Chapter 1

Introduction

Front-end development is the process of creating the user interface and user experience of a website or application. Front-end developers are responsible for implementing the design, functionality, and interactions of a website or application. This involves writing code in languages such as HTML, CSS, and JavaScript to build web pages that can be accessed by users through their browsers.

The role of a front-end developer has become increasingly important as more and more businesses and organizations rely on web applications to reach their audience. With the rise of e-commerce and online services, websites have become a crucial part of any company's marketing strategy. Front-end developers are essential to creating user-friendly websites that are easy to navigate and aesthetically pleasing.

Front-end developers work closely with designers and back-end developers to ensure that a website or application functions seamlessly. They take the design concepts provided by the designers and turn them into a functional website or application. This involves writing code to create the layout, typography, and visual elements of the website or application, as well as developing the functionality of the site.

The scope of full-stack web development is broad and multifaceted, requiring proficiency in a diverse set of technical skills, tools, and methodologies. Full-stack developers must possess a deep understanding of front-end and back-end technologies, database management, version control systems, and deployment practices to architect, develop, and maintain robust and scalable web applications.

As the demand for websites and applications continues to grow, the demand for skilled front-end developers will also continue to rise.

1.1 Full Stack Web Development Process

Full stack development refers to the end-to-end application software development, including the front end and back end. The front end consists of the user interface, and the back end takes care of the business logic and application workflows. The mani components of full stack development is shown in Fig. 1.1.

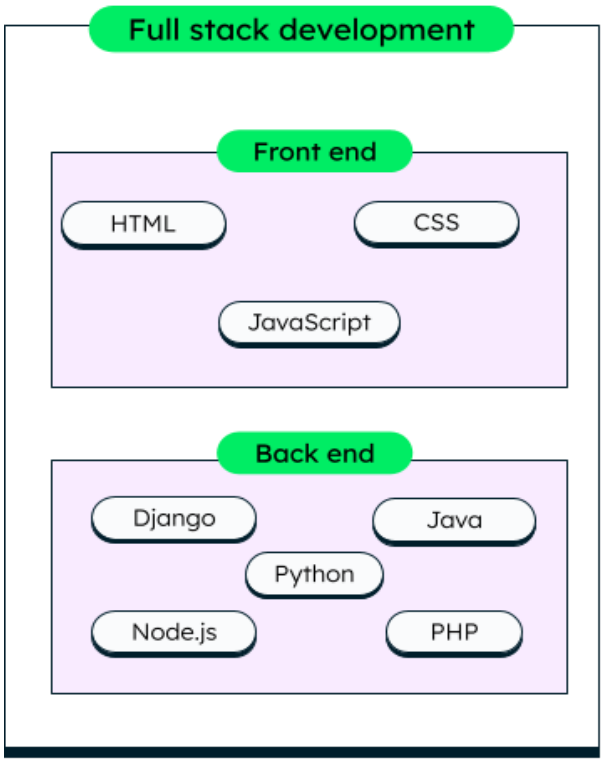

Fig. 1.1 Main components.

Consider a retail website. Users can browse or purchase specific items, delete or add items in cart, change their profile, and do many other things. All these actions require a front-end user interface (UI), as well as some business logic, written in the back-end.

- The website UI can be built using various, front-end technologies like HTML, CSS, JavaScript.
- The back end is written in programming languages like Java or Python. Further, a good web application would need scalability, event handling, and routing, which are usually handled by libraries and frameworks like Spring Boot or Django.
- The back end also consists of logic that can connect the application to other services and databases. For example, all the user and transaction data are stored in a database through specific drivers handled on the back end.

A full stack developer is one who can single-handedly implement both the front-end and back-end workflows, like placing the order or changing the user profile.

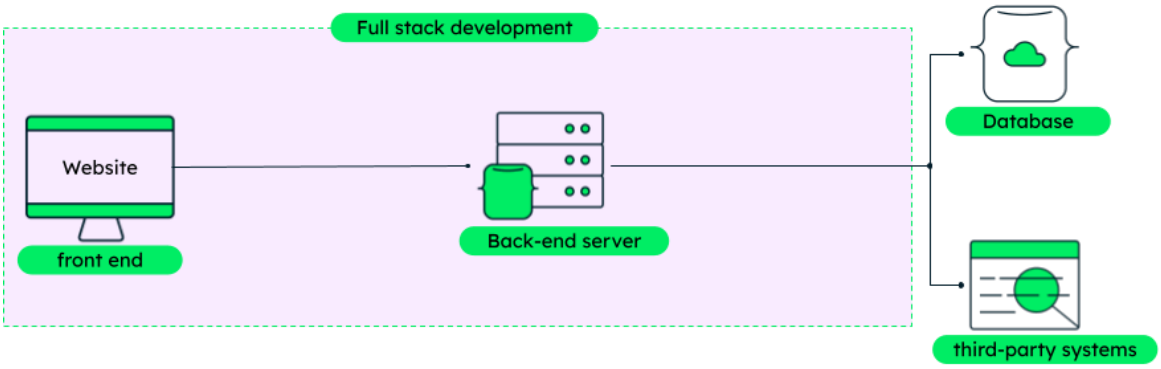

Fig. 1. 2 Demonstration of end-to-end workflow

In essence, full-stack development is the practice of building complete web applications, the end-to-end workflow is shown in Fig. 1.2, encompassing both the parts the user sees and interacts with (front end) and the behind-the-scenes components that handle data and logic (back end).

- **Website (Front End):** This is the visual, interactive side of the application. It involves technologies like:
  - **HTML:** For basic structure
  - **CSS:** For styling (colours, fonts, layout)
  - **JavaScript:** For dynamic behaviour (animations, user interactions)
- **Back-end Server:** This is the brain of the system – it's not directly seen by users. It handles:
  - **Programming Languages:** Like Python, Java, Node.js, etc.
  - **Frameworks:** Such as Django, Ruby on Rails, Express.js (these make development easier)
  - **API Development:** Creating ways for the front end to communicate and fetch data
- **Database:** This is where information is stored and retrieved. Options include:
  - **Relational Databases:** MySQL, PostgreSQL
  - **Non-Relational Databases:** MongoDB, Cassandra
- **Third-Party Systems:** The application might interact with additional services, such as:
  - Payment systems (Stripe, PayPal)
  - Cloud providers (AWS, Google Cloud, Azure)
  - Email delivery (Mailchimp, SendGrid)

How the working is together shown in Fig. 1.3 is as follows:

1. **User Request:** A user visits the website in their browser.
2. **Front End Render:** The browser loads the HTML, CSS, and JavaScript, creating the web page.
3. **User Interaction:** User clicks a button, fills a form, etc.
4. **API Call:** JavaScript sends a request to the back-end server via an API, seeking data or action.
5. **Back-end Processing:** The server processes the request, which might involve interacting with the database.
6. **Data Response:** The server sends data back to the front end.
7. **Website Update:** The front end updates the website dynamically based on the received data.

1.1.1 Front-end vs Back-end vs Full stack

Applications that require higher scalability and more complex workflows require broader skill sets and collaboration across teams. For example, the front end may be handled by the UI team, and the back end by another team. In some organizations, individuals will be required to work on both the front-end and back-end implementation of a feature. This is where full stack developers would come into play.

Developer Type	Responsibilities	Technologies Used
Front-end	Handle the UI, including visual effects, frames, navigation, and forms. Focus on user experience.	HTML, CSS, JavaScript
Back-end	Manage business logic, security, performance, scalability, and request-response handling. Design core application workflows using frameworks.	JavaScript, Python, Java, .NET

Full Stack	Code end-to-end workflows using both front-end and back-end technologies. Utilize stacks like MERN or MEAN for building complete applications.	HTML, CSS, JavaScript, Python, Java, .NET, etc.

Table 1.1 Front-end vs Back-end vs Full stack

The Table 1.1 categorizes developers into three main types based on their responsibilities and technologies used in web development: Front-end, Back-end, and Full Stack.

1.1.2 Advantages and Disadvantages of Full Stack Development

Advantages	Disadvantages
Thorough understanding of entire web development process	Broad range of skills, may lack depth in specific areas
More efficient and effective development process	Need to keep up with new developments in multiple technologies
Creation of cohesive and user-friendly websites and applications	Complexity and time-consuming nature, not ideal for small projects or tight deadlines

Table 1.2 Advantages and Disadvantages of Full Stack development

The Table 1.1 summarizes the advantages and disadvantages of full-stack development to understand both the benefits and challenges associated with this approach.

1.1.3 Features of Full Stack Web Development

Full-stack web development embodies a comprehensive approach to building web applications, characterized by a multitude of features and capabilities. At its core, full-stack development empowers developers to navigate the entire spectrum of web development

tasks, from conceptualization to deployment, ensuring a cohesive and integrated end product.

Features of Full-Stack Web Development:

- **End-to-End Development:** Full-stack web development encompasses the entire process of creating web applications, from designing user interfaces to implementing server-side logic and managing databases. Developers have the ability to work on all layers of the application stack, ensuring a cohesive and integrated approach to development.
- **Front-End Technologies:** Full-stack developers are proficient in front-end technologies such as HTML, CSS, and JavaScript, as well as modern frameworks and libraries like React.js, Angular, or Vue.js. They create visually appealing and interactive user interfaces that enhance user experience and engagement.
- **Back-End Technologies:** Full-stack developers utilize server-side scripting languages such as JavaScript (Node.js), Python, Ruby, or Java, along with frameworks like Express.js, Django, Flask, or Spring Boot, to implement business logic and handle data processing tasks. They interact with databases, manage server-side sessions, and ensure the security and scalability of the application.
- **Database Management:** Full-stack developers are proficient in designing, implementing, and managing databases using both relational (e.g., MySQL, PostgreSQL) and NoSQL (e.g., MongoDB, Firebase) databases. They use ORMs (Object-Relational Mappers) or ODMs (Object-Document Mappers) to interact with databases and perform CRUD (Create, Read, Update, Delete) operations.
- **Version Control Systems:** Full-stack developers utilize version control systems such as Git to track change, manage codebase history, and facilitate collaboration among team members. They employ branching and merging strategies, create pull requests, and resolve conflicts to ensure code quality and consistency.
- **Deployment and Infrastructure:** Full-stack developers configure web servers, application containers, and cloud infrastructure (e.g., AWS, Azure, Google Cloud Platform) to host and deploy web applications. They set up continuous integration (CI) and continuous deployment (CD) pipelines to automate testing, build, and deployment processes, ensuring efficient and reliable deployments.

- **Responsive Design and Cross-Platform Compatibility:** Full-stack developers implement responsive design principles to ensure that web applications are optimized for different devices and screen sizes. They utilize CSS frameworks like Bootstrap or Materialize to create mobile-friendly layouts and ensure cross-platform compatibility.
- **Security Considerations:** Full-stack developers incorporate security best practices throughout the development process, including data encryption, authentication, authorization, and protection against common security threats such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
- **Performance Optimization:** Full-stack developers optimize the performance of web applications by minimizing loading times, reducing server response times, and implementing caching mechanisms. They use tools like performance monitoring and profiling to identify bottlenecks and optimize critical paths.
- **Continuous Learning and Adaptation:** Full-stack developers continuously stay updated with emerging technologies, trends, and best practices in web development. They engage in lifelong learning, attend workshops, conferences, and online courses, and participate in developer communities to expand their skills and stay ahead in a rapidly evolving field.

Overall, full-stack web development offers a comprehensive and integrated approach to building modern web applications, empowering developers to create scalable, secure, and user-friendly experiences across various platforms and devices.

1.2 Scope

Full-stack web development encompasses a comprehensive range of tasks and responsibilities involved in creating dynamic and interactive web applications. At its core, full-stack development involves both front-end and back-end aspects, along with database management, version control, and deployment strategies.

Deployment and infrastructure management are essential considerations in full-stack development, involving the configuration of web servers, application containers, and cloud platforms to host and deploy web applications. Continuous integration (CI) and continuous deployment (CD) pipelines automate testing, build, and deployment processes, enabling rapid iteration and delivery of features to production environments.

The Book Blog Web Application represents a convergence of frontend and backend technologies aimed at providing users with an immersive and engaging literary experience. By leveraging full-stack development principles, this application offers a seamless integration of various features to enhance user interaction and content discovery.

At the heart of the Book Blog Web Application is its Dynamic Book Display feature. Users are presented with a curated selection of books, each accompanied by comprehensive information such as titles, authors, descriptions, and visually appealing images. This dynamic display not only showcases the breadth and depth of the literary collection but also entices users to explore further and discover new works of interest.

User-friendly Navigation lies at the core of the application's usability. Intuitive navigation elements empower users to effortlessly navigate between different sections of the platform, facilitating a smooth and intuitive browsing experience. Whether users are exploring new releases, browsing by genre, or searching for specific titles, the seamless navigation ensures they can access relevant content with ease, enhancing overall user satisfaction.

The Search Functionality feature serves as a powerful tool for users to efficiently locate specific books within the vast repository. With the ability to search by keywords, authors, or categories, users can quickly pinpoint desired titles and access relevant content without unnecessary navigation hurdles. This robust search feature not only enhances the discoverability of content but also streamlines the user experience, enabling users to find what they are looking for in a timely manner.

Furthermore, the Admin Dashboard empowers administrators with comprehensive control and management capabilities. Through a dedicated dashboard interface, administrators can oversee and manage book posts, including the addition of new entries, editing of existing content, and maintenance of overall platform integrity. The Admin Dashboard serves as a centralized hub for administrators to curate and optimize the content presented to users, ensuring relevance and quality across the platform.

In essence, the Book Blog Web Application embodies the essence of full-stack development by seamlessly integrating frontend and backend technologies to deliver a compelling user experience. Through its dynamic book display, user-friendly navigation, robust search functionality, and administrative capabilities, the application fosters a vibrant literary community where users can explore, discover, and engage with rich multimedia content in an intuitive and immersive environment.

1.3 Importance of Full Stack Web development

Full-stack web development holds paramount importance in the realm of modern software engineering and digital innovation. Its significance lies in the holistic understanding and proficiency it provides across various layers of web application development.

Firstly, full-stack developers possess a comprehensive grasp of both front-end and back-end technologies, which enables them to bridge the gap between design and functionality. This integrated approach fosters efficient collaboration within development teams, as developers can seamlessly transition between different components of the application. By having a nuanced understanding of user interface design, client-side scripting, server-side logic, and database management, full-stack developers contribute to the creation of cohesive and feature-rich web applications.

The importance of the Book Blog Web Application is multifaceted and encompasses various aspects that contribute to its significance in the digital landscape.

Firstly, the application serves as a gateway to literary exploration and discovery. In a world inundated with digital distractions, the platform provides users with a curated selection of literary works, spanning diverse genres, authors, and themes. By showcasing a wide range of books and authors, the application encourages users to explore new literary horizons, discover hidden gems, and expand their reading repertoire. In doing so, it fosters a culture of curiosity, lifelong learning, and intellectual enrichment among its user community.

Moreover, the Book Blog Web Application plays a pivotal role in fostering community engagement and interaction within the literary sphere. Through user-generated content, book reviews, and discussion forums, users can connect with like-minded individuals, share their literary experiences, and exchange recommendations and insights. The platform serves as a virtual meeting place for book enthusiasts to engage in meaningful dialogue, forge connections, and celebrate their shared love for literature. In essence, it cultivates a sense of camaraderie, belonging, and mutual support among its diverse user base.

Additionally, the application serves as a catalyst for the promotion of reading culture and literacy initiatives. By making literary content accessible, engaging, and interactive, the platform inspires users to embrace the joys of reading, explore new literary landscapes, and immerse themselves in the world of storytelling. Through book recommendations, author

spotlights, and curated reading lists, the application empowers users to embark on enriching literary journeys, fostering a lifelong love affair with books and literature.

Furthermore, the Book Blog Web Application facilitates content curation and quality control, ensuring that users have access to relevant, high-quality literary content. Through the admin dashboard, administrators can curate and manage the platform's content, ensuring that it aligns with the interests, preferences, and standards of the user community. By prioritizing content relevance, diversity, and authenticity, the application fosters a curated user experience that is both informative and engaging.

1.4 Problem Statement

Traditionally, web development was compartmentalized into distinct roles such as front-end developers responsible for building user interfaces using HTML, CSS, and JavaScript, and back-end developers tasked with implementing server-side logic and database management. This compartmentalization often led to siloed workflows, communication challenges, and slower development cycles.

In the context of previous work, the challenge lies in enhancing the existing book blog web application by introducing an admin sign-in feature for authentication purposes. Aims to enforce strict access control, permitting only authenticated users to add, create, and delete blogs. By implementing an admin sign-in mechanism, the application seeks to elevate its security posture while ensuring the integrity and authenticity of published content.

Another area of improvement centres on the search functionality within the application. Previously, users were limited to searching for books based solely on their names and authors. Description-based search functionality, empowering users to explore the vast repository of book reviews with greater precision and efficiency. This enhancement reflects a strategic effort to enhance user experience and facilitate content discovery within the platform.

Moreover, the integration of MongoDB as the database management system signifies a significant departure from traditional relational database models, which may have struggled to accommodate evolving data schemas and dynamic content structures, MongoDB offers unparalleled flexibility and adaptability. Its document-oriented architecture allows for seamless storage and retrieval of diverse data types, enabling the platform to scale

gracefully and accommodate future enhancements without sacrificing performance or stability.

Additionally, the incorporation of images alongside textual content, while previous versions may have relied solely on textual descriptions to convey information about books, the inclusion of images enhances visual appeal and aids users in quickly assessing the relevance and appeal of a particular book. This visual enhancement not only enriches the user experience but also aligns with contemporary design trends and user preferences in digital content consumption.

1.5 Objective

The main objective of this Internship project is to learn, apply and gain industrial skills and knowledge in the field of Full Stack Web Development. This project uses the knowledge gained and demonstrate a website build using Full Stack Web Development technologies learned through the course of internship. A blog website is created that by applying theoretical knowledge and practical skills acquired during the internship. The creating of blog website to learn essential software engineering principles such as code organization, documentation, testing, and debugging. The website displays a list of book reviews. Each book contains the title, author name, description, and image of the book. The users can scroll through different book titles and read any blog of it. A search can be done based on book name. The user can surf through latest collections from the various collection present in the database. An admin login allows only authorized users to add, edit, and delete the blogs in the website.

1.6 System Design

The system design shown in Fig. 1.3 shows the basic flow of the data in overall project. The design consists of the following:

- **Admin** - Creates and manages the blog. They can add, update, and delete blog entries.
- **Blog Reader** – View and read the blog entries.
- **Blog Site** - Public-facing website where the blog entries are published.
- **Server** - Computer that stores the blog entries and other data.
- **Data Storage** - Blog entries and other data are stored on the server.

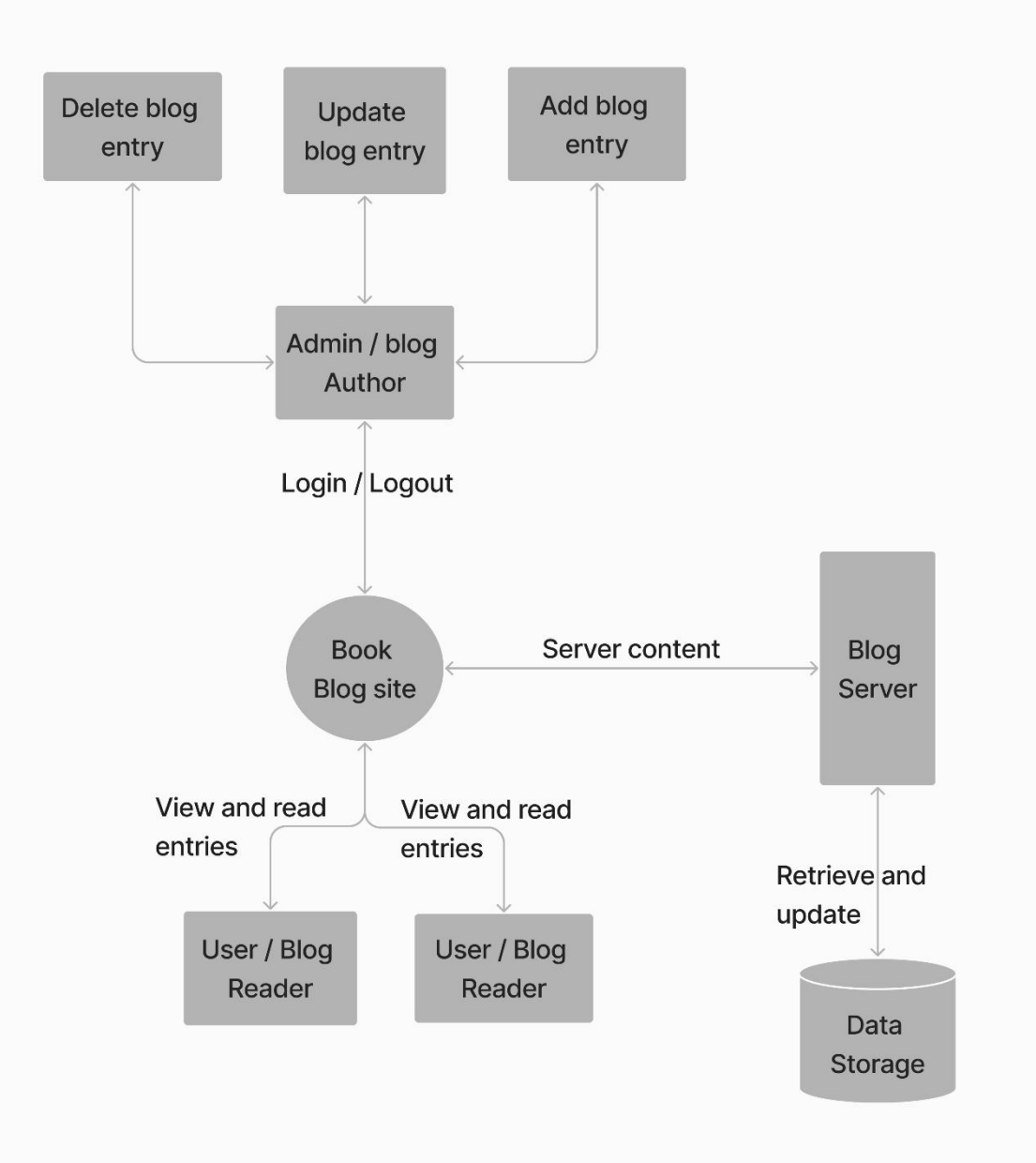

Fig. 1.3 System design

Chapter 2

Organization Profile

Nikkibuild Edtech Private Limited (NEPL) is a Private Limited Indian Non-Government Company incorporated in India on 12 October 2021. Its registered office is in Bangalore, Karnataka, India.

nikki.build is a web-based Rapid prototyping and applied learning platform that simplifies software prototyping using various technologies such as IoT, Robotics, Open Source, AI/ML, Microcontrollers, and more.

The platform empowers students, educators, hobbyists, and professionals to learn, collaborate, create, and innovate without barriers in Real time. Users can contribute their creations to the platform's GitHub community, which can be used by others to build their prototypes, fostering a culture of innovation and knowledge-sharing, enabling our users to create cutting-edge projects that push the boundaries of what's possible.

Nikkibuild Edtech Private Limited, founded on October 12, 2021, in Bangalore, stands as a beacon of innovation and progress in the realm of education technology. As a non-government entity classified under the RoC-Bangalore jurisdiction, Nikkibuild operates as a private company limited by shares, driven by a vision to revolutionize the educational landscape. Led by directors Arunkumar Shivaputrayya Hiremath and Ranjitha Arunkumar, Nikkibuild's journey embodies a commitment to excellence and a dedication to advancing educational opportunities.

At its core, Nikkibuild Edtech is committed to higher education, focusing on post-secondary and senior secondary sub-degree level education that paves the way for university degrees or equivalents. This strategic focus positions the company at the

intersection of academic advancement and technological innovation, where it seeks to bridge the gap between traditional educational methods and the demands of the digital age.

One of Nikkibuild's distinguishing features is its innovative approach to educational technology. In a rapidly evolving digital landscape, the company recognizes the transformative potential of technology in enhancing learning outcomes and expanding access to education. By harnessing the power of digital platforms, multimedia resources, and interactive tools, Nikkibuild strives to create engaging and immersive learning experiences that cater to diverse learner needs and preferences.

Moreover, Nikkibuild's commitment to innovation extends beyond the classroom, encompassing the development of cutting-edge educational solutions and services. From personalized learning algorithms to adaptive assessment tools, the company leverages the latest advancements in artificial intelligence, machine learning, and data analytics to optimize educational delivery and student performance. By embracing emerging technologies, Nikkibuild aims to redefine the educational experience and empower learners to reach their full potential.

Central to Nikkibuild's mission is a focus on inclusivity and accessibility in education. Recognizing the diverse needs and backgrounds of learners, the company endeavors to create inclusive learning environments that cater to learners of all abilities and backgrounds. Through the development of accessible educational resources, adaptive learning platforms, and inclusive pedagogical approaches, Nikkibuild seeks to break down barriers to education and promote equitable access to learning opportunities.

Furthermore, Nikkibuild's impact extends beyond the confines of traditional education settings, encompassing a range of initiatives aimed at promoting lifelong learning and skill development. Through partnerships with educational institutions, government agencies, and community organizations, the company facilitates the dissemination of knowledge and expertise to individuals across the socioeconomic spectrum. By offering flexible learning pathways and alternative credentialing options, Nikkibuild empowers learners to acquire new skills and competencies that are relevant to their personal and professional aspirations.

In addition to its focus on educational innovation, Nikkibuild Edtech is guided by a commitment to ethical and responsible business practices. Upholding the highest standards of integrity, transparency, and accountability, the company strives to foster trust and

confidence among its stakeholders. Through adherence to regulatory requirements, ethical guidelines, and industry best practices, Nikkibuild seeks to build enduring partnerships and relationships based on mutual respect and shared values.

Looking ahead, Nikkibuild Edtech Private Limited is poised to play a pivotal role in shaping the future of education in India and beyond. With its unwavering commitment to innovation, inclusivity, and excellence, the company is well-positioned to lead the charge towards a more accessible, equitable, and transformative educational landscape. As technology continues to evolve and society grapples with the challenges of the 21st century, Nikkibuild stands ready to champion the cause of education and empower learners to thrive in an increasingly complex and interconnected world.

Chapter 3

System Requirements

3.1 Hardware Requirements

- Platform : Windows 10 and 11
- Central Processing Unit (CPU): A dual-core processor with at least 2 GHz clock speed.
- Processor type : Intel core i3 processor or faster
- Processor speed : Minimum 2.4 GHz or faster
- Memory : 512 MB or more
- Hard Disk : 20GB or more
- Monitor : VGA or higher resolution 800x600 or higher resolution
- Pointing device : Microsoft Mouse or compatible pointing device
- CD-ROM : Actual requirements will vary based on system configuration
- Keyboard : 110 keys enhanced

3.2 Software Requirements

- Operating System: Windows 10 or 11.
- Front-end Language: HTML, CSS, JavaScript, ejs.
- Back-end Languages: Nodejs, ejs.
- Frameworks: Express.
- Database : MongoDB.
- Text Editor : VS Code.

Chapter 4

Assessments And Conclusion

4.1 Implementation

The project is built in a structure form. The required packages are downloaded first and the database connection is defined. The server and database connection are build first followed building the models and creating routes for each operations. The views are created for the front-end of the website. The public folder is created to store the basic css, javascript and images.

4.1.1 Page Navigations

The server connection and dependencies are contained in a JavaScript file called app.js. Here, first the required packages are included using require( ). The port number described in the dotenv file is created and the app is made to listen on that port. This file contains all the dependencies, database and middleware's used for the project.

The public is used as static so that anytime a script is required the path is automatically created. The public folder contains CSS, js, img as subfolders. The CSS folder contains styles used for the website. The js folder contains JavaScript functionalities required in the project. The img and uploads contains images for the project.

The views folder created contains the front-end files. The views contain layouts as subfolder which contains main.ejs for user site and admin.ejs for admin site which contains the header, main as body and footer for the website, the header and footer for these files are contained within partials folder. This file contents are displayed in each page in the front-end of the website.

The server folder is created to contain basic server routes, models, configurations, and helpers used for the project. The routes to each page in the project used in included here. This allows the get and post request to be performed on the page using the asynchronous function. The models contain the structure of the database which includes schemas used for the project. The config's folder contains database connection to mongoose. The helpers is used for active route.

4.1.2 Server

Config

Config consists of db.js which contains the mongoose connection. The database url is obtained from env file and build the connection.

Helpers

Contains function to return the current route. Used to identify the active route. Required so that user can understand on which page they are present in.

Models

This contains the structure of the database and database connection. The file's Post and User is used for creating relevant schemas. The Post.js contains schema for the blog which include of title, description, author, image, date on which it was created and updated. The User.js contains schema for username and password.

Routes

This contains app routes for the project. The main.js is created and a router object is created to handle middleware functions to handle requests and routers. The router main.js uses get to obtain the homepage, display post by referring their id, and about page using asynchronous function to apply relevant operations. The post method is used for posting search which ignores the special characters added and searches the book.

The home page post are displayed by counting the documents available in the database and performing aggregate to display from latest posts limiting to 10 per page. The remaining can be viewed in different pages based on the number of available posts. On clicking on of the post it leads to display the post. The id is set by get and blog is displayed. The search term is obtained by post operation which finds the book excluding any special characters used in the search term to fin in the database. Finally, the about page is created by rendering the page.

The router admin.js creates an authentication middleware which is responsible to generate token for every time an admin has logged in while not allowing any unauthorized person

to have admin properties. The get operations are used to display the admin login page, dashboard, add post, edit post based on id, and logout.

The post applies relevant operations for checking the valid credentials of the admin in admin login page, successful adding of new posts. To check admin login database is searched using findOne( ) operation of the username. The password validation is checked using bcrypt since the hash of the password was initially stored in the database. The posts are added to the database by request from the body of the particular field. The image is obtained as a file. The create( ) is performed on the post.

The put allows admin to edit the post of the particular id selected. Similar to add post the required field is obtained by request from body and applying findByIdAndUpdate( ) on the updated post.

The delete allows the deletion of respective post. The deletion occurs by obtaining the id of the particular post and applying deleteOne( ) operation on the id obtained from request.

4.1.3 Views

The index.ejs contains the operations for home page display. The book posts are displayed using forEach( ), which obtains each posts by looping over each document in reverse order as defined in the route to display in latest order. The about page contains basic details of about the blog. The posts can be viewed by clicking on the particular post which navigates to the post.ejs where the post is displayed based on its id, displays it's other properties. The search button operation is described and displays result in a separate page search.ejs, where the search term is obtained by its id and displayed.

Layouts

The layouts contain global layout of the website. It contains main.ejs and admin.ejs for user and admin respectively. The main.ejs is includes the header and footer for the user website and includes body for the global layout. Similarly, the admin.ejs includes the header and footer for the admin website and includes body for the global layout.

Partials

This folder contains the global header and footer for user and admin website, and the search operation ejs file. The header for users contains the logo, navigation to home, about and search button. Here, the helper is used to define the active route.

Admin

Admin page contains the index.ejs where the login page is defined. The dashboard which defines each posts title and option to edit or delete based on its id. The add-post.ejs contains form defined to fill and add the database. The edit-post.ejs is similar to add-post containing the form to fill with the original content already filled, so the admin can make changes in the required field.

4.2 Results

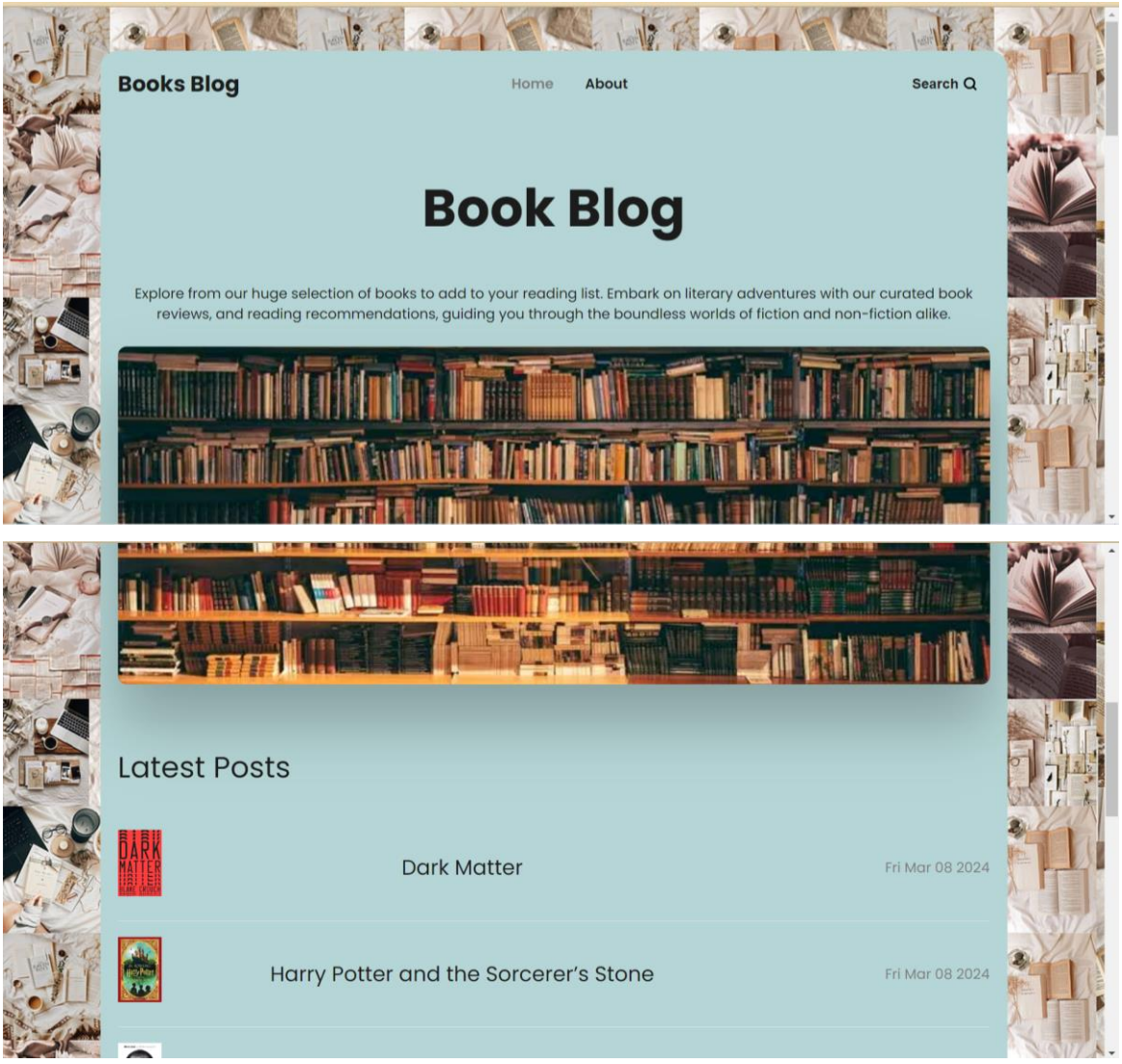

Fig. 4.1 Home Page

Home page is shown in Fig. 4.1, it displays navigation to other pages and a search bar to search for books. The user can select which book they want to read review.

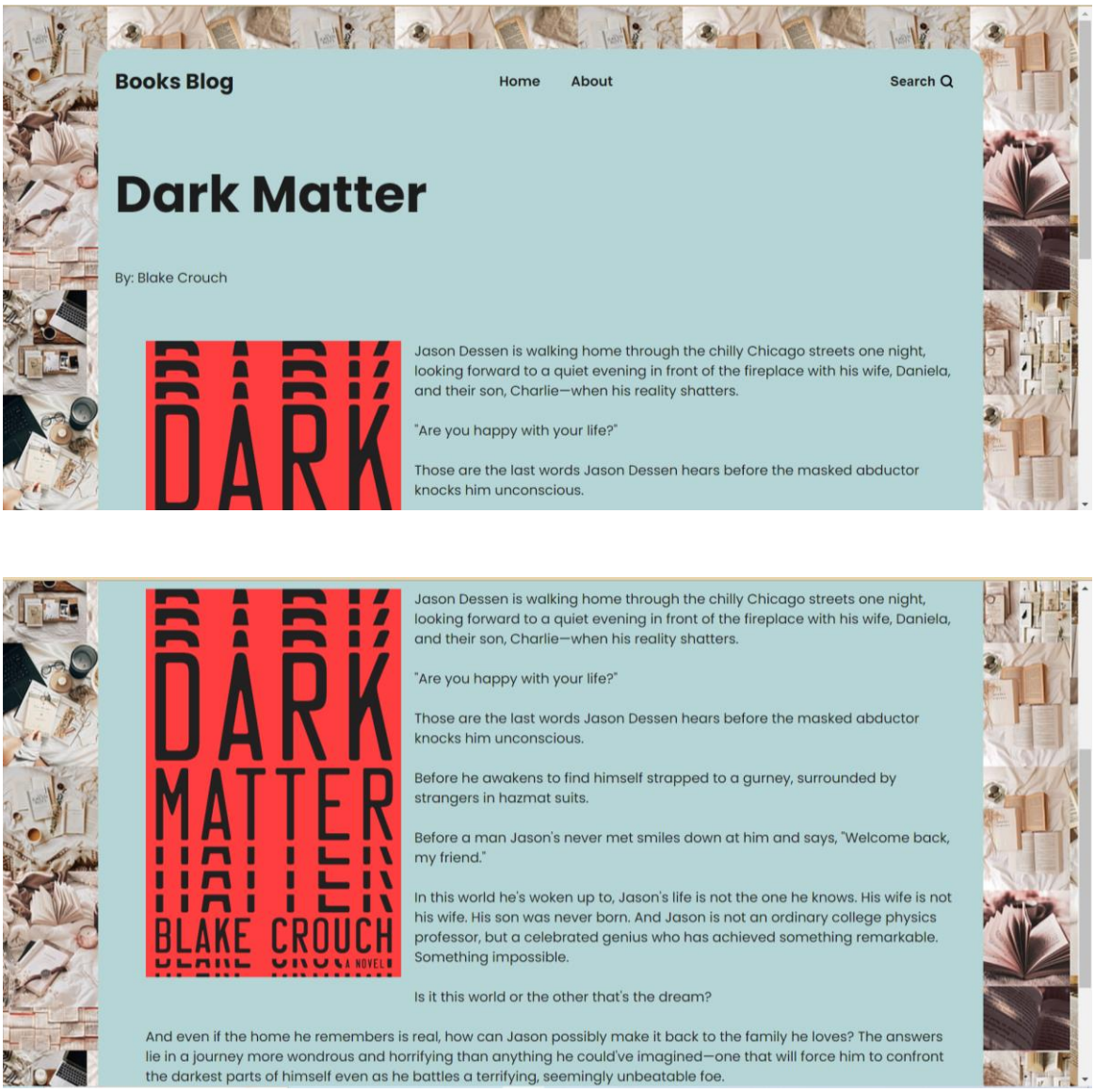

Fig. 4.2 Display of the book review

The Fig. 4.2 displays a book with author name and image and the review written of it.

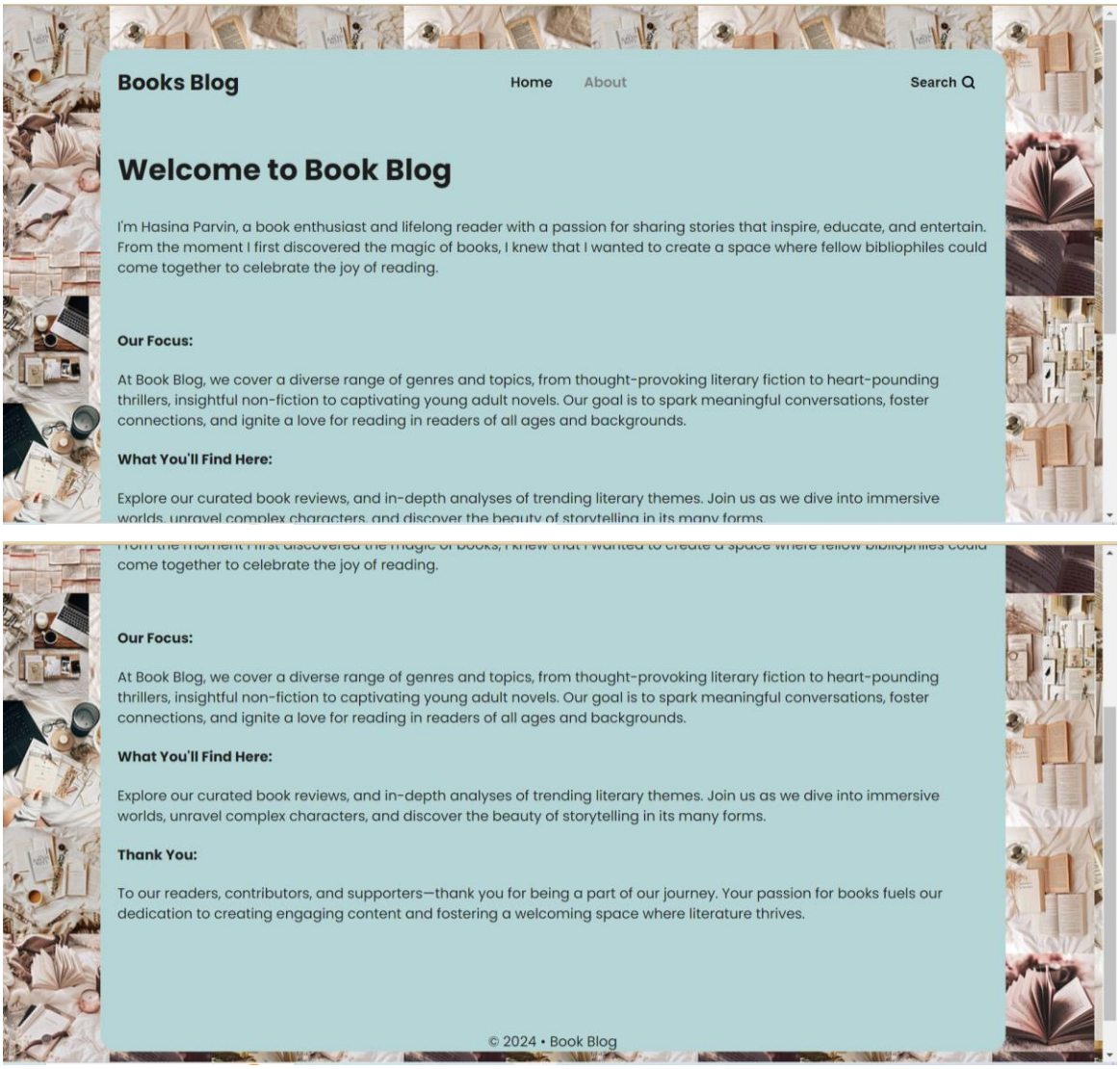

Fig 4.3 About Page

About page contains description about what this website is about as shown in Fig. 4.3.

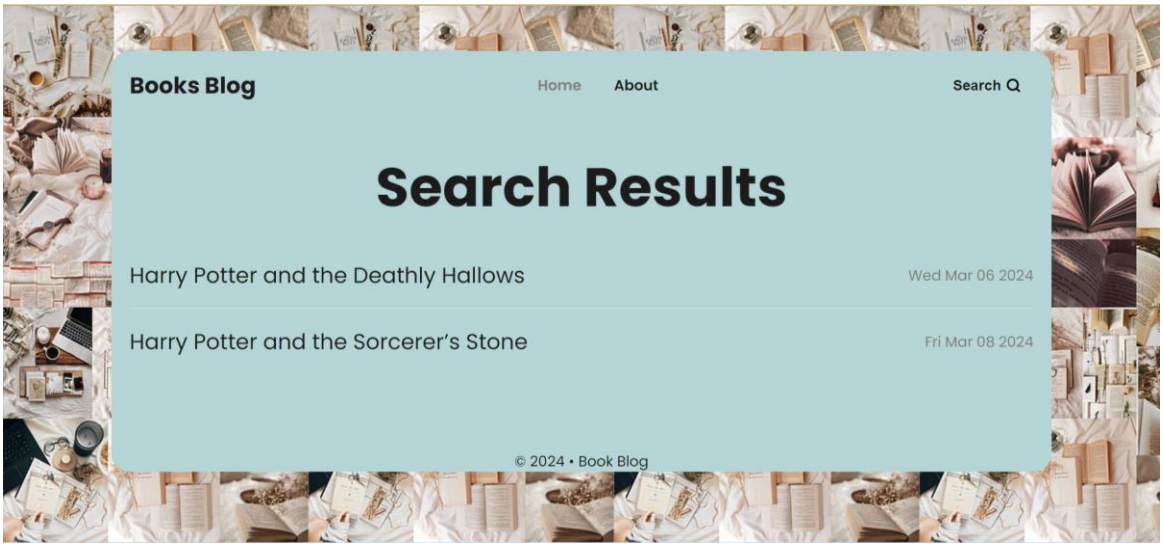

Fig. 4.4 Search

The user can search any book name and display the relevant book if available in the database is shown in Fig. 4.4.

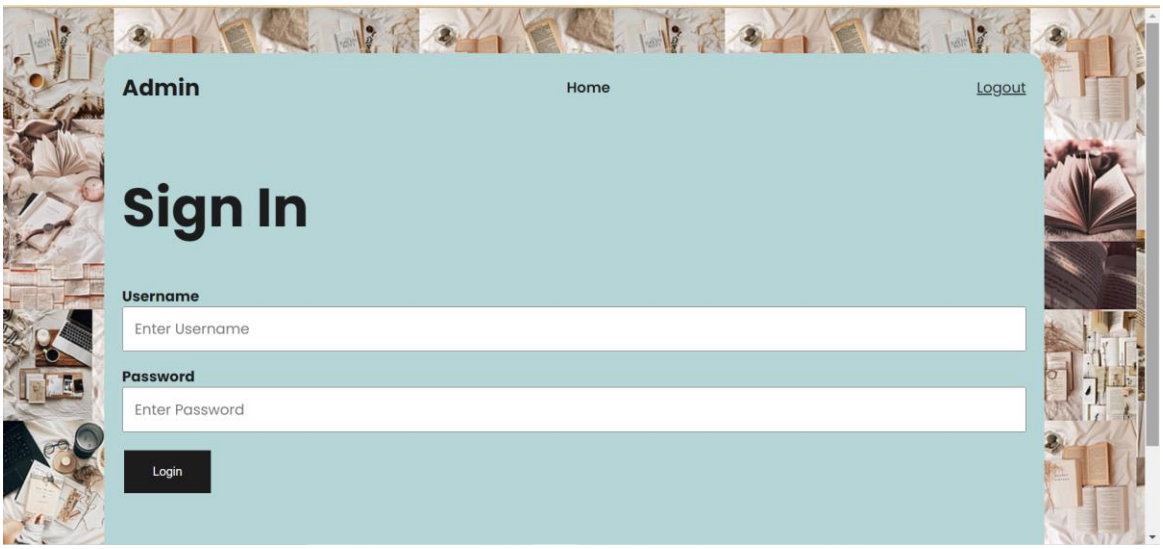

Fig. 4.5 Admin Login

The admin login page as shown in Fig. 4.5 allows only authorized users to login and edit the blog.

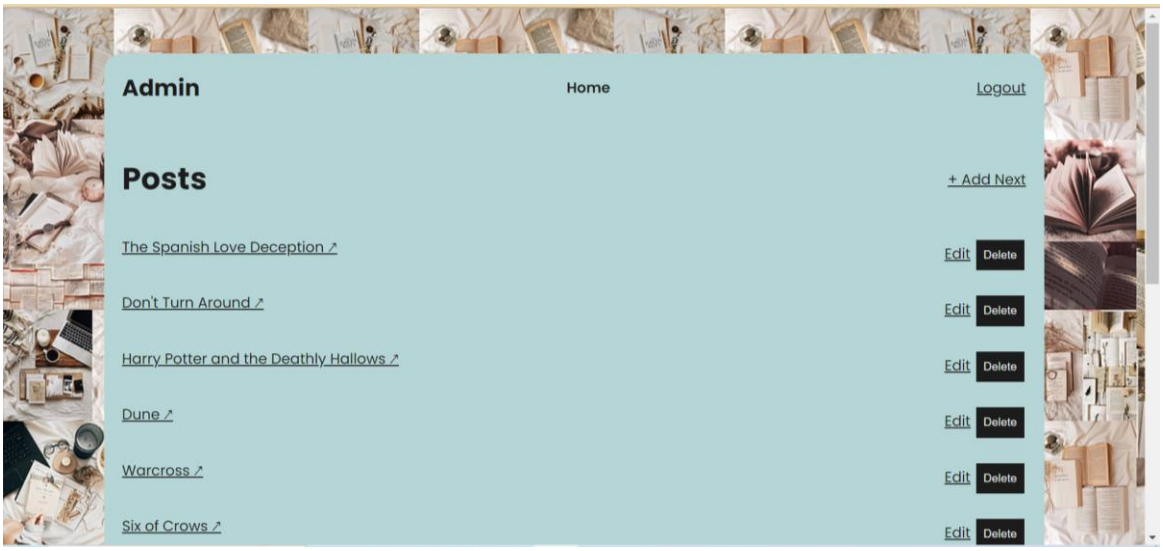

Fig. 4.6 Admin Home Page

The admin can view the reviews, add, edit, and delete review as shown in Fig. 4.6.

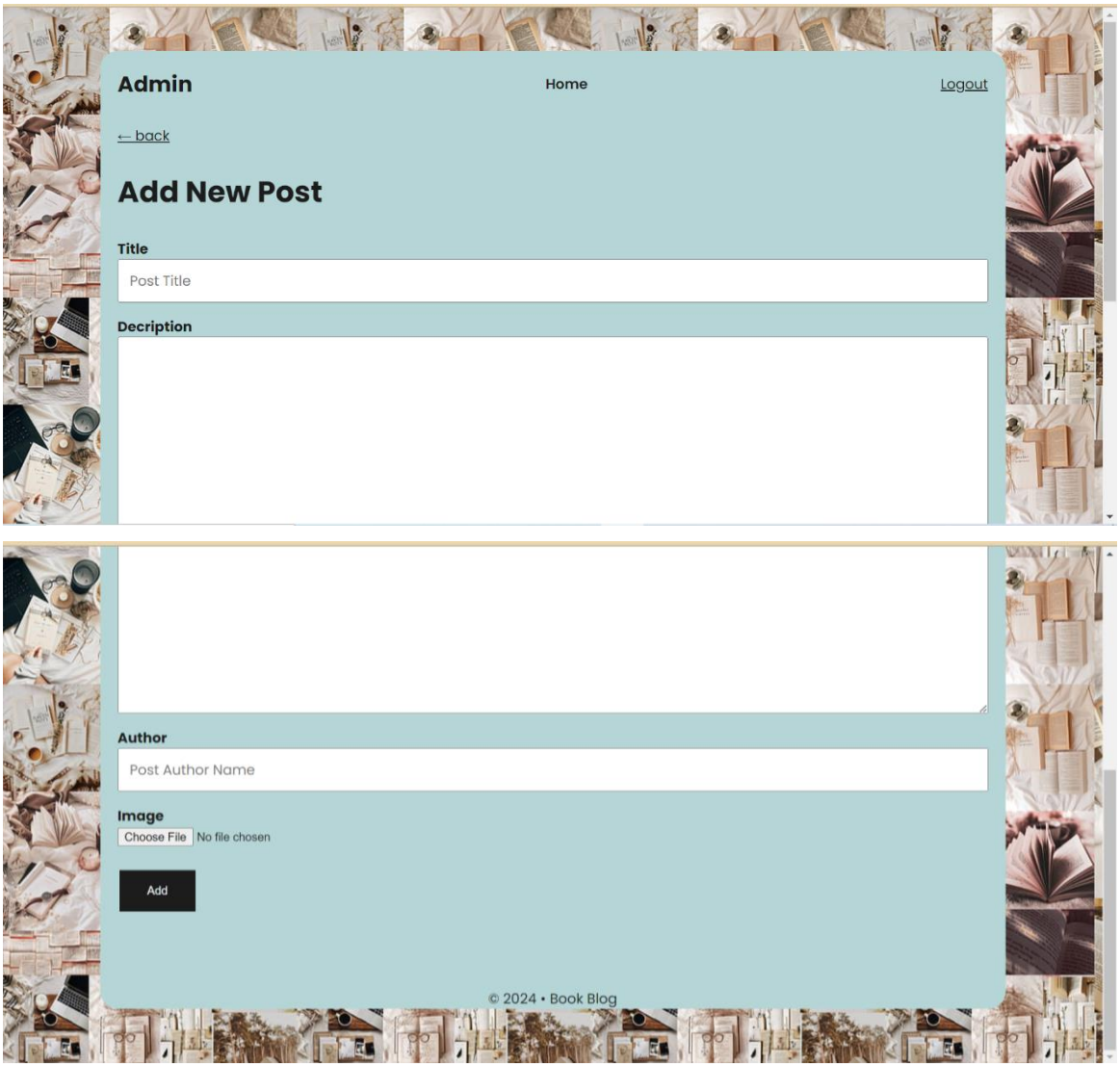

Fig. 4.7 Add New Post

The admin can add a review by filling the form consisting of the book name, author, description and an image of the book as shown in Fig. 4.7.

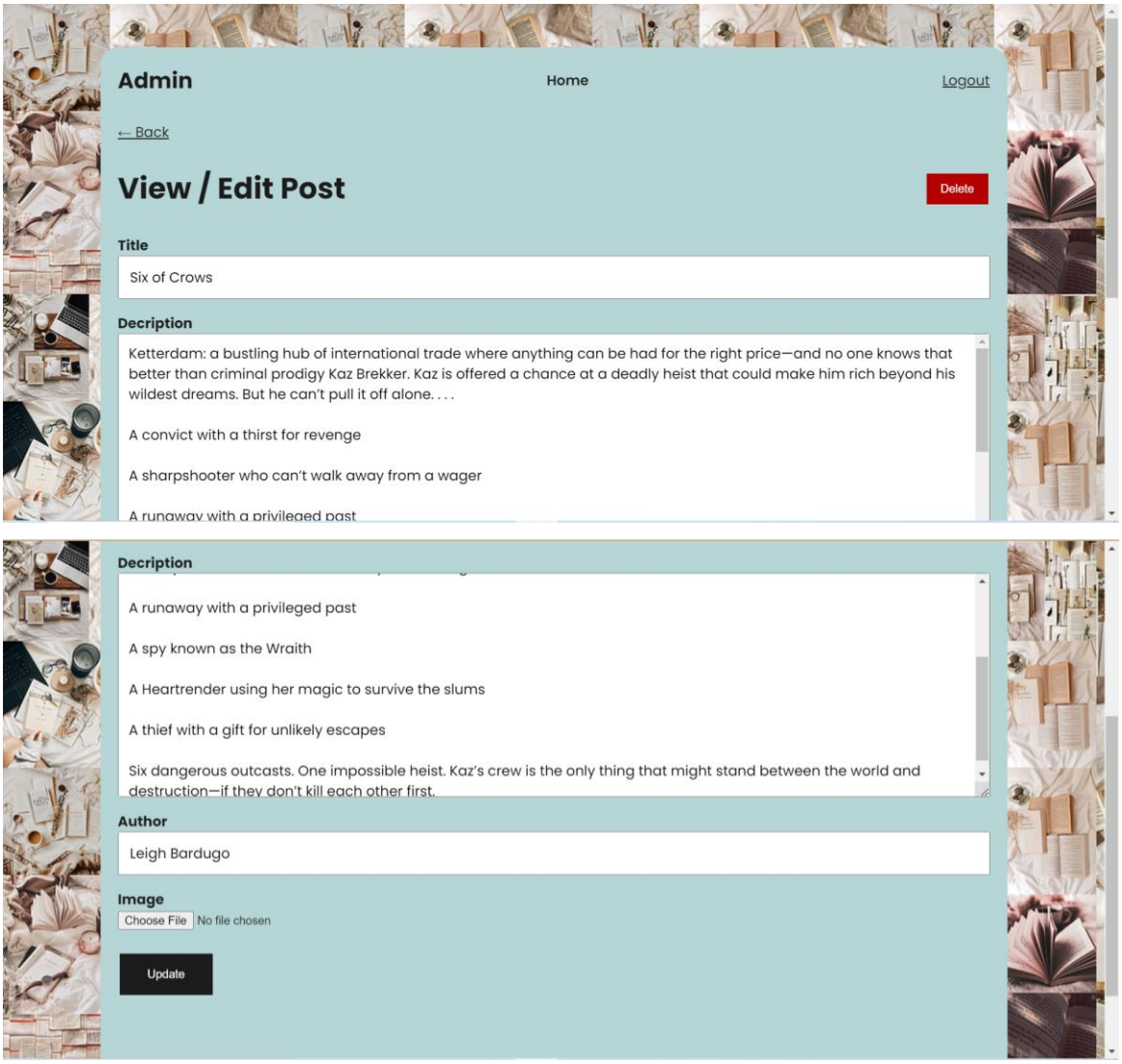

Fig. 4.8 Edit Post

The admin can edit an existing review as shown in Fig. 4.8. The original content already filled, so the admin can make changes in the required field.

Chapter 5

Conclusion

In conclusion, front-end development is an important and rapidly evolving field in web development. Front-end developers play a crucial role in creating the user interface and experience of websites and application. The project allows significant learning and understanding of full stack web development. The Book Blog contains implementation using various languages and framework learnt through the course of internship. It demonstrates the understanding of the frameworks and different languages. One of the project's key accomplishments lies in its user-centric design and functionality. The website allows users to read different reviews on different categories of book. Features such as book reviews, recommendations, and interactive discussions foster a sense of community and engagement, transforming the book blog website into a vibrant hub for literary exploration. The user can surf through various categories or search for a particular book by name. The latest reviews added to the database can also be surfed by the user. Using routes and controllers for proper function of the website. Users can also be displayed by a random book from the database. Moreover, the project has allowed to explore and experiment with a diverse array of technologies, frameworks, and tools in full-stack web development. The skills, experiences, and insights gained throughout the project will serve as a solid foundation for our future efforts. To enhance the functionality and user experience of the Book Blog project, several key improvements can be implemented. Firstly, incorporating reviews for each book listed on the platform. Moreover, dividing the book reviews into categories specific to each book genre or theme can improve content organization and navigation. Enhancements in layout, typography, and overall design aesthetics can elevate the user experience, making it more engaging and enjoyable. Furthermore, integrating links to online stores where users can purchase the books featured in specific reviews adds practical value to the platform.

References

Textbook Reference

[1] James Webb, *Web Development And Design For Beginners: Learn And Apply The Basic Of* HTML5*, CSS3, JavaScript, Jquery, Bootstrap, DOM, Unix Command, And Github - Tools for building responsive websites,* Alberta Inc. , 2020.

[2] Bruno Joseph D'mello, Mithun Satheesh, and Jason Krol, *Web Development with MongoDB and Node*, Packt Publishing Ltd, Third Edition, 2017.

Website Reference

[1] https://www.mongodb.com/languages/full-stack-development.

[2] https://www.simplilearn.com/what-is-full-stack-development-article.