MangaVerse: A Comic Website

Project Documentation

Submitted by:
Michael George E. Aquino
James Adrian B. Castro
Rafael Iñigo T. Lopez
Alvin G. Pableo
Oscar Cloud Vincent C. Yumang

Course: BS Computer Science
Subject: Data Structures and Algorithms
Institution: National University – Philippines
Date: October 2025

I: Introduction

MangaVerse began as a small comic website designed to provide readers with an insight into modern Japanese comics, known as manga. Manga is very diverse and really packed with good literature. Manga is really popular nowadays due to its known animation counterpart, which is the anime. Anime adaptation gives life to any Manga or Light Novel. It gives readers an aspect of excitement when they are picking up a manga or reading it online. This website is designed to cater to such cravings to read comics and bring life to any kind of Japanese literature to their mobile phones.

Background:

Recent decades have seen the consumption of digital media rise substantially, particularly as a result of online reading environments such as web novels, comics, and manga. Manga, a Japanese art form that weaves compelling storytelling and expressive images, has attracted a huge amount of global popularity because of its simplicity and ubiquity in a variety of genres. But, many available platforms do not have many useful and personalizing features, especially for web-based systems intended for students or casual readers, but they are growing in popularity.

MangaVerse is a site designed to be well-organized, user-friendly, and educational, providing users with the space and information to easily access, search, organize, and track manga reading habits. In the most current manga site, content delivery is most common, but data management, user interaction, and backend integration are important themes that are most important to understanding modern web systems.

Within the context of academia, this project has served as a bridge between web development theory and practical applications and integrated database management, user authentication, and algorithmic design into a cohesive system. Also highlighted here is how data structures and algorithms, such as arrays, stacks, and sorting, are essential to improving efficiency, organization, and overall system performance. The project was thus created both as entertainment and as a tool to show how professional-grade web systems operate using modular code architecture and full-stack implementation.

Purpose:
MangaVerse is a full-stack web application that forms a digital manga library. I hope to address common problems found in the online manga browsing community: disorganized collections, lack of personalization, lack of customer feedback mechanisms, and a system that allows users to create accounts, log in securely, browse the manga titles, post reviews, and keep track of their reading history by using this system.

The purpose of this project is to demonstrate the practical application of computer science concepts, including data structures, database design, and sorting algorithms, in real-world development scenarios. On the backend, Node.js and MongoDB provide efficient data processing and secure user authentication. The frontend utilizes HTML, CSS, and JavaScript to deliver a visually appealing and responsive web interface. Additionally, the backend provides a visually pleasing and responsive frontend for HTML, CSS, and JavaScript.

MangaVerse serves both educational and practical purposes: it facilitates users' access to a virtual manga library, while providing a foundation for students and developers to learn how different technologies (frontend, backend, database) work together to create a coherent and secure online system.

II. Objectives

General Objective:

To design and develop a full-stack web application that serves as an online manga library for readers.

Specific Objectives:

- Implement a backend using Node.js and Express.js connected to MongoDB.
- Create an interactive frontend with HTML, CSS, and JavaScript.
- Allow users to register, log in, and manage their reading activity.
- Integrate a sorting feature using both MongoDB sorting and custom frontend sorting algorithms (Bubble Sort, Quick Sort, Selection Sort).
- Enable users to leave reviews on manga titles.

III: System Overview

System Concept:

MangaVerse is a web-based application that provides users with access to a digital manga library. It enables account creation, manga browsing, and review submission.

Users:

Regular Users – can create an account, browse manga, view details, and post reviews.

System Requirements:

Hardware:

- Minimum 4GB RAM
- Dual-core processor

Software:

- Node.js v18+ (much better if the latest version)
- MongoDB
- Any modern browser (Chrome, Edge, Firefox)
- NetBeans IDE, VS Code (for development and testing)

IV. Data Structures Used

Data Structure	Purpose	Justification
Array	Stores the list of manga retrieved from the backend.	Arrays allow easy iteration, sorting, and filtering.
Object	Represents individual manga, user data, and reviews.	Provides flexibility for JSON-based API data.
Stack (via History Feature)	Tracks recently read manga.	Simulates last-read-first-display behavior.
Queue	Manages asynchronous fetch tasks.	Ensures smooth, ordered data retrieval.

V. Algorithm Design

System Algorithms Overview

The backend of MangaVerse applies a combination of data handling, authentication, and data organization algorithms. While it doesn't include traditional sorting algorithms like Bubble Sort directly in the backend, it utilizes MongoDB's query-based sorting, authentication verification, and history tracking algorithms through structured logic.

1. **Authentication Algorithm (Login & Registration):**

Pseudocode:

```lua
function registerUser(username, email, password):
    hashedPassword = bcrypt.hash(password)
    newUser = { username, email, password: hashedPassword }
    save newUser to MongoDB
    generate JWT token for session

function loginUser(email, password):
    user = findUserByEmail(email)
    if user exists and bcrypt.compare(password, user.password):
        generate JWT token
        return success
    else:
        return error ("Invalid credentials")
```

Explanation:
This ensures secure access control by hashing user passwords before storage and verifying them during login. The use of JWT tokens allows authenticated communication between the frontend and backend.

2. **Manga Sorting Algorithm:**

When displaying manga collections, the system sorts manga titles alphabetically (A–Z or Z–A) depending on user selection.
This is achieved through MongoDB's built-in .sort() method.

```js
Manga.find({ title: { $regex: search, $options: 'i' } })
     .sort({ title: sort === 'asc' ? 1 : -1 });
```

Type:	Algorithm	Description:
Indexed Sort	B-Tree Index Traversal	When a field (e.g, title) is indexed, MongoDB leverages its B-tree index structure to read data in sorted order. This method is extremely efficient, with logarithmic complexity O(log n), and does not require an additional in-memory sort.
Non-Indexed Sort	In-Memory Merge Sort (Top-K Merge Sort)	When no index exists, MongoDB uses an optimized merge sort variant internally to arrange results. This algorithm is stable, ensuring consistent order for identical fields, and is designed to handle large datasets by merging sorted batches in memory.

3. **Reading History Algorithm**

Pseudocode:

```
pgsql

function addHistory(userId, mangaTitle, chaptersRead):
    user = findById(userId)
    user.history.push({ mangaTitle, chaptersRead, date: now })
    save user
```

Explanation:
When a user reads a manga, the system logs it with the manga title, chapters read, and timestamp. This acts like a stack (most recent entries appear first), tracking reading progression efficiently.

4. **Review Submission Algorithm**

Pseudocode:

```
css

function addReview(mangaId, username, reviewText):
    manga = findById(mangaId)
    manga.reviews.push({ username, reviewText, date: now })
    save manga
```

Explanation:
Reviews are appended directly to a manga's document. This approach simplifies retrieval and avoids separate review collections, allowing quick user-to-manga association.

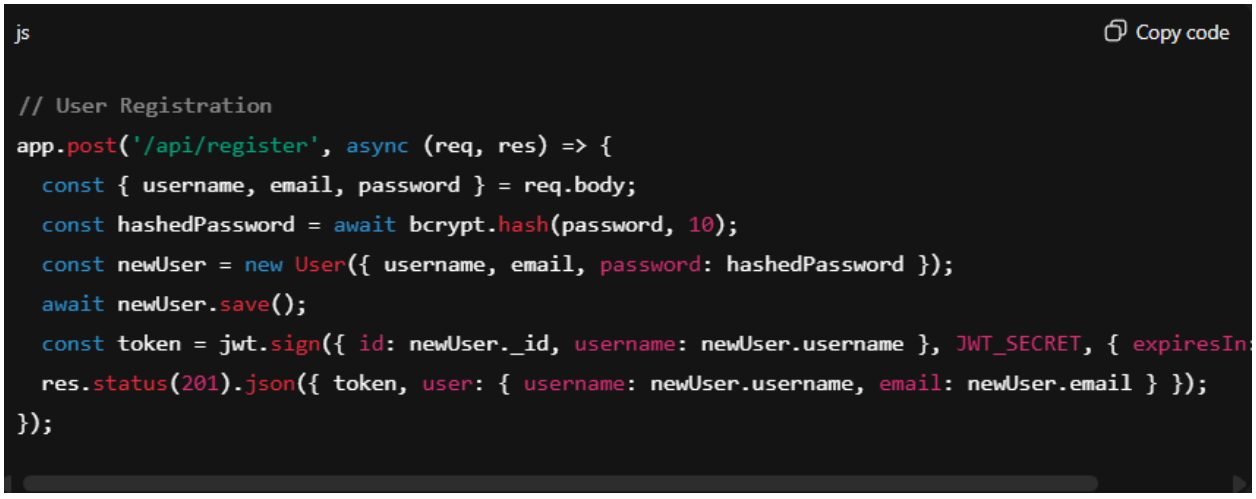
VI. Implementation

Programming Language and Tools

- Backend: Node.js (Express.js framework)
- Database: MongoDB (via Mongoose ORM)
- Authentication: JSON Web Token (JWT)
- Security: bcrypt.js for password hashing
- Configuration: dotenv for environment variable management
- CORS: For safe API communication between front-end and back-end

VI.1:

Key Code Snippet (Authentication):



```
gistration
/api/register', async (req, res) => {
username, email, password } = req.body;
shedPassword = await bcrypt.hash(password, 10);
wUser = new User({ username, email, password: hashedPassword });
wUser.save();
ken = jwt.sign({ id: newUser._id, username: newUser.username }, JWT_SECRET, { expiresIn: '1d' });
us(201).json({ token, user: { username: newUser.username, email: newUser.email } });
```

VI.2
Key Code Snippet (Reading History):

```js
// Update History
app.post('/api/history', verifyToken, async (req, res) => {
  const { mangaTitle, chaptersRead } = req.body;
  const user = await User.findById(req.user.id);
  user.history.push({ mangaTitle, chaptersRead, date: new Date() });
  await user.save();
  res.status(201).json(user.history);
});
```

VI.3
Key Code Snippet (Sorting Manga):

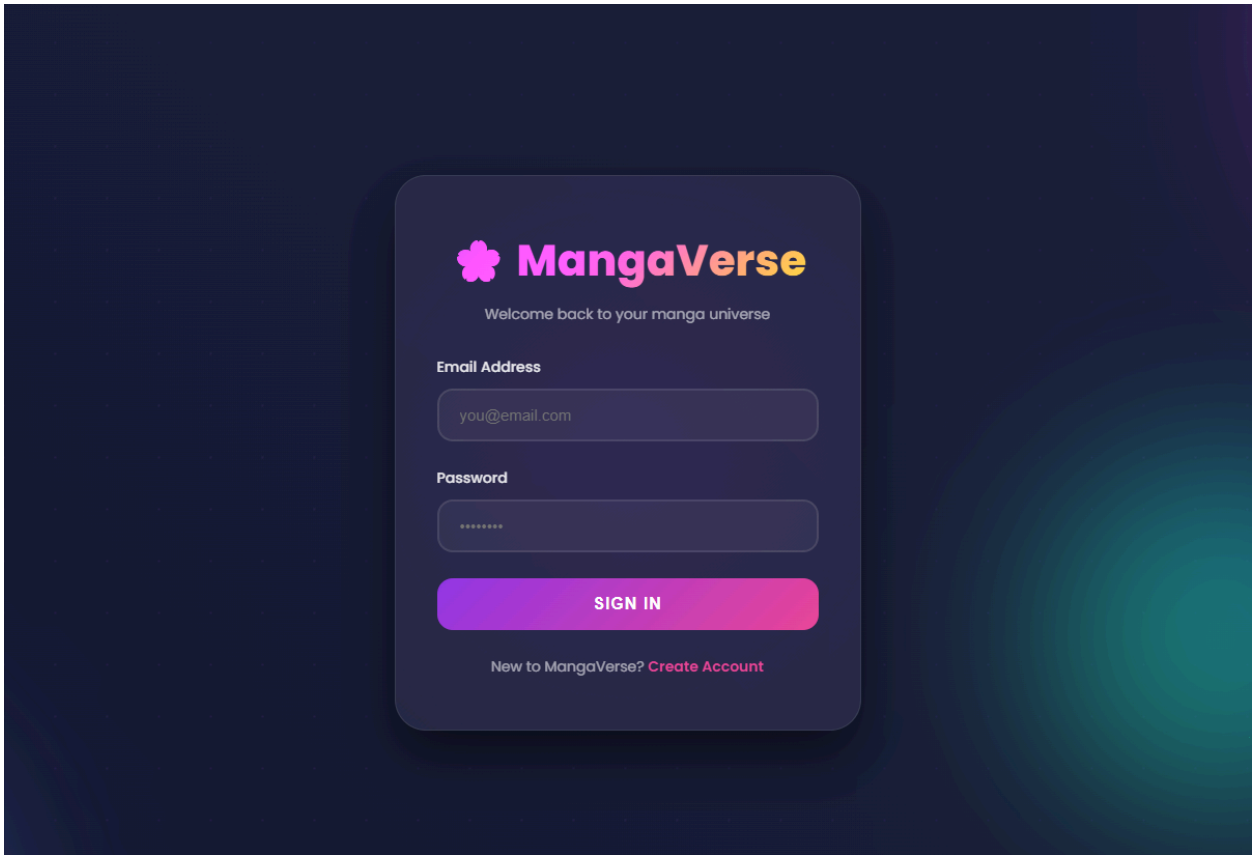
```js
app.get('/api/mangas', verifyToken, async (req, res) => {
  const { sort = 'asc', search = '' } = req.query;
  const mangas = await Manga.find({ title: { $regex: search, $options: 'i' } })
                            .sort({ title: sort === 'asc' ? 1 : -1 });
  res.json(mangas);
});
```

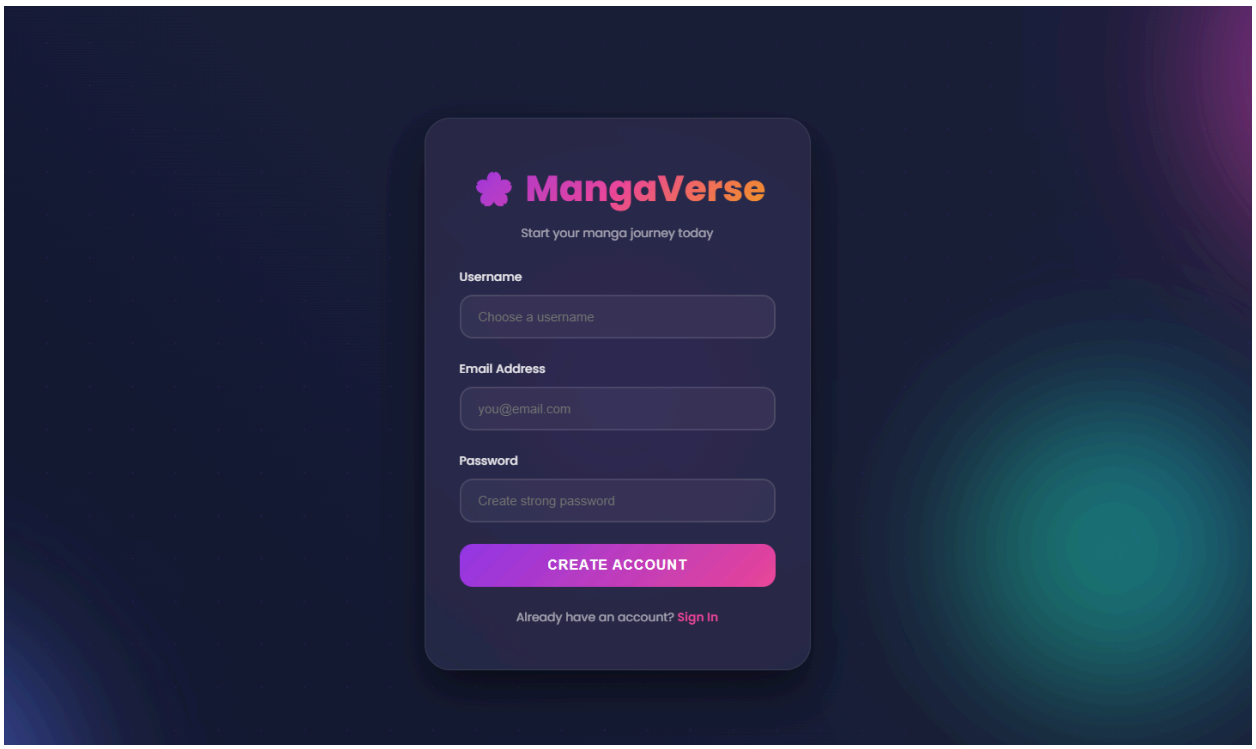
VII. Testing and Results

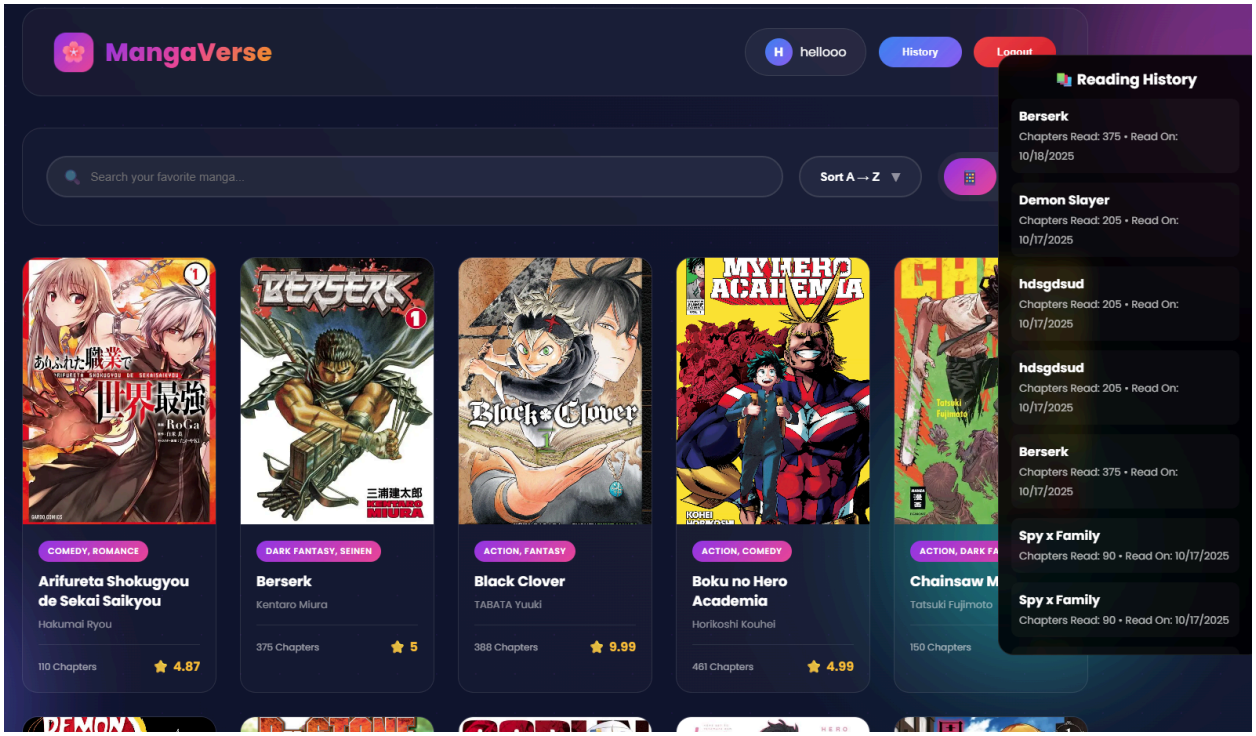
VII Table I. Test Table

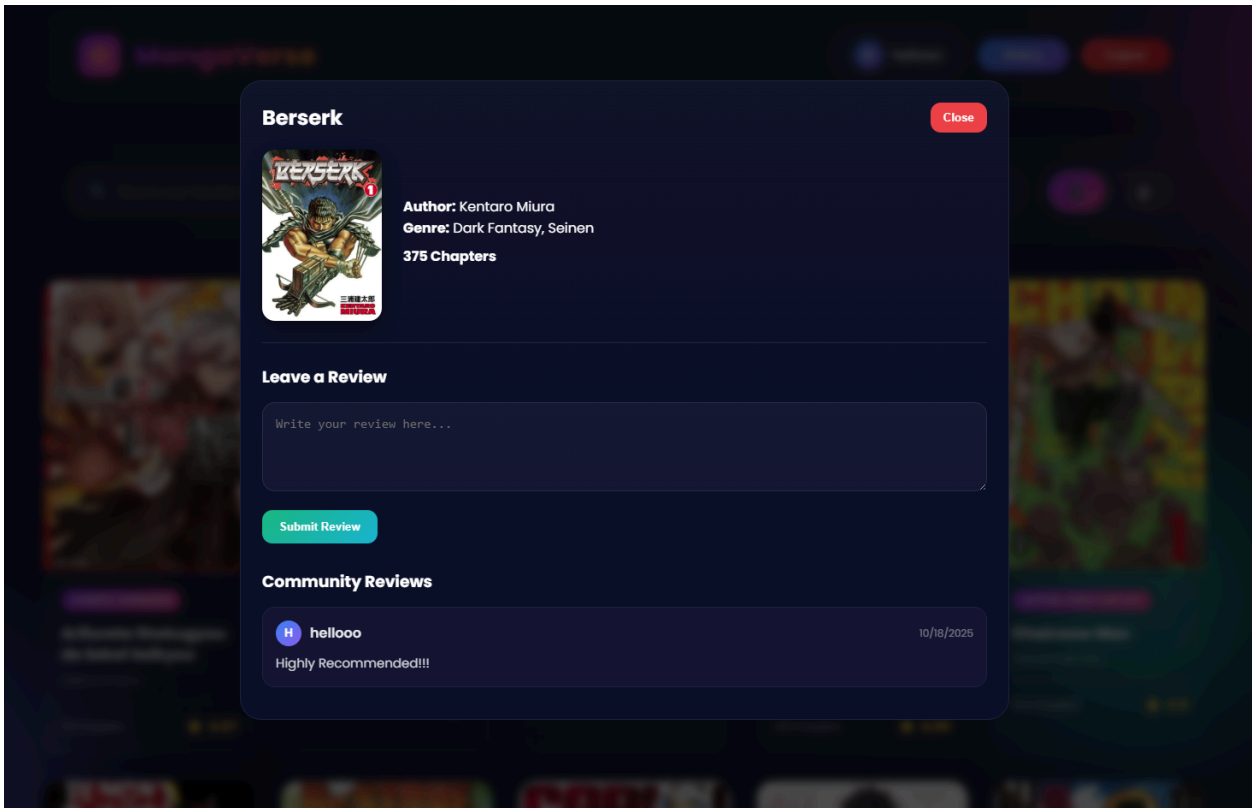
Test Case	Input	Expected Output	Actual Output	Result
Register User	username: “Leo”, email: “leo@test.com” , password: “12345”	Token returned & user saved	Success	✅
Login User	Valid email & password	JWT Token returned	Success	✅
Load Manga	/api/mangas?sort=asc	Sorted manga list	Sorted correctly	✅
Add Review	Valid manga ID & review text	Review added successfully	Works correctly	✅
Add History	mangaTitle: “Bleach”, chaptersRead: 5	History updated	Works correctly	✅
Load History	GET /api/history	List of read manga	Displays properly	✅
Invalid Token	Missing JWT	Access denied	Proper error message	✅

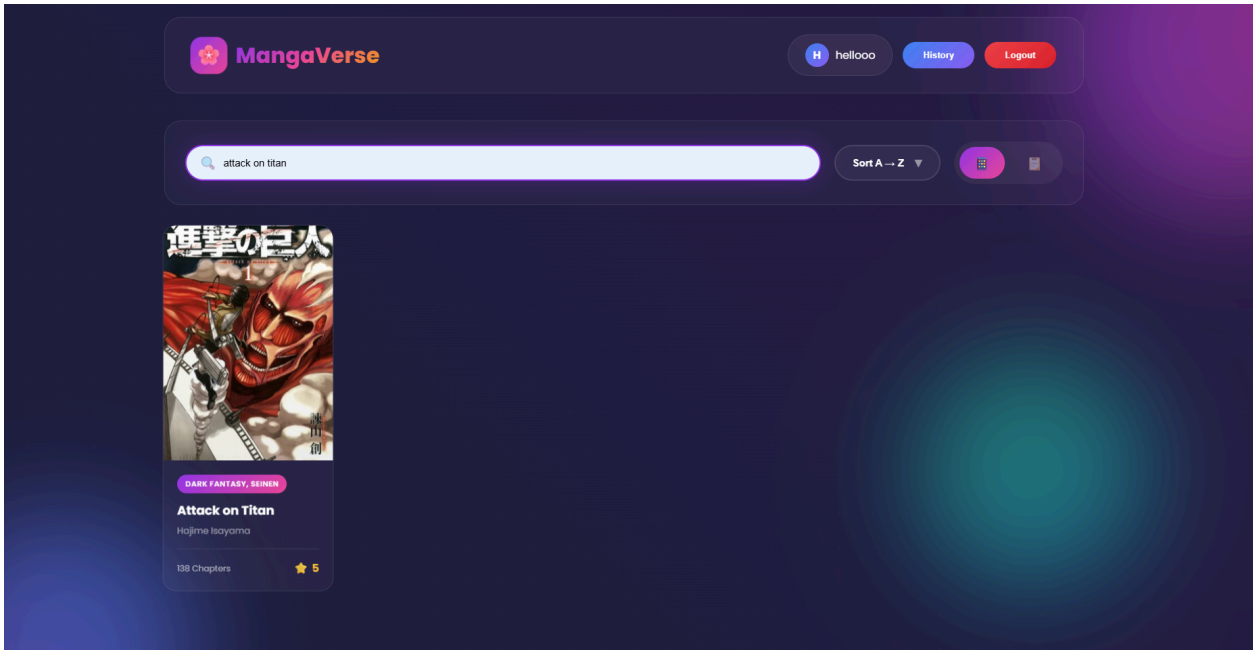
VII. Test Results: (Screenshots of output)

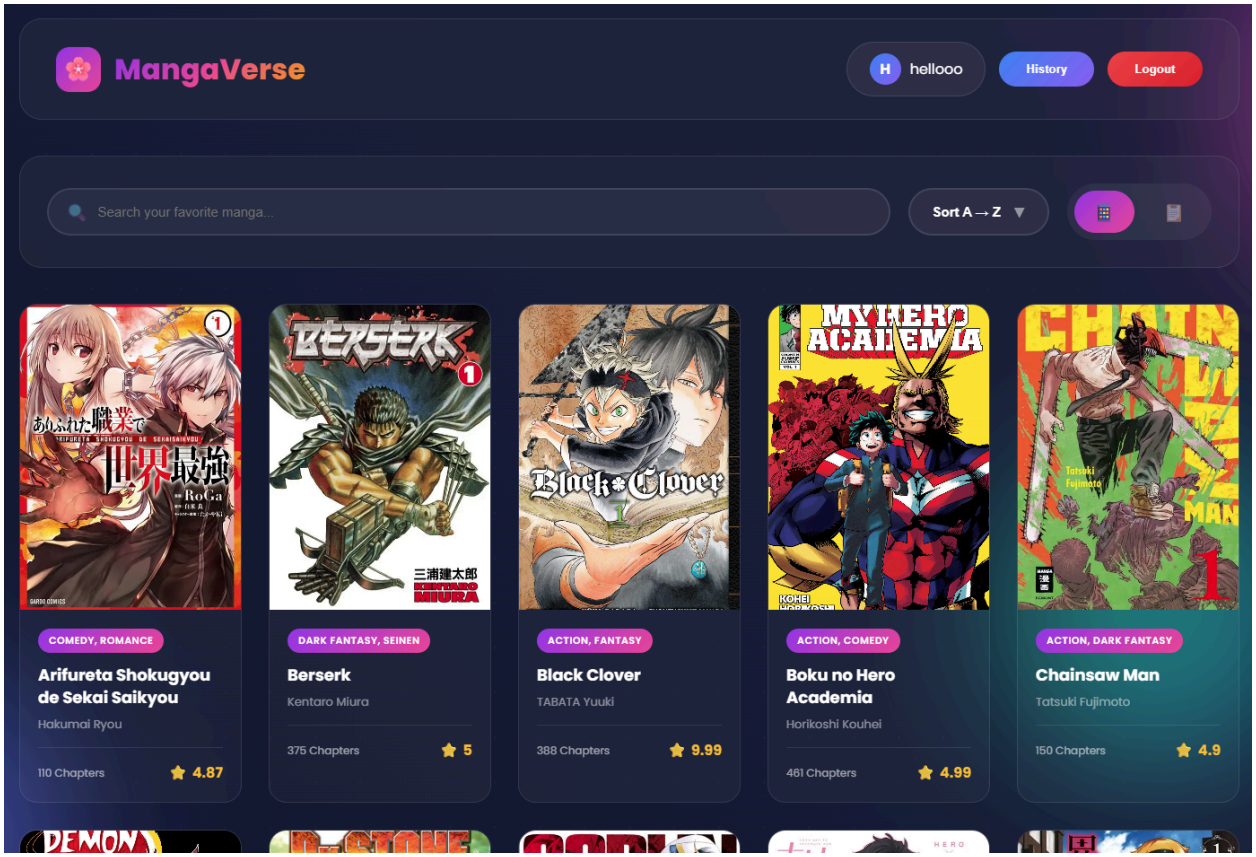





MangaVerse
H
hellooo
History
Logout
Reading History
Berserk
Chapters Read: 375 • Read On: 10/18/2025
Demon Slayer
Chapters Read: 205 • Read On: 10/17/2025
hdsgdsud
Chapters Read: 205 • Read On: 10/17/2025
hdsgdsud
Chapters Read: 205 • Read On: 10/17/2025
Berserk
Chapters Read: 375 • Read On: 10/17/2025
Spy x Family
Chapters Read: 90 • Read On: 10/17/2025
Spy x Family
Chapters Read: 90 • Read On: 10/17/2025
Search your favorite manga...
Sort A → Z
COMEDY, ROMANCE
Arifureta Shokugyou de Sekai Saikyou
Hakumai Ryou
110 Chapters
4.87
DARK FANTASY, SEINEN
Berserk
Kentaro Miura
375 Chapters
5
ACTION, FANTASY
Black Clover
TABATA Yuuki
388 Chapters
9.99
ACTION, COMEDY
Boku no Hero Academia
Horikoshi Kouhei
461 Chapters
4.99
150 Chapters


Berserk
Close
Author: Kentaro Miura
Genre: Dark Fantasy, Seinen
375 Chapters
Leave a Review
Write your review here...
Submit Review
Community Reviews
H
hellooo
10/18/2025
Highly Recommended!!!


MangaVerse
H
hellooo
History
Logout
attack on titan
Sort A → Z
DARK FANTASY, SEINEN
Attack on Titan
Hajime Isayama
138 Chapters
5


MangaVerse
H
hellooo
History
Logout
Search your favorite manga...
Sort A → Z
COMEDY, ROMANCE
Arifureta Shokugyou de Sekai Saikyou
Hakumai Ryou
110 Chapters
4.87
DARK FANTASY, SEINEN
Berserk
Kentaro Miura
375 Chapters
5
ACTION, FANTASY
Black Clover
TABATA Yuuki
388 Chapters
9.99
ACTION, COMEDY
Boku no Hero Academia
Horikoshi Kouhei
461 Chapters
4.99
ACTION, DARK FANTASY
Chainsaw Man
Tatsuki Fujimoto
150 Chapters
4.9

VIII. Conclusion and Recommendation

Conclusion:

The MangaVerse system demonstrated both the potential of the front-end and back-end technologies to create a dynamic, interactive, and user-friendly manga browsing platform. The front-end delivers visually striking effects and animation, animation, and responsive design by using HTML, CSS, and JavaScript. Both Node.js and Express.js back-end make accessing data effortless; MongoDB is a flexible, powerful NoSQL database for users, manga, reviews, and reading histories; and MongoDB is flexible and efficient, providing users, manga data, reviews, and reading histories with high data accuracy.

The greatest success of this project is the security built into the JWT system for authentication on JWT, which safeguards users' sessions in a way that reflects the simplicity of the interface. It also combines MongoDB's powerful sorting algorithms, B-tree index traversal, and in-memory merge sort to display manga results in ascending or descending order. It is the combination of speed and scale that allows a powerful application to manage multiple users, large datasets without an increase in performance loss.

In addition, users can spend more time engaging and capturing information by including user reviews and reading history features, and thereby further enhance the experience. These features mimic the personalized experience of a real-world manga platform.

The project shows how a small team can design and implement a fully functioning full-stack web application that connects design, functionality, and performance. This document also highlights the importance of API integration, database management, and synchronous communication between client and server in modern web applications.

Recommendations:

While MangaVerse has achieved its core goals, several areas can be improved for improved functionality and scalability.

- Add Manga Upload and Management System: Add an admin panel to allow authorized users to upload, edit, or remove manga entries dynamically without needing a static seeding solution.

- Integration and infinite scroll: In addition to pagination and infinite scroll, to increase performance, a possible scroll or infinite scroll can be added to the pagination and infinite scroll settings to speed up the loading of manga entries in large datasets.

- Improve Security: Add password hashing to the system using bcrypt and rate limiting to improve data protection.

- Involve cloud storage: Store manga cover images and data on a cloud service like AWS S3, Cloudinary, or Firebase Storage, and be able to scale and access all the data for global access.

To run a mobile version of MangaVerse on mobile devices with React Native or Flutter to provide a cross-platform reading experience.

Recommendation System: Use a simple AI or machine learning model that suggests manga based on user preferences or reading history to suggest manga for user-defined preferences.

- Improve Review System: Add the ability to submit or comment on reviews and to display the average rating in each manga.

- Use cloud MongoDB (Atlas) for remote deployment and multi-user access.

- Incorporate real-time features (e.g., live reviews, reader statistics) using WebSockets.

In conclusion, MangaVerse successfully lays the groundwork for an advanced, user-centric manga platform that merges aesthetic design with powerful functionality. The project not only serves as a strong example of practical web development but also opens pathways for future enhancement using modern technologies such as machine learning, cloud computing, and progressive web app (PWA) integration.

IX. References

1. GeeksforGeeks. (n.d.). Sorting Algorithms in JavaScript. Retrieved from https://www.geeksforgeeks.org/sorting-algorithms/

2. MongoDB Documentation. (n.d.). Sorting Query Results. Retrieved from https://www.mongodb.com/docs/manual/reference/method/cursor.sort/

3. W3Schools. (n.d.). Node.js Tutorial. Retrieved from https://www.w3schools.com/nodejs/

4. MangaHere. (n.d.). MangaHere Official Website. Retrieved from https://www.mangahere.cc/

5. Mozilla Developer Network (MDN). (n.d.). JavaScript Array Methods. Retrieved from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

6. FreeCodeCamp. (2021). Understanding MongoDB for Beginners. Retrieved from https://www.freecodecamp.org/news/learn-mongodb-a4ce205e7739/

7. CSS-Tricks. (n.d.). Creating Animated Backgrounds in CSS. Retrieved from https://css-tricks.com/

8. Stack Overflow. (n.d.). Implementing Sorting Algorithms in JavaScript. Retrieved from https://stackoverflow.com/

9. Tutorialspoint. (n.d.). Express.js Overview. Retrieved from https://www.tutorialspoint.com/expressjs/index.htm