request. When a POST request is received, it extracts garden-related details such as garden type, experience level, climate region, specific plan, and available space from the form data. It then creates a new Request object with these details, associating it with the current user, and saves it to the database. After saving, it redirects the user to the dashboard page. This function ensures that users can submit and save their garden plans efficiently.



Fig 10: Final tips for gardening and resources

5. CONCLUSION

The development of a SQL-enabled website for sharing gardening tips and resources using the Django framework provides a modern and efficient platform for gardening enthusiasts. By transitioning from traditional methods such as printed publications and local workshops to a web-based system, this approach addresses several key limitations associated with earlier methods. Traditional gardening resources often faced issues with delayed information dissemination, inconsistent content management, and limited community engagement. In contrast, the integrated web-based platform developed in this project ensures real-time updates, centralized content management, and enhanced community interactions. This is achieved through the use of a robust SQL backend, which facilitates

ISSN 2319-5991 www.ijerst.com
Vol. 20, Issue 4, 2024

secure and scalable data management. The use of Django, a high-level Python web framework, simplifies the development process and provides built-in features for user authentication, form handling, and administrative functionalities. By leveraging Django's ORM (Object-Relational Mapping) capabilities, the application efficiently manages user-generated content, such as gardening tips and community interactions, while ensuring data integrity and security. One of the key benefits of this system is its ability to offer personalized experiences to users. Through tailored content and collaborative features, users can access gardening tips, contribute their own advice, and engage in community discussions. This enhances the overall user experience and fosters a vibrant online gardening community. Additionally, the SQL backend supports efficient data retrieval and storage, which is crucial for handling the diverse and dynamic nature of gardening information. This enables the platform to handle a growing volume of user contributions and interactions without compromising performance. Exploring integration with e-commerce features, such as an online store for gardening supplies or affiliate links to recommended products, could offer additional value to users and generate revenue for the platform. As the user base grows, scaling the infrastructure to ensure performance and reliability will be crucial.

REFERENCES

[1]. Nishath Shireen and Banita Devi. Smart garden management system. 2018.
[2]. Norakmar binti Arbain Sulaiman and Muhamad Dan Darrawi bin Sadli. An iot-based smart garden with weather station system. In 2019 IEEE 9th Symposium on Computer Applications Industrial Electronics.
[3]. J. Kumar, N. Gupta, A. Kumari And S. Kumari, "Automatic Plant Watering And Monitoring System Using Nodemcu," 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2019.
[4]. G Kapinesh, Gayatri Konduru, V Thanikaiselvan, "A New Approach for Effective Garden Monitoring using IoT Applications", 2022 7th International Conference on Communication and Electronics Systems.
[5]. A. Sulaiman, M. Dan, S. Alam and A. S. Garden, "System", 2019 IEEE 9th Symp. Comput. Appl. Ind. Electron, pp. 38-43, 2019.
[6]. M. Sheth and P. Rupani, "Smart Gardening Automation using IoT with BLYNK App", Proc. Int. Conf. Trends Electron. Informatics ICOEI 2019, vol. 2019-April.
[7]. S. N. Ishak, N. N. N. Abd Malik, N. M. Abdul Latiff, N. Effiyana Ghazali and M. A. Baharudin, "Smart home garden irrigation system using Raspberry Pi", 2017 IEEE 13th Malaysia Int. Conf. Commun. MICC 2017.
[8]. S. Devika, S. Khamuruddeen, S. Khamurunnisa, J. Thota and K. Shaik, "Arduino based automatic plant watering system", International Journal of Advanced Research in Computer Science and Software Engineering.
[9]. S Sajithra Varun, Sudhanwa M Rao, Meghana P, Sanket S Desai, Tilak Kumar S H, "An IOT Based Smart Monitoring and Controlling System for Garden", 2023 7th International Conference on Design Innovation for 3 Cs Compute Communicate Control .
[10]. Mahfuzulhoq Chowdhury, Alve Ahmad, "Gardener: A Gardening Assistance Mobile Application With Plant Details, Plant Shopping, Expert Contact, Alert, and Plantation Tips Facilities", 2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions.