



Emotion Web: An Open-Source Python Library for Automated Generation of Personalized Event Websites

Aniket Vishwakarma¹, Suhani Priya², Zeenat Praween³, Isha Kumari⁴, Manoraj Babu⁵,
Md. Irfan Alam⁶

^{1,2,3,4,5} Students, Faculty of CSE & IT, Jharkhand Rai University, Ranchi, Jharkhand, India.

⁶ Faculty of CSE & IT, Jharkhand Rai University, Ranchi, Jharkhand, India.

To Cite this Article: Aniket Vishwakarma¹, Suhani Priya², Zeenat Praween³, Isha Kumari⁴, Manoraj Babu⁵, Md. Irfan Alam⁶, “Emotion Web: An Open-Source Python Library for Automated Generation of Personalized Event Websites”, Indian Journal of Computer Science and Technology, Volume 05, Issue 03 (September-December 2026), PP: 30-37.



Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: When designing customized websites for various events like birthdays, anniversaries, proposals, and celebrations, developers frequently spend a lot of time writing comparable code. Development time is increased and productivity is decreased by this iterative procedure. In order to overcome this difficulty, this work introduces EmotionWeb, an open-source Python toolkit that uses JSON -based setup and reusable templates to automatically create customized event websites. EmotionWeb eliminates tedious coding tasks and creates adaptable, configurable websites by simply requiring the most basic user data. The suggested library enhances code reusability, streamlines website creation, and provides developers with an effective tool to create customized websites faster and more efficiently.

Key Word: Python, Open-Source Library, Website Generation, Website Automation, Personalized Websites, Template-Based Generation, JSON [10] Configuration, Code Reusability, Developer Productivity.

I. INTRODUCTION

1.1 Background

The World Wide Web has developed into a platform for delivering interactive, visually rich, and customized digital experiences from a medium mostly utilized for conveying static textual information. Text, photos, navigation elements, animations, multimedia content, forms, and interactive features are just a few of the many aspects found on contemporary websites. The significance of web-based platforms for communication, information exchange, company promotion, and personal expression has grown dramatically as a result of this growth [1]. However, it frequently takes a great deal of technical expertise, development work, and iterative design to create a website that is both esthetically pleasing and customized to the unique needs of individual users.

Requirement analysis, interface design, implementation, testing, deployment, and future maintenance are some of the processes that are often included in web development. Even while some of these tasks have been made easier by frameworks, content management systems, templates, and visual development tools, creating a highly customized website can still require a significant amount of manual labor. Although predefined templates can speed up creation, they frequently limit modification and might not accurately reflect the individual needs, interests, and identity of a given person or event [1]. These restrictions become especially important for websites that are event-based, since depending on the type of event, users may need specific layouts, themes, photos, timetables, galleries, invitations, and other content.

Researchers and developers have been inspired to investigate methods that can lower the complexity and expense of traditional web development because to the growing need for automated website creation. With less human interaction, automatic website generation seeks to convert user requirements, design specifications, samples, or other types of information into usable online interfaces [1]. Mock-up-driven generation, example-based generation, and artificial intelligence-driven website generation are examples of current methods. These methods show that automation can lower development effort while allowing users to create usable websites without having to manually integrate each component [1]. However, a lot of current solutions concentrate on creating general-purpose websites and could not particularly address the needs of customized event websites or offer a reusable, developer-focused foundation for their creation.

A software systems architecture and structure, in addition to automation, have a significant impact on its adaptability, maintainability, and capacity for change. The system must be separated into suitable components with clearly defined roles and interactions in order to have good software architecture [2].

Instead than adopting technologies or architectural approaches just because they are fashionable, architectural decisions should be informed by the functional and behavioral requirements of the system. A well-structured architecture facilitates the development of software that can adjust to changing requirements and permits individual components to change with little effect on the rest of the system [2].

These guidelines are especially crucial for an automated website generating library since various website elements, themes, layouts, content, and customization choices must cooperate while still being manageable on their own.

Another crucial component of creating adaptable and maintainable systems is reusable software design. When a system needs to accommodate several requirements and future modifications, object-oriented software architecture becomes more challenging. Therefore, rather than fixing every issue from scratch, skilled software designers employ tried-and-true solutions to recurrent design issues. These recurrent solutions, often referred to as design patterns, offer reusable methods for class and object organization and can improve the flexibility, comprehensibility, and reusability of software systems [3]. Applying such principles to an automated website generation system can help separate individual website components and allow new layouts, themes, and features to be incorporated without extensively modifying the existing system.

A particularly good application domain for fusing automation, customization, modular architecture, and reusable design is event websites. Weddings, birthdays, conferences, cultural events, academic gatherings, festivals, and other social or professional occasions sometimes call for websites with similar basic components but very different visuals and content.

For instance, two wedding websites might have different themes, colors, font, layouts, and content, but they might still need event details, photos, venue details, timetables, and contact information. Manually creating each of these websites is tedious and time-consuming, and using just pre-made templates restricts the opportunity to customize the final product.

This study suggests **Emotion Web**, an open-source Python module for the automatic creation of customized event websites, as a solution to these problems. With the help of the suggested library, event-specific websites may be created with less manual development work thanks to its reusable and configurable components. **Emotion Web** seeks to offer a structured mechanism for defining event information, choosing or customizing visual qualities, and creating a working website in accordance with the defined requirements, rather than requiring developers or users to build each online component independently. The library can be incorporated into Python-based development environments and possibly expanded by developers in accordance with their application requirements thanks to the use of Python.

Automation, customization, modularity, reusability, and extensibility are highlighted in the suggested method. Emotion Web seeks to minimize repetitive implementation work while preserving event-specific features in created websites by structuring website-generation software into reusable components. The library's open-source status is also meant to promote community-driven development, expansion, and modification. In this way, the project combines the motivation behind automatic website generation [1], the principles of modular and requirement-driven software architecture [2], and the benefits of reusable software design patterns [3].

Thus, designing and developing Emotion Web as a reusable Python-based framework for automatically creating customized event websites is the main goal of this project.

The study looks into how automated generation methods and reusable software components might be used to streamline website development while preserving customization and flexibility. The final solution is meant to show that automated generation does not always mean losing personalization; rather, a well-thought-out, modular generation framework can offer both event-specific customization and development speed.

II.LITERATURE REVIEW

The demand for website development has increased significantly in the last 10 years, and many website builders, content management systems, and web design frameworks have been developed. Popular platforms like WordPress, Wix and Squarespace [4] enable users to create websites using graphical interfaces with pre-defined templates. Although these platforms make it easier to create websites, they are mostly intended for end users and frequently provide developers who need more automation and customization less freedom.

Because they provide strong backend capabilities and organized project management, Python-based web frameworks like Django and Flask [5] have gained popularity as options for creating dynamic web applications. For each new website, developers must still manually generate HTML templates, CSS styling, JavaScript functionality, and project resources. Repetitive coding is therefore still a frequent problem while developing customized websites.

To increase code reusability, a number of template engines and static website generators [11] have also been established. By isolating presentation from business logic and enabling reusable templates, these solutions minimize development work. However, rather than creating customized event websites, the majority of current solutions concentrate on creating general websites. They don't offer a specialized method for automatically creating entire websites using user data specific to an event, like names, messages, graphics, and themes. In order to increase developer productivity, modern software engineering techniques place a strong emphasis on automation, modularity, and code reusability [2-4]. Despite these developments, there aren't many lightweight Python libraries that are explicitly made to automate the creation of customized event websites while combining template management, JSON [10]-based configuration, and automated asset handling into a single reusable package.

EmotionWeb fills this research vacuum by providing an open-source Python toolkit [5,11] that uses configuration-driven generation and reusable templates to automate tedious website construction procedures. EmotionWeb focuses on minimizing repetitious writing for developers while preserving flexibility, scalability, and simplicity of integration with Python applications, in contrast to conventional website builders or general-purpose frameworks.

III.PROBLEM STATEMENT

Although the creation of applications has been made easier by modern web development frameworks, developers still have to deal with the problem of writing the same code over and over again while building customized websites for various events like birthdays, anniversaries, proposals, celebrations, and invitations. The general project structure, templates, styling, animations, and resource management don't really change, even though the content of the website does. Coding effort is increased, important development time is wasted, productivity is decreased, and project maintenance is made more challenging by this repeating development process.

Current content management systems and website builders are mainly made for end users rather than software developers. These platforms are difficult to incorporate into Python-based development workflows and offer little automation options. Similarly, web frameworks such as Django and Flask [6] provide powerful backend capabilities but do not eliminate the repetitive front-end development required for multiple personalized websites.

A reusable and automated solution is therefore required so that developers may effectively create customized websites without having to rewrite the same code for each job. The proposed EmotionWeb library addresses this problem by automating website generation through reusable templates, JSON [10]-based configuration, and automatic resource management, thereby reducing repetitive coding and improving developer productivity.

IV.OBJECTIVES

The primary objective of this research is to design and develop EmotionWeb, an open-source Python library [5,11] that automates the generation of personalized event websites while reducing repetitive coding and improving developer productivity.

The specific objectives of this research are as follows:

4.1 To Automate Website Generation

Create a Python library that can leverage pre-made templates and user-supplied data to automatically create customized event websites.

4.2 To Reduce Repetitive Coding

Reduce the amount of time developers must spend creating identical HTML, CSS, JavaScript, and project structures for each new customized website.

4.3 To Improve Developer Productivity

By automating routine development operations, you may shorten the time it takes to construct a website and free up developers to concentrate on feature development and customization.

4.4 To Provide a Reusable Template-Based Architecture

Create a modular framework that allows for the efficient creation of websites for various event categories by supporting reusable templates.

4.5 To Implement JSON [10]-Based Configuration

Use JSON [10] files to manage website content dynamically, making customization simple, flexible, and easy to maintain.

4.6 To Manage Website Resources Automatically

During the creation of a website, automate the handling of static resources, templates, and images.

4.7 To Develop an Open-Source Python Library

Make a Python package that is expandable and reusable so that developers may include it into many projects and keep improving it.

4.8 To Support Future Scalability

Build the library modularly to allow for the addition of new themes, templates, and advanced features such as cloud deployment and content created by AI in future versions.

V.PROPOSED SYSTEM

5.1 System Overview

Emotion Web, the suggested system, is an open-source Python package [5,11] created to automate the creation of customized event websites. Developers merely need to supply the necessary event information rather than manually constructing HTML pages, CSS stylesheets, JavaScript files, and project folders for each new event. After processing this data, the library maintains website resources, applies pre-made themes, and instantly creates a fully functional responsive website.

Each module of Emotion Web's modular architecture carries out a distinct function. This modular approach greatly minimizes repetitious development labor while increasing maintainability, scalability, and code reusability.

5.2 Input Module

All of the data needed to create a website is gathered by the input module. Event-specific information, including recipient name, event title, customized message, photos, theme choice, and further customization choices, are supplied by the developer. Prior to additional processing, these inputs are verified.

The module is made to reduce the amount of manual configuration and make the process of creating websites easier.

5.3 JSON [6] Configuration Module

EmotionWeb saves all of the data it receives from the user in a JSON [10] configuration file. Website content and presentation are kept apart by the JSON [10] file, which serves as a centralized data source.

Because developers can alter website content without altering the source code, this method increases flexibility. Additionally, it makes future maintenance and template processing easier.

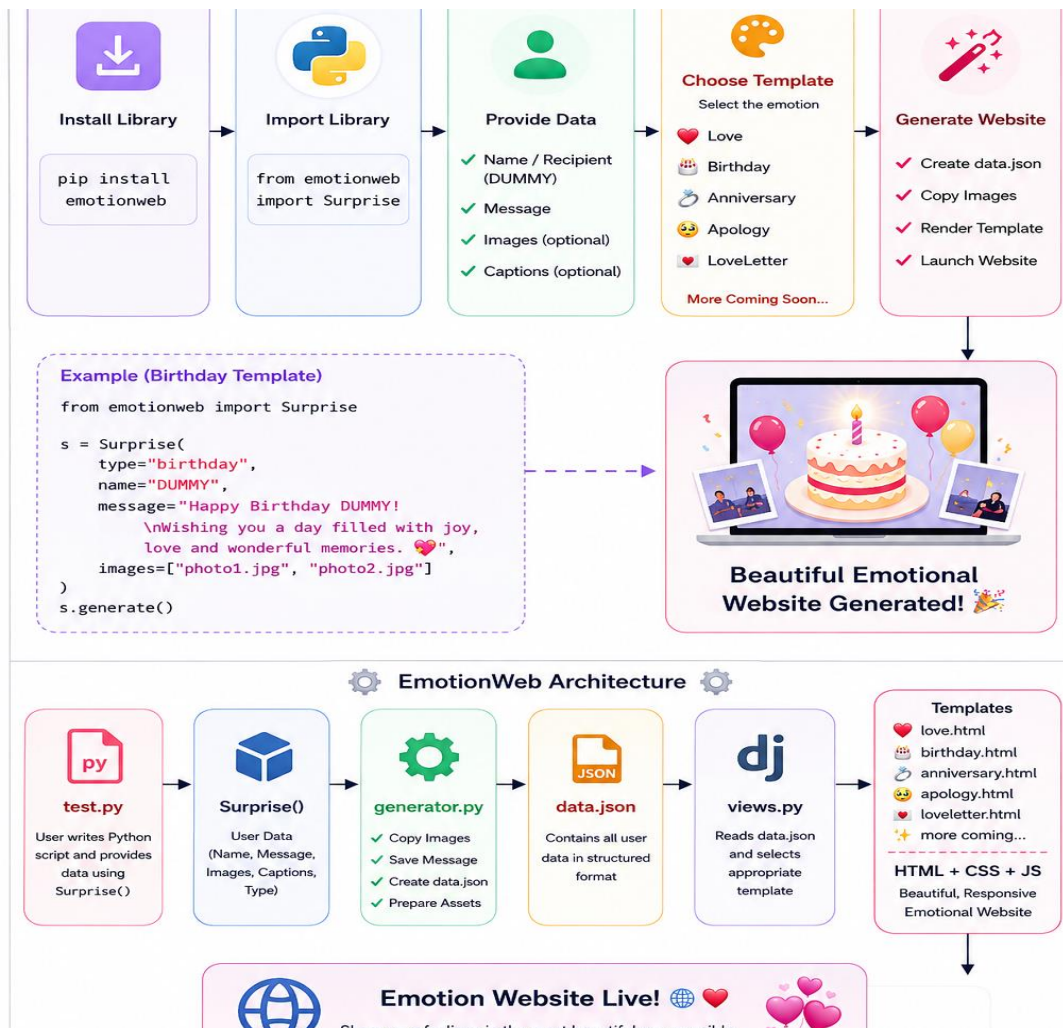


Fig 1. Overall System Architecture

5.4 Template Engine

One of the fundamental elements of Emotion Web is the Template Engine. It replaces specified placeholders with the relevant user data provided in the JSON [10] configuration and chooses the relevant website template based on the event type.

This makes it possible for developers to reuse the same template for several customized websites and removes the need for repetitive HTML tweaking.

5.5 Asset Management Module

Images, CSS files, JavaScript files, fonts, and other static assets are automatically copied and arranged by the Asset Management Module into the project directory that is created.

Emotion Web minimizes human file handling and preserves a consistent project structure by automating resource management.

5.6 Website Generation Module

Emotion Web creates a responsive and fully functional website after processing templates and assets. All of the files needed for local execution or deployment are included in the created project. The resulting website can be previewed right away by developers without the need for further configuration.

While drastically cutting down on development time, the created website maintains responsiveness, personalization, and template consistency.

5.7 Working Flow of EmotionWeb

The overall working process of EmotionWeb can be summarized as follows:

- Developer provides event information.
- User data is stored in a JSON [10] configuration file.
- The Template Engine loads the selected template.
- Placeholders are replaced using JSON [10] data.
- Images and static resources are copied automatically.
- The final responsive website is generated.
- The website is ready for preview or deployment.
- This workflow eliminates repetitive coding and provides a fast, reusable, and developer-friendly

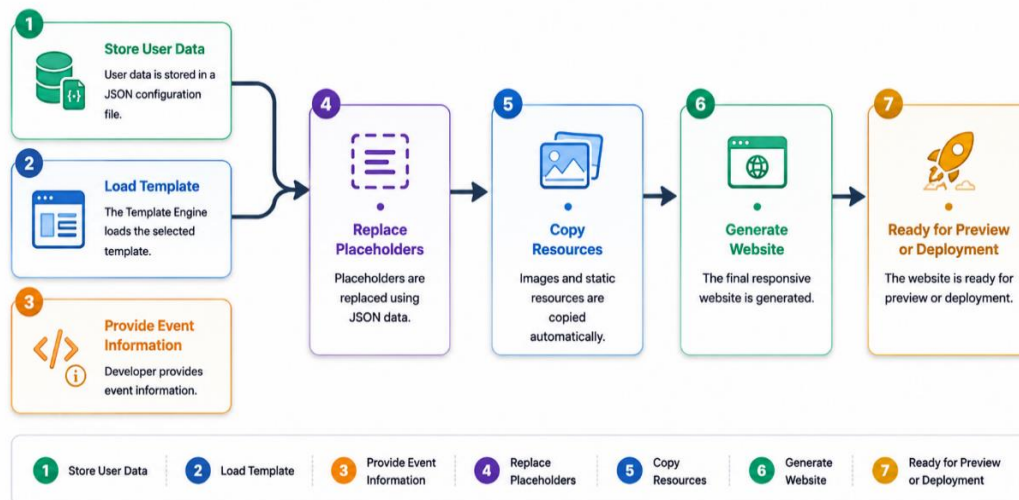


Fig 2.Solution for personalized website generation

VI.METHODOLOGY

6.1 Data Collection and Input Processing

Gathering the data needed to create a customized website is the first step in the suggested process. EmotionWeb only needs the recipient's name, event title, personalized message, images, theme preference, and other customization details, in contrast to traditional website development, where developers manually edit numerous HTML, CSS, and JavaScript [7–11] files. To guaranty accuracy and consistency, these inputs are verified prior to processing.

The library creates a structured JSON [11] configuration file after gathering the necessary data. Website content and presentation logic are separated by the JSON [11] file, which serves as an intermediary data source. This method makes the entire creation process more flexible and maintainable by enabling developers to alter website information without changing the source code. EmotionWeb also increases scalability and makes future improvements easier by storing user data in an organized fashion.

6.2 Automated Template Processing and Resource Management

After processing the input data, EmotionWeb automatically selects the appropriate website template based on the requested event category. Each template contains predefined placeholders representing dynamic content such as names, messages, images, dates, and other personalized information. The library replaces these placeholders using values stored in the JSON [11, 13, 17] configuration file, thereby eliminating the need for repetitive manual editing of HTML pages.

In addition to template processing, EmotionWeb automatically manages website resources such as images, CSS files, JavaScript files, fonts, and other static assets. These resources are copied into the generated project directory while preserving a well-organized folder structure. This automation ensures consistency across different websites and significantly reduces manual file management. The combination of reusable templates and automatic resource handling enables developers to generate multiple personalized websites with minimal effort while maintaining code quality and project organization.

6.3 Website Generation and Output Validation

Emotion Web creates a fully functional responsive website in the last phase by combining the controlled resources and processed templates. Developers can preview the website right away without doing any additional settings because the created project includes all the files needed for execution or deployment. The library makes sure that all resources are successfully linked, graphics are loaded, and tailored content is displayed correctly.

The system performs output validation by examining template rendering, content placement, responsiveness, and general operation to confirm the accuracy of the created website. This validation procedure guaranties that the resulting website maintains usability and design consistency while accurately reflecting the user information supplied. EmotionWeb greatly reduces repetitious coding, boosts developer productivity, and offers a scalable solution for customized website development by automating the entire workflow from data collection to final website generation.

VII.IMPLEMENTATION

7.1 Development Environment

Python was the main programming language used in the development of EmotionWeb [5] because to its ease of use, versatility, and wide library support. Git was utilized for version control, and Visual Studio Code [16] served as the development environment. JSON [11] was employed for organized setup and data storage, while HTML, CSS, and JavaScript [7–10] were utilized to develop responsive website designs. Python modules, reusable templates, static resources, and configuration files are all part of the project's modular framework.

7.2 Core Library Implementation

The generator.py module, which controls configuration processing, template selection, resource handling, and website production, is where Emotion Web's fundamental functionality is implemented. A structured JSON [10] configuration file contains user inputs including recipient name, message, theme, event category, and graphics.

Following the loading of the chosen HTML template, values from the JSON [10] data are automatically substituted for preset placeholders. Additionally, the system copies static resources into the produced project directory, including fonts, pictures, CSS, and JavaScript. The finished website is now ready for deployment or preview. 7.3 System Modules and Functional Workflow

Emotion Web consists of the following major modules:

Input Module: Collects event details, messages, images, and theme preferences.

Configuration Module: Stores user information in a JSON [10] configuration file.

Template Processing Module: Loads templates and replaces predefined placeholders with dynamic data.

Asset Management Module: Copies images and other static resources while maintaining the required directory structure.

Website Generation Module: Combines templates, configuration data, and resources to generate the final responsive website.

This modular design reduces repetitive development work and improves maintainability, reusability, and scalability.

7.4 Technology Stack

Emotion Web employs JSON for configuration management, HTML, CSS, and JavaScript for responsive website templates, and Python for backend processing. Git and Visual Studio Code were utilized as version-control and programming tools, respectively. The table below provides an overview of the technologies and libraries utilized in the implementation.

Technology	Purpose
Python [1]	Core programming language
Django [2]	Backend framework
HTML5 [3, 10]	Webpage structure
CSS3 [4, 11]	Styling and responsive design
JavaScript [5]	Interactive webpage functionality
JSON [6]	Dynamic data storage
json (Python Library) [1]	Configuration file handling
os (Python Library) [1]	File and directory management
shutil (Python Library) [1]	Copying static resources
webbrowser (Python Library) [1]	Automatic website preview

Table 1. Technology Stack Used

The technologies used in EmotionWeb were selected based on their reliability, modularity, and suitability for automated website generation. Python provides the core programming environment [5], Django supports backend processing [6], HTML5, CSS3, and JavaScript enable responsive frontend development [7–8, 14, 15], while JSON and Python's standard libraries (json, os, shutil, and webbrowser) facilitate configuration management, file handling, resource management, and automatic website preview [5, 10].

VIII.RESULTS AND DISCUSSION

8.1 Experimental Results

In order to create customized event websites, the suggested EmotionWeb library was successfully deployed and tested. Different event categories were developed during testing by supplying user data, including the recipient's name, a customized message, pictures, and theme choice. The supplied data was automatically analyzed by the library, which then created a JSON [11] configuration file, chose the right template, arranged the necessary static resources, and created a fully functional responsive website.

The created websites constantly updated individualized content based on input while maintaining a consistent layout. Every new website no longer required the manual editing of HTML, CSS, and JavaScript [7–8] files thanks to the automated procedure. EmotionWeb can create reusable and configurable websites with little developer work, as the trial implementation showed.

Feature	Result
User Input Processing	Successfully Implemented
JSON Configuration Generation	Successfully Implemented
Template Processing	Successfully Implemented
Image and Asset Management	Successfully Implemented
Responsive Website Generation	Successfully Implemented
Website Preview	Successfully Implemented

Table 2. Functional Validation

8.2 Performance Analysis and Discussion

The implementation results demonstrate how EmotionWeb streamlines the production of customized websites by automating resource management, project structure generation, and template processing. Its modular design makes it easier to maintain and enables content and templates to be updated separately. Configuration based on JSON [10] offers flexibility without changing the original code. All things considered, EmotionWeb facilitates the consistent creation of several customized websites, decreases repetitious coding, and increases developer productivity.

Parameter	Manual Development	EmotionWeb
Repetitive Coding	High	Very Low
Template Reusability	Limited	High
Development Efficiency	Moderate	High
Content Customization	Manual	JSON-Based
Project Organization	Manual	Automatic
Scalability	Moderate	High

Table 3: Comparison Between Manual Development and Emotion Web

The outcomes of the trial show that Emotion Web effectively accomplishes its main goal of automating the creation of customized websites. By decreasing tedious coding, increasing code reusability, and streamlining website construction while maintaining flexibility and customization, the library offers developers a workable alternative.

IX.FUTURE WORK

9.1 Content Creation Driven by AI

Artificial intelligence can be incorporated into future iterations of Emotion Web to automatically produce customized welcomes, messages, website content, and design recommendations depending on the chosen event.

9.2 Expanded Template Library

A greater selection of templates for a variety of categories, including commercial websites, portfolios, educational websites, wedding invitations, festivals, and professional landing pages, can be added to the library.

9.3 Cloud-Based Website Deployment

Future iterations might enable developers to publish websites immediately by supporting the automatic deployment of generated websites to cloud hosting services. Furthermore, advanced cryptographic techniques can be employed to strengthen cloud security and address evolving security threats and vulnerabilities[9,18].

9.4 Enhanced Theme Customization

To increase design versatility and enhance the user experience, more themes, animations, color schemes, typefaces, and interactive UI elements can be added.

9.5 Plugin and Extension Support

To enable developers to design and incorporate unique templates, themes, and other features without changing the main library, a plugin architecture can be implemented.

9.6 Package Distribution and Community Development

Emotion Web can be released as a fully documented Python package with community contributions and version control. The library can be strengthened and its use among developers increased by ongoing enhancements, bug fixes, and feature additions from open-source contributors.

X.CONCLUSION

This research presented Emotion Web, an open-source Python library [5,12] developed to automate the generation of personalized event websites. The primary objective of the library is to reduce the repetitive coding involved in creating similar websites for different occasions while improving developer productivity and code reusability. By utilizing reusable templates, JSON [10]-based configuration, and automated resource management, Emotion Web simplifies the website development process and minimizes manual effort.

Emotion Web, an open-source Python toolkit [5,12] created to automate the creation of customized event websites, was presented in this study. The library's main goal is to increase developer productivity and code reusability while reducing the amount of repetitious coding required to create identical websites for various situations. Emotion Web reduces manual labor and streamlines the website building process through the use of reusable templates, JSON [11]-based setup, and automated resource management.

All things considered, Emotion Web is a useful tool for developers that helps close the gap between automated and human website creation. In addition to saving development time, the suggested approach lays a solid basis for upcoming improvements including cloud deployment, enhanced template management, AI-powered content creation, and community-driven open-source development.

REFERENCES

1. Kaluarachchi, T., & Wickramasinghe, M. (2023). A systematic literature review on automatic website generation. *Journal of Computer Languages*, 75, 101202. <https://doi.org/10.1016/j.cola.2023.101202>
2. Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures* [Doctoral dissertation, University of California, Irvine].
3. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design patterns: Elements of reusable object-oriented software*. Addison-Wesley.
4. Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson.
5. Python Software Foundation. (n.d.). *Python 3 documentation*. Retrieved July 31, 2026, from <https://docs.python.org/3/>
6. Django Software Foundation. (n.d.). *Django documentation*. Retrieved July 29, 2026, from <https://docs.djangoproject.com/>
7. Mozilla Developer Network. (n.d.). *HTML: HyperText Markup Language*. Retrieved August 2, 2026, from <https://developer.mozilla.org/docs/Web/HTML>
8. Mozilla Developer Network. (n.d.). *CSS: Cascading Style Sheets*. Retrieved August 4, 2026, from <https://developer.mozilla.org/docs/Web/CSS>
9. Alam, I., & Singh, S. N. (2015). Mitigating cloud security challenges with intrusion detection technique. *Splint International Journal of Professionals*, 2(6), 7–15.
10. Mozilla Developer Network. (n.d.). *JavaScript guide*. Retrieved May 12, 2026, from <https://developer.mozilla.org/docs/Web/JavaScript/Guide>
11. JSON.org. (n.d.). *Introducing JSON*. Retrieved May 10, 2026, from <https://www.json.org/>
12. Python Packaging Authority. (n.d.). *Python packaging user guide*. Retrieved May 4, 2026, from <https://packaging.python.org/>
13. Python Package Index. (n.d.). *Python Package Index (PyPI)*. Retrieved July 6, 2026, from <https://pypi.org/>
14. WHATWG. (n.d.). *HTML: The living standard*. Retrieved August 6, 2026, from <https://html.spec.whatwg.org/>
15. World Wide Web Consortium. (n.d.). *CSS specifications*. Retrieved July 5, 2026, from <https://www.w3.org/Style/CSS/>
16. Microsoft. (n.d.). *Visual Studio Code documentation*. Retrieved July 3, 2026, from <https://code.visualstudio.com/docs>
17. Vishwakarma, A. (2026). *EmotionWeb: An open-source Python library for automated generation of personalized event websites* [Computer software]. PyPI. <https://pypi.org/project/emotionweb/>
18. Alam, M. I. (2020). Enhancing cloud security using multi-level DNA cryptography. *Splint International Journal of Professionals*, 7(1), 75–82.