iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** II **Month of publication:** February 2026

DOI: https://doi.org/10.22214/ijraset.2026.77616

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue II Feb 2026- Available at www.ijraset.com

MAKEFOLIO - A Web-Based Portfolio

V.S.S.P.L.N. Balaji Lanka[1], Chindam Vaishnavi[2], Thanniru Venkatasai[3], Vankudoth Bhaskar[4], Thotla Shainee[5]
[1]*Assistant Professor, Department of CSE, Vignan Institute of Technology and Science, Hyderabad*
[2, 3, 4, 5]*Student, Department of CSE, Vignan Institute of Technology and Science, Hyderabad*

Abstract: ***In the present digital era, online portfolios have become an important medium for showcasing skills, academic achievements, and professional experience. However, creating a portfolio website often requires technical knowledge and design skills, which many students and professionals do not possess. To overcome this challenge, the MakeFolio project proposes a simple web-based portfolio management system that allows users to create, customize, and download professional portfolios with ease. The system provides secure user authentication, dynamic portfolio generation, customization options, and PDF download functionality, making it an efficient and user-friendly solution for digital portfolio creation.***

Keywords: Portfolio Management System, Web Application, User Authentication, PHP, MySQL, Responsive Design, PDF Generation, Digital Portfolio.

I. INTRODUCTION

In recent years, the rapid growth of the internet and digital technologies has significantly transformed the way individuals present their professional identity. Employers, academic institutions, and clients increasingly prefer digital portfolios over traditional resumes, as they provide a comprehensive and interactive representation of an individual's skills, projects, and achievements. A digital portfolio not only highlights qualifications but also demonstrates creativity, technical competence, and professional maturity.
Despite the growing importance of online portfolios, many students and professionals face difficulties in creating them. Developing a personal portfolio website typically requires knowledge of web technologies such as HTML, CSS, JavaScript, and backend programming. Additionally, users must invest time in designing layouts, structuring content, and ensuring responsiveness across devices. These requirements often discourage non-technical users or beginners from building their own portfolio websites

II. RELATED WORK

Several systems and tools have been developed to support digital resumes and online portfolios. Online resume builders allow users to generate resumes using predefined templates, but they mostly focus on static documents rather than interactive portfolios. Website builders and portfolio platforms provide customization options but often require subscriptions and technical understanding. Some academic institutions use internal portfolio systems, but they are limited to institutional use and lack personalization. Static HTML portfolio templates are also available, but they require coding knowledge and manual editing. Compared to these existing solutions, MakeFolio offers a simpler, more accessible approach by combining ease of use, dynamic portfolio generation, customization, and PDF download functionality without requiring technical expertise.

III. EXISTING SYSTEM

Current practices in digital portfolio creation rely on a mix of traditional document-based methods, static websites, and online platforms. Many students and professionals create portfolios using word processors such as Microsoft Word or Google Docs. These portfolios are static, require manual formatting, and lack interactivity. Another common approach is the use of static HTML and CSS templates to build portfolio websites. While these templates offer better visual presentation, they require technical knowledge and manual code updates. Online resume builders and portfolio platforms also exist, allowing users to generate profiles using predefined templates. However, these platforms often focus only on resumes, provide limited customization, and restrict advanced features behind paid subscriptions. Academic portfolio systems used by institutions are mainly designed for internal evaluation and do not support professional branding or public sharing. Overall, existing systems provide partial solutions but fail to meet the needs of non-technical users seeking flexible and efficient portfolio creation.

A. Disadvantages

1) High Technical Dependency: Static website portfolios and templates require coding skills, making them unsuitable for non-technical users.
2) Limited Customization: Resume builders and hosted platforms offer restricted design and personalization options.

3) Time-Consuming Updates: Manual and static portfolios require repeated effort to modify or update content.
4) Lack of Secure User Management: Many systems do not provide proper user authentication or secure data storage.
5) Cost and Accessibility Issues: Advanced features are often available only through paid subscriptions, limiting accessibility for students.

IV. PROPOSED SYSTEM

The proposed system, MakeFolio, is designed using a modular web-based architecture that simplifies portfolio creation while ensuring flexibility, scalability, and ease of maintenance. The system integrates frontend presentation, backend processing, and data management into clearly defined functional blocks.

A. Modular Web Architecture

1) PHP-Based Backend Backbone
- Handles all core web operations including user registration, login, logout, and session management.
- Manages portfolio form submissions, editing actions, and routing with minimal server overhead.
- Ensures secure user authentication using session-based access control.

2) Portfolio Data Processing
- Structured form inputs are used to collect user details such as skills, projects, education, and contact information.
- User-selected profile images and color themes are processed dynamically.
- JavaScript enhances short inputs by generating meaningful descriptive content automatically.

3) Dynamic Portfolio Generation Core
- HTML, CSS, and Bootstrap are used to dynamically render portfolio layouts.
- JavaScript applies real-time customization such as color themes and layout updates.
- Generated portfolios are responsive and compatible across devices.

4) PDF Export and Download Module
- html2canvas captures the generated portfolio layout accurately.
- jsPDF converts the captured content into a downloadable PDF document.
- Users can export professional portfolios for offline sharing and submissions.

5) User Interaction & Editing Module
- Allows users to revisit and edit portfolio content at any time.
- Pre-filled forms ensure easy modification without data loss.
- Updated portfolios are regenerated instantly.

B. Advantages

1) Modular and Scalable Design
- Each system component (authentication, portfolio generation, PDF export) operates independently.
- Backend modules can be extended or modified without affecting the entire system.

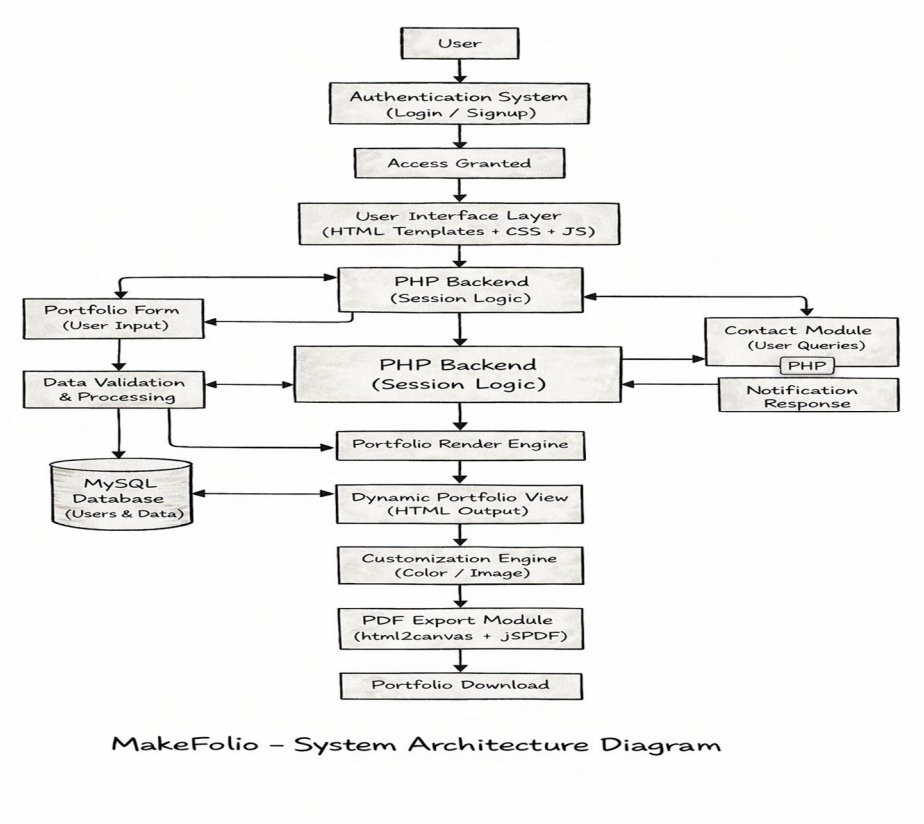
2) User-Friendly and Low Overhead
- No coding knowledge is required to create portfolios.
- Lightweight frontend and backend design ensures fast response times.

3) Customization and Flexibility
- Users can personalize portfolios using color themes and profile images.
- Editable content ensures long-term usability.

4) Cost-Effective and Open Technologies
- Built using open-source tools such as PHP, MySQL, JavaScript, and Bootstrap.
- No dependency on paid APIs or proprietary services.

5) Real-World Ready
- Secure session handling ensures data protection.
- PDF download functionality supports academic and professional use cases.
- Responsive design ensures accessibility on desktops, tablets, and mobile devices.

C. System Architecture Diagram



MakeFolio – System Architecture Diagram

1) User-Friendly Frontend: Users interact with MakeFolio through responsive web pages developed using HTML, CSS, Bootstrap, and JavaScript. The frontend provides login, signup, portfolio creation, editing, and download options. A simple form-based interface allows users to enter their personal and professional details without requiring any technical knowledge.

2) Authentication System: The system includes a secure authentication mechanism for user login and signup. User credentials are verified using a MySQL database, and session-based access control is implemented to ensure that only authenticated users can access portfolio creation and editing features.

3) Core Processing Unit – PHP Backend: The PHP backend acts as the central controller of the MakeFolio system. It manages session handling, processes form submissions, handles user requests, and coordinates communication between the frontend, database, and portfolio generation modules.

4) Portfolio Data Processing: When users submit portfolio details, the system validates and processes the input data to maintain accuracy and consistency. Information such as skills, projects, education, and contact details is structured properly before being stored securely in the database.

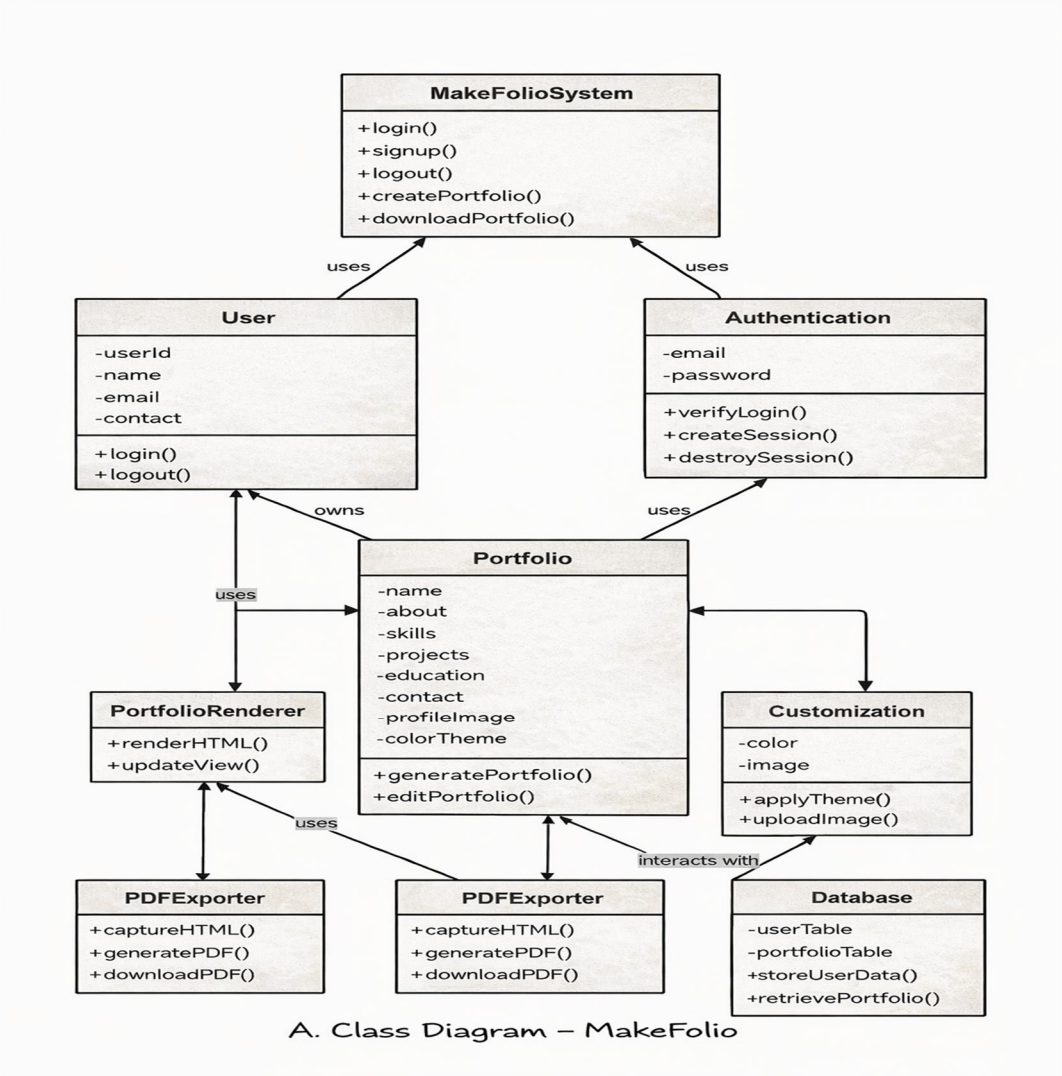
5) Dynamic Portfolio Generation: The portfolio render engine dynamically generates a complete portfolio layout using the stored user data. The generated portfolio is displayed as an HTML output and updates instantly whenever the user edits any information.

6) Customization Engine: MakeFolio allows users to customize their portfolios by selecting color themes and uploading profile images. These customization options ensure that each portfolio is visually unique and professionally presented.

7) PDF Export and Download Module:The generated portfolio is converted into a downloadable PDF using html2canvas and jsPDF. This feature enables users to download and share their portfolios offline for academic or professional purposes.

8) Contact and Notification Module: The contact module allows users to submit queries through a contact form. The backend processes these requests and provides appropriate responses or notifications, completing the overall functionality of the system.

D. Class Diagram



A. Class Diagram – MakeFolio

The class diagram illustrates the overall structure of the MakeFolio Portfolio Management System. It shows how different classes are organized and how they interact with each other to provide a smooth portfolio creation process. The system is modular and scalable, where each class has a clearly defined responsibility and all components are coordinated through a central controller.

1) Core Controller Class: The MakeFolioSystem class acts as the central controller of the application. It manages high-level operations such as user authentication, portfolio creation, and portfolio download. This class interacts with the User, Authentication, Portfolio, and PDFExporter classes. The main methods include login(), signup(), logout(), createPortfolio(), and downloadPortfolio(), which allow users to access and manage the system securely.

2) User Class: The User class represents an individual user of the MakeFolio system. It stores essential details such as userId, name, email, and contact information. The class supports basic operations like login() and logout(). Each user is associated with a single portfolio, establishing ownership of portfolio data.

3) Authentication Class: The Authentication class is responsible for secure access control. It verifies user credentials, creates sessions after successful login, and destroys sessions during logout. The methods verifyLogin(), createSession(), and destroySession() ensure that only authorized users can access the system features.

4) Portfolio Class: The Portfolio class stores all portfolio-related information including name, about, skills, projects, education, contact details, profile image, and color theme. It provides methods such as generatePortfolio() and editPortfolio() to dynamically create and update portfolio content. Each portfolio belongs to a specific user.

5) PortfolioRenderer Class: The PortfolioRenderer class converts portfolio data into a structured HTML format for display. It uses methods like renderHTML() and updateView() to ensure that portfolio content is presented clearly and professionally on the user interface.
6) Customization Class: The Customization class manages personalization features such as color themes and profile image uploads. It includes methods like applyTheme() and uploadImage() to give each portfolio a unique and customized appearance.
7) PDFExporter Class: The PDFExporter class handles the conversion of portfolio content into a downloadable PDF document. It includes methods such as captureHTML(), generatePDF(), and downloadPDF(), enabling users to share their portfolios offline.
8) Database Class: The Database class represents the data storage layer of the system. It manages user and portfolio data using methods like storeUserData(), retrievePortfolio(), and updatePortfolio(). This ensures data persistence and secure storage.
9) Interaction Between Classes: The MakeFolioSystem class coordinates all other classes in the system. The User class owns the Portfolio class, Authentication ensures secure access, PortfolioRenderer and Customization handle display and personalization, PDFExporter generates downloadable content, and all classes interact with the Database for data storage and retrieval

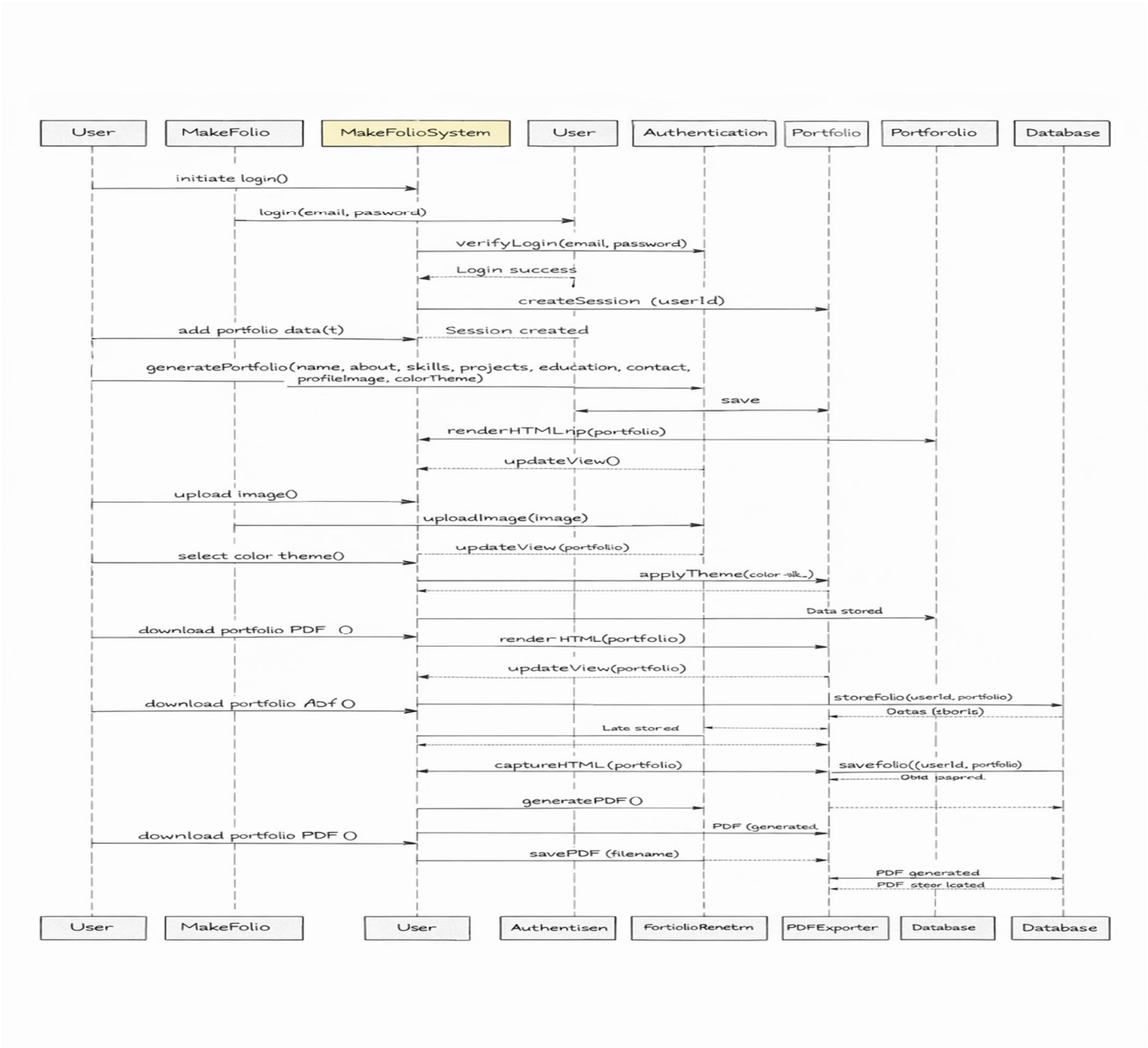


Fig.3. Sequence Diagram

The system sequence diagram illustrates the step-by-step interaction between the user and the core modules of the MakeFolio Portfolio Management System. It shows how requests flow from the user interface to the backend services and database, resulting in dynamic portfolio generation and download. The system follows a modular architecture where each component performs a specific role to ensure smooth and secure operation.

1) Authentication Flow: The sequence begins when the user initiates the login or signup process through the MakeFolio interface. User credentials are submitted via the frontend and forwarded to the backend authentication module. The authentication system validates the credentials against stored records in the database. Upon successful verification, a user session is created, granting secure access to portfolio-related features.
2) Portfolio Data Entry Flow: After authentication, the user enters portfolio details such as personal information, skills, projects, education, and contact details. These inputs are sent to the backend, where data validation and processing take place. The validated information is then stored securely in the database for future retrieval and updates.
3) Portfolio Generation Flow: Once the user submits the portfolio data, the system triggers the portfolio generation module. The backend processes the stored data and passes it to the portfolio rendering engine. The engine dynamically generates an HTML-based portfolio view, which is displayed instantly to the user through the frontend interface.
4) Customization Flow: The customization flow allows users to personalize their portfolios by uploading a profile image and selecting a color theme. The backend processes these customization inputs and applies them to the generated portfolio. The updated portfolio view is rendered again to reflect the selected design changes.
5) PDF Download Flow: When the user requests to download the portfolio, the system captures the rendered HTML content of the portfolio. The PDF export module converts this content into a downloadable PDF document. The generated PDF is saved and made available to the user for download.
6) Data Storage and Update Flow: Throughout the process, all portfolio data, customization settings, and generated outputs are synchronized with the database. This ensures that users can revisit, edit, and regenerate their portfolios at any time without data loss.
7) Session Management Flow: The session management module controls user login and logout activities. When the user logs out, the active session is terminated, and access is revoked to protect user data. The system then redirects the user back to the login interface, ensuring secure session handling.

A. *Summary of Methodology*

The MakeFolio project is developed using a full-stack web development approach. The frontend is designed using HTML, CSS, and JavaScript, with responsive styling supported through Bootstrap. PHP is used as the server-side scripting language to handle backend logic, session management, and data processing. MySQL is employed as the database for storing user credentials, portfolio details, and customization data securely. The system follows a modular structure to ensure clarity, scalability, and ease of maintenance. User authentication is implemented using session-based login, where user credentials are validated against the database and secure sessions are created after successful login. This ensures that only authorized users can create, edit, and download portfolios.

The project is organized as follows:

PHP files handle authentication, portfolio creation, customization, and PDF generation logic.

The database stores user profiles, portfolio data, and customization preferences.

The templates directory contains HTML files for login, registration, portfolio form, preview, and download pages.

CSS and JavaScript files support styling and interactivity.

The application is hosted on a local server environment using XAMPP, which provides Apache and MySQL services. Users access the system through a browser without requiring any technical setup.

1) *Portfolio Creation and Processing*

After successful login, users enter personal and professional details such as name, skills, education, projects, and contact information through a structured form interface. The backend validates this data and stores it in the database. A dynamic rendering engine processes the stored information and generates a real-time portfolio preview in HTML format.

2) Customization and PDF Generation

The system allows users to customize their portfolios by uploading a profile image and selecting color themes. These preferences are applied dynamically to the portfolio layout. When the user selects the download option, the system captures the rendered HTML portfolio and converts it into a PDF document using JavaScript-based PDF generation libraries. The generated PDF is then made available for download.

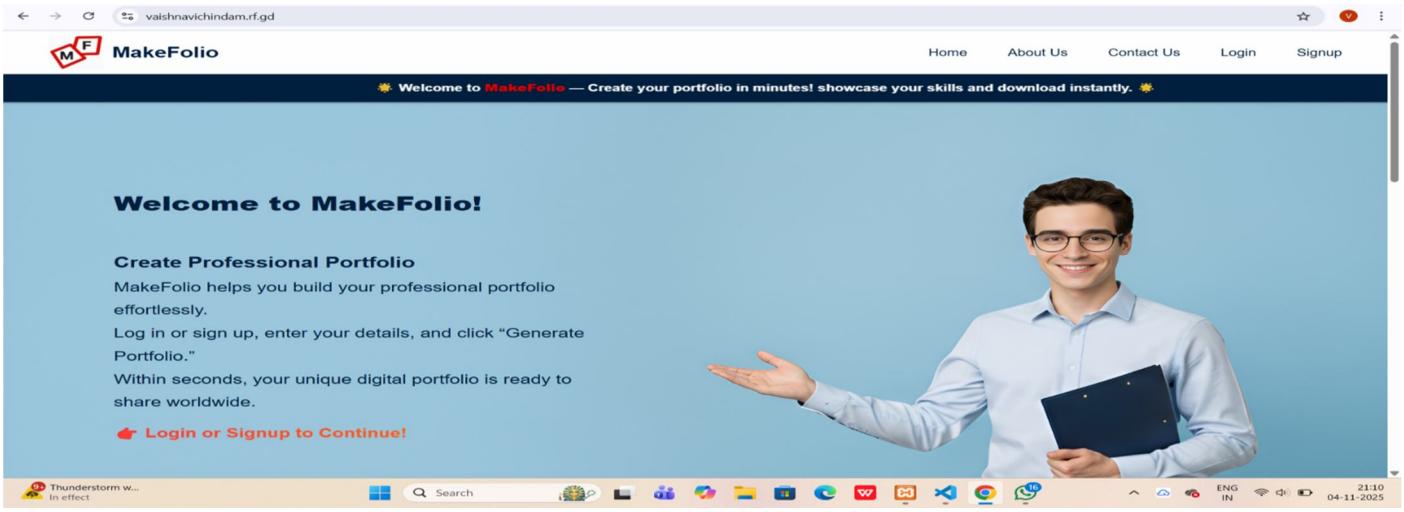
3) Frontend and Usability

The user interface is designed to be simple and intuitive, making the platform accessible to users with no technical background. All interactions—including login, data entry, customization, preview, and download—are seamlessly connected through backend routes, ensuring a smooth user experience.

V. RESULTS

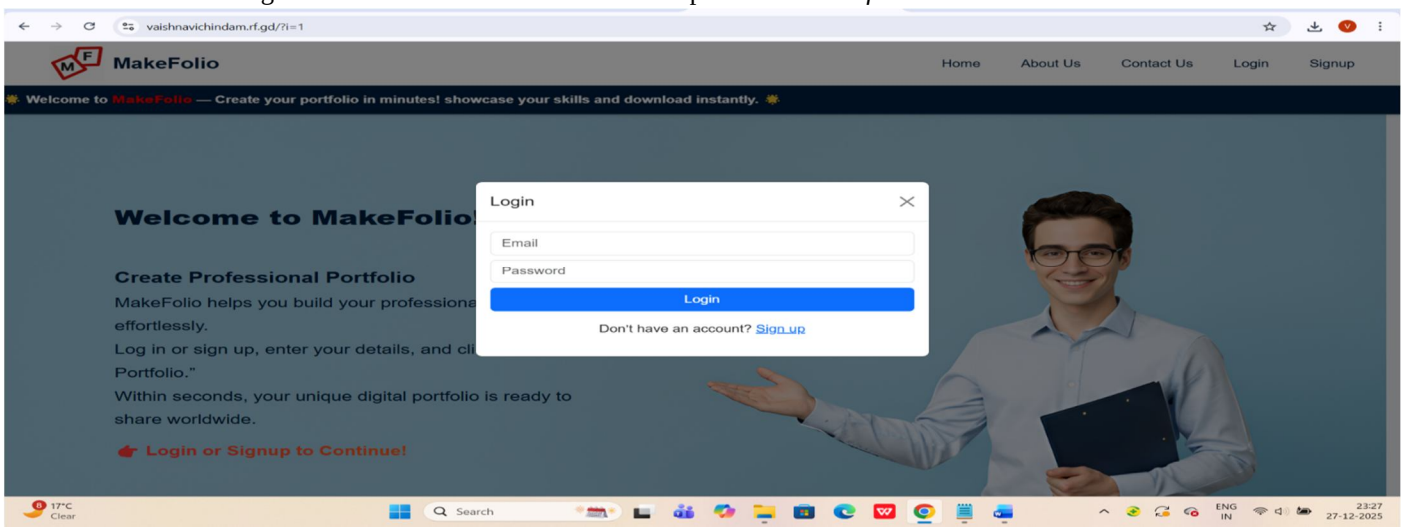
A. Homescreen

The home screen of the MakeFolio system serves as the entry point for users. Developed using HTML, CSS, and Bootstrap, it introduces the purpose of the platform—enabling users to create professional digital portfolios easily. The interface provides navigation options such as Home, Register, and Login, guiding users toward *portfolio creation.*



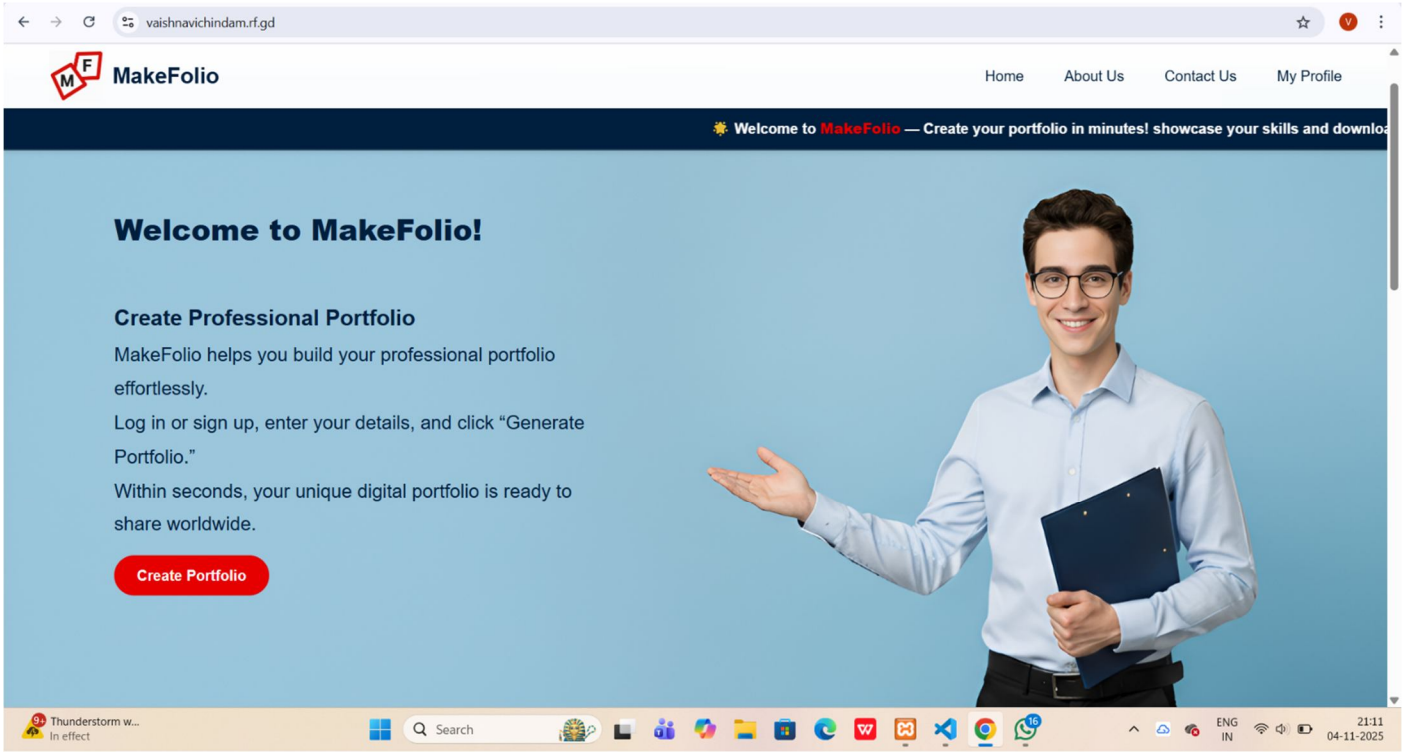
B. Login Screen

The login screen acts as a secure access point for registered users. Designed using PHP and HTML forms, it validates user credentials against the MySQL database. On successful authentication, a session is created and the user is redirected to the dashboard. Error handling ensures incorrect credentials do not expose sensitive *information.*



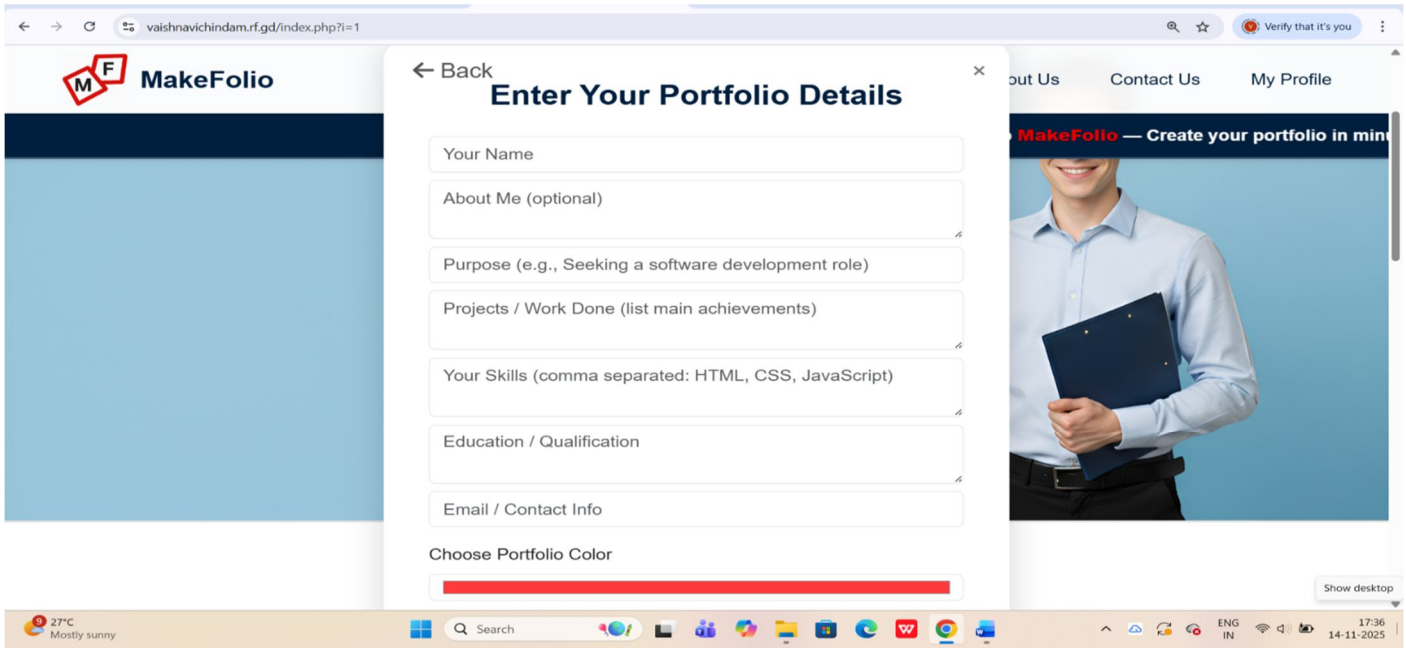
C. Dashboard

The dashboard serves as the central control panel of the MakeFolio system. After login, users can create a new portfolio, edit existing details, upload a profile image, choose color themes, preview their portfolio, and download it as a PDF. All dashboard actions are directly connected to backend *processing modules.*



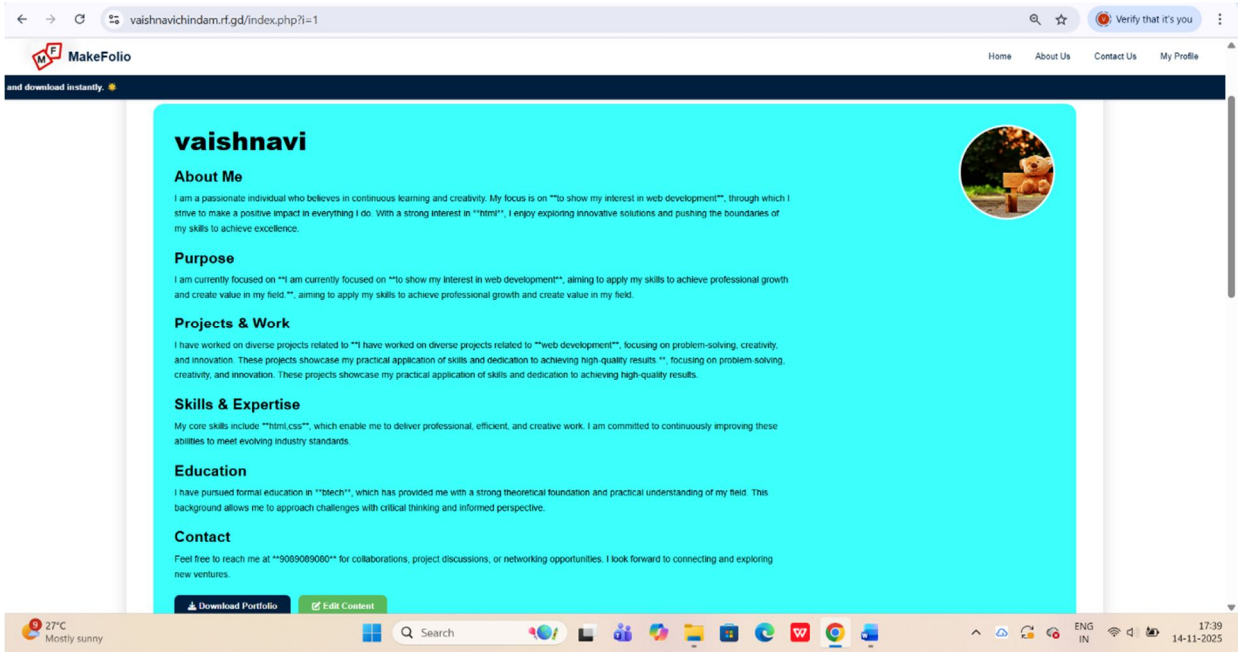
D. Portfolio Form Screen

This screen allows users to enter portfolio details such as personal information, skills, education, projects, and contact details. The data entered is validated and stored securely in the database. This structured input ensures consistent and professional portfolio generation.



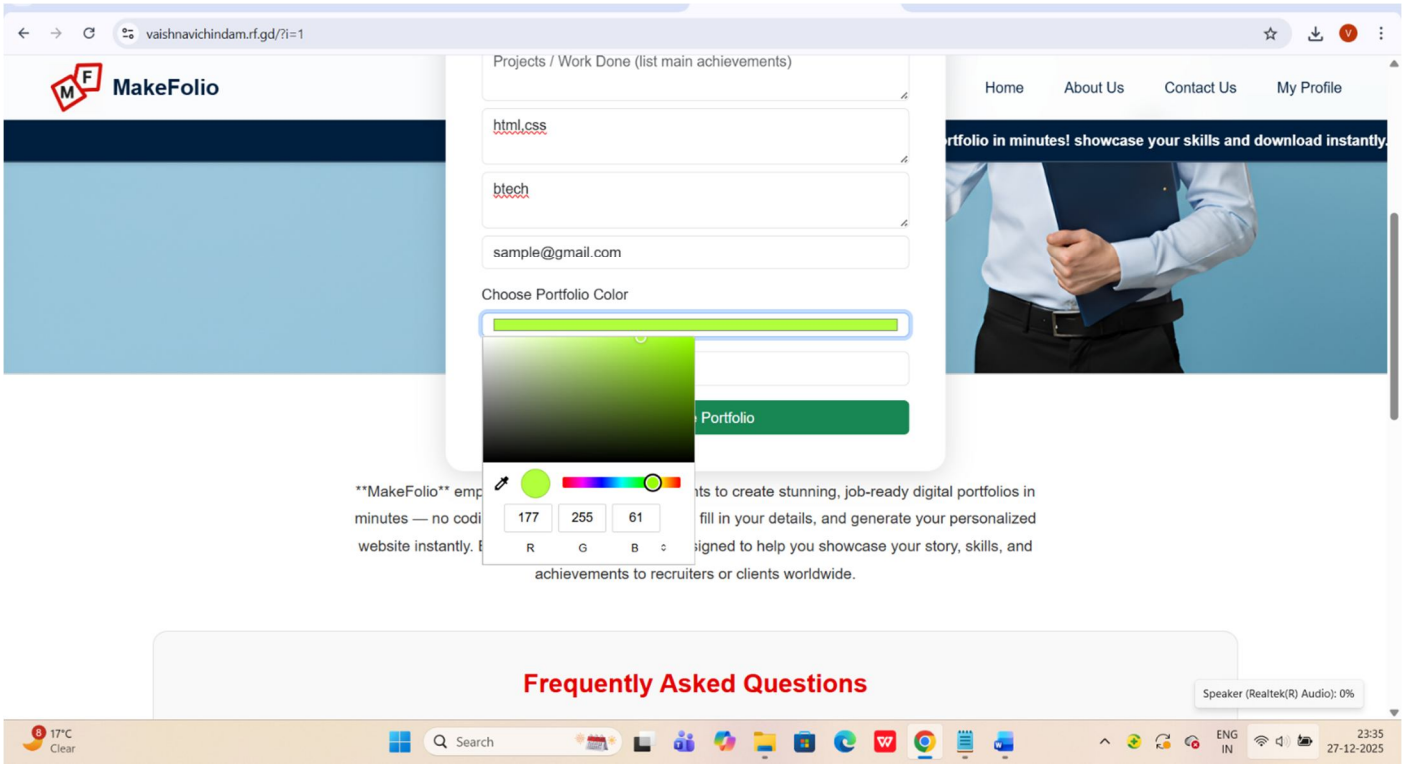
E. Portfolio Preview Screen

The portfolio preview screen dynamically displays the generated portfolio using HTML and CSS. Any changes made by the user—such as theme selection or image upload—are reflected instantly. This real-time preview helps users refine their portfolio before final download.



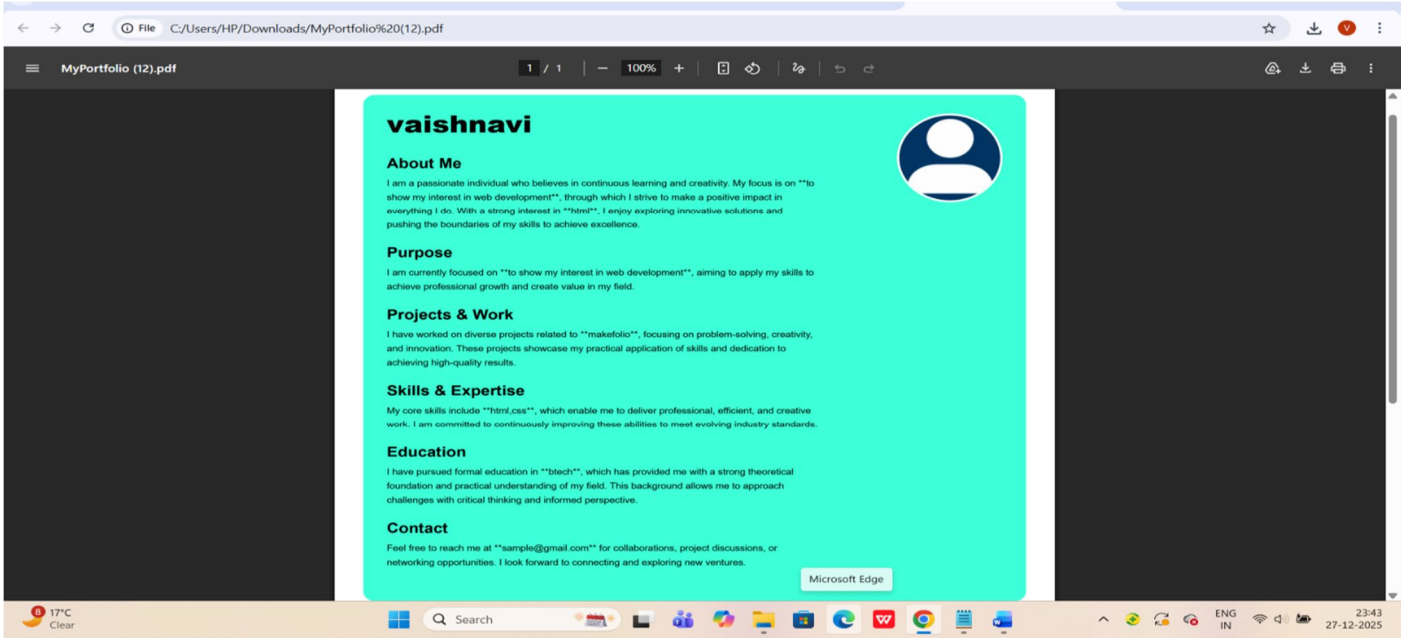
F. Customization Screen

The customization screen enables users to personalize their portfolio by selecting color themes and uploading a profile image. These customizations enhance visual appeal and allow users to match their portfolio style with *personal preferences.*



G. PDF Download Screen

The PDF download screen allows users to export their portfolio in PDF format. The system converts the dynamically generated HTML portfolio into a downloadable PDF file, making it easy to share with *recruiters and organizations.*



VI. FUTURE SCOPE

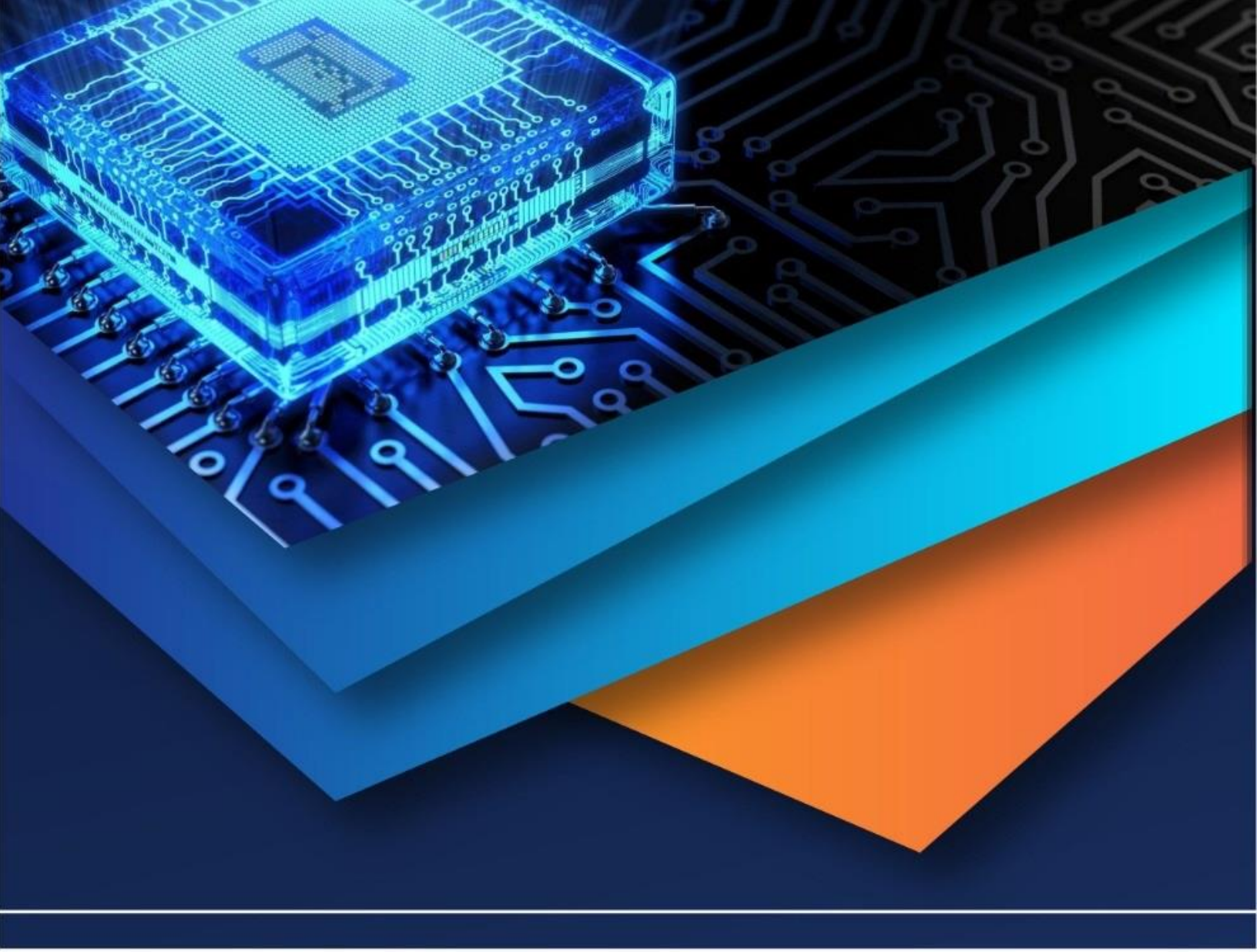
The MakeFolio system can be enhanced by adding more portfolio templates and advanced design customization options. Integration with cloud hosting can allow users to publish portfolios online using unique URLs. AI-based content suggestions can be introduced to automatically improve portfolio descriptions. Social media and LinkedIn integration can help users share portfolios easily. Support for multiple languages can expand usability for a wider audience. Mobile application support can further improve accessibility. With these enhancements, MakeFolio can evolve into a comprehensive digital branding platform.

VII. CONCLUSION

MakeFolio successfully addresses the need for a simple, efficient, and user-friendly platform to create and manage professional portfolios. It enables users to showcase their skills, projects, and achievements in a structured and visually appealing manner without requiring advanced technical knowledge. By providing customizable templates and easy content management, MakeFolio helps students, job seekers, and professionals build a strong online presence. Overall, the project demonstrates how modern web technologies can be used to simplify personal branding and enhance career opportunities in the digital era.

REFERENCES

[1] R. Nixon, Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5, 6th ed., O'Reilly Media, 2021.
[2] PHP Group, "PHP: Hypertext Preprocessor – Official Documentation," https://www.php.net/docs.php
[3] Oracle Corporation, "MySQL 8.0 Reference Manual," https://dev.mysql.com/doc/
[4] W3C, "HTML5 Specification," https://www.w3.org/TR/html5/
[5] W3C, "Cascading Style Sheets (CSS) — Level 3," https://www.w3.org/Style/CSS/
[6] Bootstrap Team, "Bootstrap Documentation," https://getbootstrap.com/docs/
[7] Mozilla Developer Network (MDN), "JavaScript Guide," https://developer.mozilla.org/en-US/docs/Web/JavaScript
[8] Parallax, "jsPDF – JavaScript PDF Generation Library," https://github.com/parallax/jsPDF
[9] Niklas von Hertzen, "html2canvas Documentation," https://html2canvas.hertzen.com/
[10] A. Tanenbaum and H. Bos, Modern Operating Systems, 4th ed., Pearson Education, 2015.
[11] OWASP Foundation, "Authentication Cheat Sheet," https://cheatsheetseries.owasp.org/
[12] Apache Software Foundation, "Apache HTTP Server Documentation," https://httpd.apache.org/docs/
[13] R. Pressman, Software Engineering: A Practitioner's Approach, 8th ed., McGraw-Hill, 2014.
[14] XAMPP Project, "XAMPP Apache + MySQL + PHP," https://www.apachefriends.org/
[15] Sommerville, I., Software Engineering, 10th ed., Pearson Education, 2016.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY