WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Modern Crowd Funding Platform

Babeetha shalini S[1], Karthick S[2], Navin k[3], Ms.kavithra[4], Ranjith kumar[5]

[1]Student, Department of computer Science and Engineering United Institute of Technology(An Autonomous Institution) Coimbatore,India babeethababeetha09@gmail.com

[2]Student,Department of computer Science and Engineering United Institute of Technology(An Autonomous Institution) Coimbatore,India karthickkd764@gmail.com

[3]Student,Department of computer Science and Engineering United Institute of Technology(An Autonomous Institution) Coimbatore,India nn4929071@gmail.com

[4]Assistant Professor,Department of computer Science and Engineering United Institute of Technology(An Autonomous Institution) Coimbatore,India kavithra.mtp@gmail.com

[5]Student,Department of computer Science and Engineering United Institute of Technology(An Autonomous Institution) Coimbatore,India ranjithkumarmahesan@gmail.com

Abstract—

This paper presents the design and development of a backend system for a funding platform that connects startups with potential investors. The primary objective of the system is to facilitate secure, scalable, and efficient management of data and user interactions through the use of modern web technologies. The backend is developed using Node.js and Express.js, while MongoDB serves as the database to store user and project information. The system supports distinct user roles—startups and investors—with appropriate access control mechanisms. JSON Web Tokens (JWT) and crypt are employed for secure authentication and password management, ensuring confidentiality and data integrity. The RESTful API architecture promotes modularity and reusability, allowing seamless integration with various frontend clients. Furthermore, the system utilizes environment-based configurations to enhance deployment flexibility. This backend framework serves as a robust foundation for building comprehensive funding solutions that streamline the investment process, reduce manual intervention, and promote real-time engagement between stakeholders.

Index Terms— Funding platform, backend development, Node.js, Express.js, MongoDB, REST API, user authentication, JSON Web Token (JWT), b crypt, web application, startup investment, role-based access control, secure data handling, scalable architecture, MERN stack.

INTRODUCTION

In recent years, the demand for digital platforms that bridge the gap between startup enterprises and potential investors has significantly increased. Traditional methods of securing funding involve time-consuming processes, manual documentation, and a lack of transparency, which often lead to inefficiencies in the funding ecosystem. With the advancement of web technologies and cloud-based infrastructures, there is a critical need to develop secure, scalable, and efficient backend systems that can support the dynamic requirements of such platforms.

This paper introduces the backend architecture of a funding platform designed to streamline interactions between startups and investors. The backend system is developed using Node.js and Express.js to ensure asynchronous, event-driven operations and scalable API management. MongoDB is utilized as the primary database, providing a NoSQL structure that accommodates dynamic data and offers high performance for read/write operations. Security is a key consideration in the system, achieved through the integration of JSON Web Tokens (JWT) for stateless authentication and bcrypt for secure password hashing. The architecture enforces role-based access control, distinguishing user privileges between investors and startups to maintain data integrity and system usability. RESTful APIs facilitate interaction with frontend clients and other external systems, promoting modularity and ease of maintenance. The backend also incorporates environment-based configuration using dot, enabling flexible deployment across various stages of development and production. This approach not only improves system reliability but also supports future scalability and integration with other services.
This paper discusses the design rationale, implementation details, and key technologies employed in developing the backend system. The proposed solution addresses the shortcomings of existing platforms and offers a robust infrastructure for building future-ready financial interaction portals.

SYSTEM ARCHITECTURE

The Crowd Funding Platform Website is designed using a three-tier architecture consisting of the presentation layer (frontend), application layer (backend), and data layer (database). This modular architecture enhances scalability, maintainability, and security.

- *Frontend (Presentation Layer)*

The frontend is built using **React.js**, providing a responsive and dynamic user interface. It includes pages for user registration, login, campaign creation, campaign browsing, and contribution. React's component-based structure allows for reusable UI components and efficient rendering.

- *Backend (Application Layer)*

The backend is developed using **Node.js** with the **Express.js** framework. It handles core functionalities including:

- User authentication and authorization using JWT
- Campaign creation and management
- Payment and transaction processing (integration-ready with gateways like Stripe/PayPal)
- API endpoints to serve frontend requests

The backend follows RESTful principles and includes middleware for error handling, input validation, and logging.

- *Database (Data Layer)*

The platform uses **MongoDB**, a NoSQL document-based database, for data storage. Collections are used to store user profiles, campaigns, transactions, and comments. Mongoose ODM is utilized for schema definition and query optimization.

- *Security Measures*

Security is enforced through:

- **JWT-based authentication**
- **HTTPS protocol** for secure data transmission
- **Input sanitization** to prevent injection attacks
- **Password hashing** with bcrypt

- *Deployment*

The application can be containerized using **Docker** and deployed on cloud platforms like **Heroku** or **AWS**. The architecture allows horizontal scaling to accommodate increasing user loads.

IMPLEMENTATION

The implementation of the Crowd Funding Platform Website focuses on developing a secure, scalable, and user-friendly web application. It is divided into the frontend development, backend development, database management, and system integration.

- *Frontend Implementation*

The frontend was implemented using **React.js**, a JavaScript library for building user interfaces. Key UI features include:

- **User Authentication Forms** – Registration and login forms with client-side validation.
- **Campaign Dashboard** – Allows users to create new campaigns, upload images, and track funding progress.
- **Explore Page** – Lists all active campaigns with filters based on categories, popularity, and date.
- **Contribution Workflow** – Users can view campaign details and contribute via a simple and intuitive form.

The UI is designed to be responsive, ensuring compatibility across desktop and mobile devices.

- *Backend Implementation*

The backend is developed using **Node.js** and **Express.js**, providing a lightweight yet powerful environment for building scalable web services.

- **API Development**: RESTful APIs handle user requests, such as campaign creation, donation transactions, and comment posting.
- **Authentication**: Implemented using **JWT (JSON Web Tokens)**, ensuring secure login sessions.
- **File Handling**: Campaign images and assets are stored and managed using middleware like Multer.
- **Validation & Error Handling**: All incoming data is validated using Joi, and a centralized error handler ensures proper debugging and logging.

- *Database Integration*

The platform uses **MongoDB** as its NoSQL database due to its flexibility and scalability. Data is structured into collections:

- **Users**: Stores account information, including email, password (hashed), and roles.
- **Campaigns**: Contains campaign title, description, funding goal, current amount, deadlines, and creator ID.
- **Transactions**: Logs each donation with amount, donor ID, campaign ID, and timestamp.

Mongoose ODM is used for schema definition and seamless interaction with MongoDB.

- *Payment Integration (Optional/Future Work)*

While the current implementation uses mock transactions for development, it is designed for easy integration with payment gateways like **Stripe** or **PayPal** through SDKs and webhooks.

- *Hosting and Deployment*

The project can be containerized using **Docker** and deployed on platforms like **Heroku**, **Vercel**, or **AWS EC2**. Environment variables are used to manage sensitive data and ensure secure deployment.

CONCLUSION

The Crowd Funding Platform Website successfully demonstrates the potential of web technologies in facilitating digital fundraising. By integrating a robust backend with a responsive frontend and secure database architecture, the platform offers a complete solution for users to initiate and contribute to fundraising campaigns.

The system provides core functionalities such as user authentication, campaign management, and contribution tracking, all while maintaining scalability and security. The modular structure of the application ensures that it can be easily extended to incorporate additional features such as real-time analytics, advanced payment integrations, and administrative tools.Future improvements could include AI-based campaign recommendation systems, blockchain-based transaction transparency, and integration with mobile platforms to expand reach and usability. Overall, the platform represents a modern and efficient approach to online crowd funding, empowering users and communities to bring their ideas to life.

ACKNOWLEDGEMENT

The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ... .” **Sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page, not here. (** The authors would like to thank the faculty and staff of the Computer Science and Engineering Department for their guidance and support throughout the development of this project.)

References

[1] E. Belleflamme, T. Lambert, and A. Schwienbacher, “Crowdfunding: Tapping the right crowd,” *Journal of Business Venturing*, vol. 29, no. 5, pp. 585–609, 2014.

[2] A. Agrawal, C. Catalini, and A. Goldfarb, “The geography of crowdfunding,” *NBER Working Paper*, no. 16820, 2011.

[3] M. Mollick, “The dynamics of crowdfunding: An exploratory study,” *Journal of Business Venturing*, vol. 29, no. 1, pp. 1–16, 2014.

[4] T. Belleflamme et al., “Crowdfunding: An industrial organization perspective,” *Digital Economy Conference*, 2012.

[5] S. Sharma, R. Chatterjee, and M. Gupta, “Design and implementation of a secure web-based payment system,” *International Journal of Computer Applications*, vol. 139, no. 4, pp. 22–29, 2016.

[6] J. Rosenberg and A. Mateosian, “Building Secure and Scalable Web Applications,” *IEEE Internet Computing*, vol. 22, no. 5, pp. 74–81, 2018.

[7] K. Wang, Y. Yang, and X. Wu, “A survey on security issues in online payment systems,” *Journal of Computer Science and Technology*, vol. 30, no. 3, pp. 561–575, 2015.

[8] N. Aggarwal and D. Dahiya, “Web-based crowdfunding platform using MERN stack,” *International Journal of Engineering Research & Technology (IJERT)*, vol. 10, no. 4, pp. 345–349, 2021.

[9] J. Kaur and P. Singh, “User experience design for web applications,” *IEEE Int. Conf. Human-Computer Interaction*, pp. 89–94, 2017.

[10] S. Patel, “MEAN and MERN Stack Web Development,” *TechPress*, 2020.

[11] R. Fielding, “Architectural styles and the design of network-based software architectures,” *Doctoral dissertation, University of California, Irvine*, 2000.

[12] D. Crockford, “JSON: The fat-free alternative to XML,” *IEEE Software*, vol. 24, no. 2, pp. 90–93, 2007.

[13] D. D’Souza and A. Joshi, “Microservices architecture: Make the architecture shine in cloud,” *IEEE Software*, vol. 34, no. 5, pp. 81–85, 2017.

[14] J. Bell, “Using MongoDB with Node.js and Mongoose,” *WebDev Journal*, vol. 7, no. 2, pp. 14–18, 2019.

[15] Stripe Inc., “Stripe API Documentation,” [Online]. Available: https://stripe.com/docs/api