VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY
THE INTERNATIONAL UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

SayST – A minimal social platform

By
Le Cao Nhat Hoang – ITITIU20205
Truong Tan Phat – ITITIU20140

A thesis submitted to the School of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Bachelor of Information Technology/Computer Science/Computer Engineering

Ho Chi Minh City, Vietnam
Year

SayST – A minimal social platform

APPROVED BY:

_____ ,
Nguyen Van A, Ph.D, Chair (*Example*)
*(*Type Committee names beneath lines*)

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

_____
(*Typed Committee name here*)

THESIS COMMITTEE
(Whichever applies)

TABLE OF CONTENTS

ABSTRACT

This report details the development of a minimalist social media platform built using Next.js, inspired by the core functionalities of Thread. The project prioritizes a lightweight and performant user experience, focusing on essential features like creating and viewing posts, commenting on existing posts, and liking content.

This report outlines the project's architecture, key functionalities, and implementation choices made using Next.js. It will discuss the benefits of utilizing Next.js for this type of application and explore potential areas for future development to expand the platform's features.

CHAPTER 1

INTRODUCTION

1.1. Background

Social media platforms have become an integral part of modern communication. They offer individuals and communities a space to connect, share ideas, and engage in discussions. However, these platforms can often become overwhelming due to their extensive feature sets and complex interfaces. This project aims to address this by creating a lightweight and focused social media experience.

Traditionally, social media platforms have been built using server-side rendered frameworks. While this approach offers flexibility, it can sometimes lead to slower page load times and a less optimal user experience, especially on mobile devices.

1.2. Problem Statement

The current landscape of social media platforms often suffers from:

Feature Overload: Platforms can be cluttered with a multitude of features, leading to a confusing and overwhelming user experience. This can be particularly detrimental for new users or those seeking a simpler way to connect.

Performance Issues: Traditional server-side rendered applications can struggle with scalability and performance, especially on mobile devices. This can lead to slow loading times and a negative user experience.

Lack of Focus: Many platforms prioritize user engagement over meaningful interaction. Algorithms can create echo chambers and encourage passive consumption rather than active participation.

This project seeks to address these issues by creating a social media platform with a focus on:

- Simplicity: Prioritizing essential features like creating and viewing posts, commenting, and liking content.
- Performance: Utilizing Next.js for server-side rendering and static site generation to ensure fast loading times and a smooth user experience across all devices.
- Meaningful Interaction: Encouraging active participation and fostering connection through focused and purposeful interactions.

1.3. Scope and Objectives

Scope:

SayST is designed as a minimalist social platform, concentrating on essential social media functionalities to provide a clean and efficient user experience. The scope of SayST includes:

Public Posts: Users can create and view public posts, ensuring that all content is easily accessible without the need for private profiles or complex privacy settings.

Comments: Users can comment on posts, facilitating simple and straightforward interactions.

Likes: Users can like posts, enabling quick and easy content engagement.

By focusing on these core features, SayST aims to eliminate the complexity often found in larger social media platforms, thereby enhancing usability and performance.

Objectives:

The primary objectives of SayST are:

Simplicity: To provide a user-friendly interface that prioritizes ease of use, making it accessible to a broad audience.

Performance: To leverage Next.js for server-side rendering and static site generation, ensuring fast load times and a responsive user experience.

Scalability: To build a platform that can efficiently handle a growing user base without compromising on performance or user experience.

Customization: To lay the groundwork for future customization options, allowing users to personalize their profiles and post formatting while maintaining the platform's minimalist ethos.

Engagement: To foster user interaction through straightforward functionalities like viewing, liking, and commenting on posts, thereby creating a vibrant and active user community.

By adhering to these objectives, SayST aims to deliver a streamlined, high-performance social media platform that meets the needs of users seeking simplicity and efficiency in their online interactions.

1.4. Assumption and Solution

Assumptions:

User Base: It is assumed that the target user base prefers a minimalist social media platform over feature-rich alternatives. Users are looking for a straightforward way to share and engage with content without the need for private messaging or complex privacy settings.

Technology Stack: The use of Next.js for server-side rendering and static site generation is assumed to significantly enhance the platform's performance and scalability, making it suitable for handling a large number of users.

Engagement Patterns: It is assumed that core functionalities such as viewing, liking, and commenting on posts will be sufficient to foster user engagement and community interaction.

Future Customization: It is assumed that offering customization options in the future will enhance user satisfaction and retention without complicating the user experience.

Resource Availability: Adequate resources (time, expertise, and technology) are assumed to be available for the development and ongoing maintenance of the platform.

Solutions:

Minimalist Feature Set:

- By focusing on essential features like public posts, comments, and likes, SayST addresses the assumption that users prefer simplicity. This keeps the interface clean and the user experience intuitive.

Next.js Implementation:

- Leveraging Next.js for server-side rendering and static site generation ensures fast load times and responsive interactions, validating the assumption that the technology stack will enhance performance and scalability.

User Engagement:

- Implementing a streamlined commenting and liking system caters to the assumption that these features will be sufficient for user engagement. This approach encourages straightforward and frequent interactions among users.

Future Enhancements:

- Building the platform with the potential for future customization aligns with the assumption that personalized features will be valuable. This ensures that SayST can evolve according to user needs without compromising its minimalist design.

Efficient Resource Use:

- By planning for a lean development process and leveraging existing expertise in Next.js, SayST can effectively use available resources to achieve its goals, addressing the assumption of resource availability.

CHAPTER 2

LITURATURE REVIEW/RELATED WORK

2.1. Thread

This project draws inspiration from Thread [1], a minimalist social media application that emphasizes core functionalities such as creating and viewing posts, commenting on existing posts, and liking content. Thread prioritizes a clean interface and a smooth user experience, which aligns with the goals of this project, SayST.

However, there are key differences between SayST and Thread:

Feature Scope: While Thread includes features like direct messaging and private profiles, SayST focuses solely on public posts, comments, and likes to maintain a simpler and faster user experience.

Scalability: SayST leverages Next.js for server-side rendering and static site generation, aiming for better scalability to accommodate a potentially larger user base compared to Thread.

Customization: SayST plans to explore additional customization options beyond Thread's basic functionality, potentially allowing users to personalize their profiles or post formatting in the future.

By comparing and contrasting with Thread, SayST aims to demonstrate the value of a focused social media experience while exploring how Next.js can enhance performance and scalability for a similar application.

CHAPTER 3

METHODOLOGY

3.1. Overview

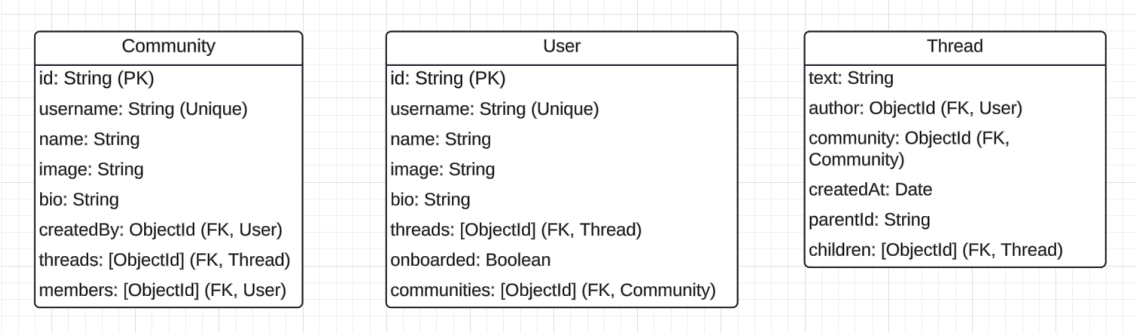
3.2. User requirement analysis

The user requirement analysis for SayST involves understanding the needs and expectations of potential users to ensure that the platform meets their demands effectively. This analysis is critical in defining the functionalities and design principles that guide the development of a user-centric social media application.

3.3. System Design

3.3.1. Datadase design

- Entities and Attributes:
    - Community:
        id: String (Primary Key)
        username: String (Unique)
        name: String
        image: String
        bio: String
        createdBy: ObjectId (References User)
        threads: [ObjectId] (References Thread)
        members: [ObjectId] (References User)
    - Thread:
        text: String
        author: ObjectId (References User)
        community: ObjectId (References Community)
        createdAt: Date
        parentId: String
        children: [ObjectId] (References Thread)
    - User:
        id: String (Primary Key)
        username: String (Unique)
        name: String
        image: String
        bio: String
        threads: [ObjectId] (References Thread)
        onboarded: Boolean
        communities: [ObjectId] (References Community)



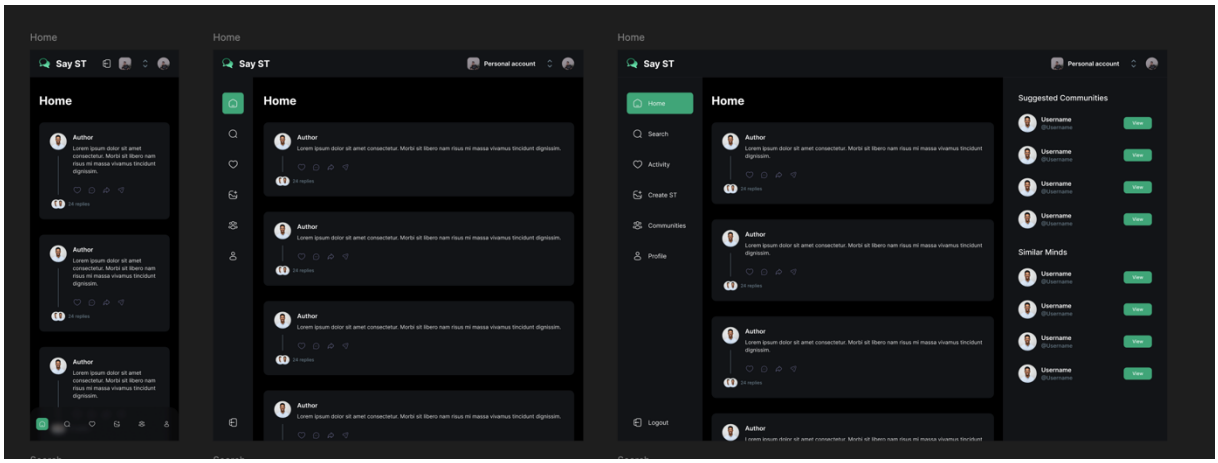
3.3.2. User Interface design

We using Figma to build our User Interface(UI) of the entire application:

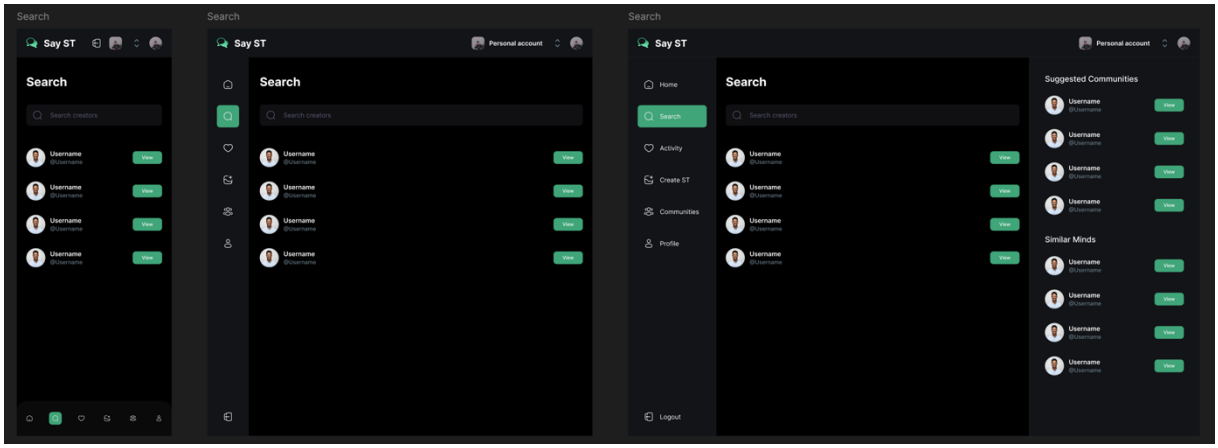
- Each screen has 3 layouts for a common web app shoud have: Web – Tablet – Mobile Repsonsive
- Manage Color and Typography using Styles feature in Figma
- Manage common UI by turn them into Components like we already did in Coding

Figma Link: https://www.figma.com/design/DPvpAnTiffOxajfo8TI0rH/SayST?node-id=0-1&t=XYW0ZVdwSqQStcSq-1

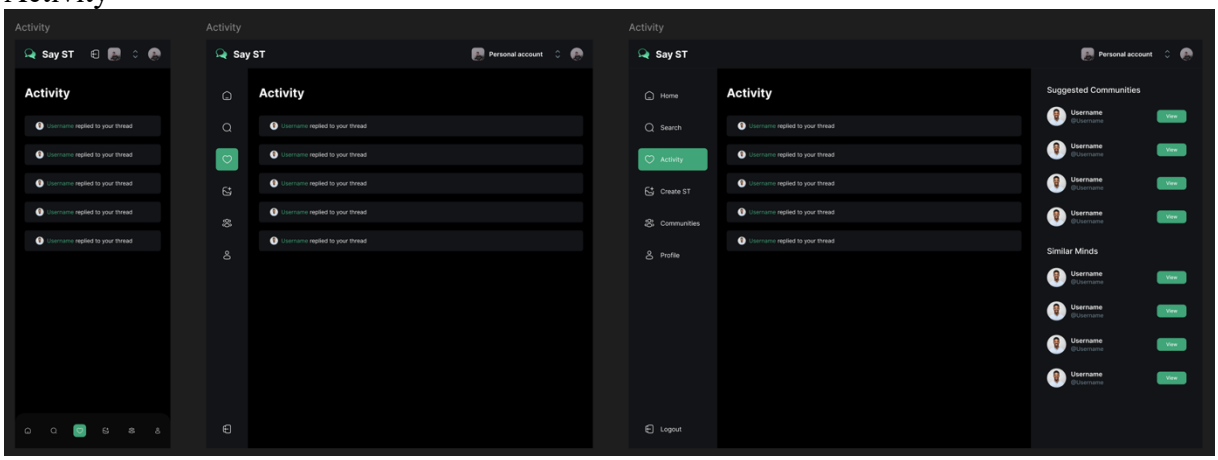
Home:



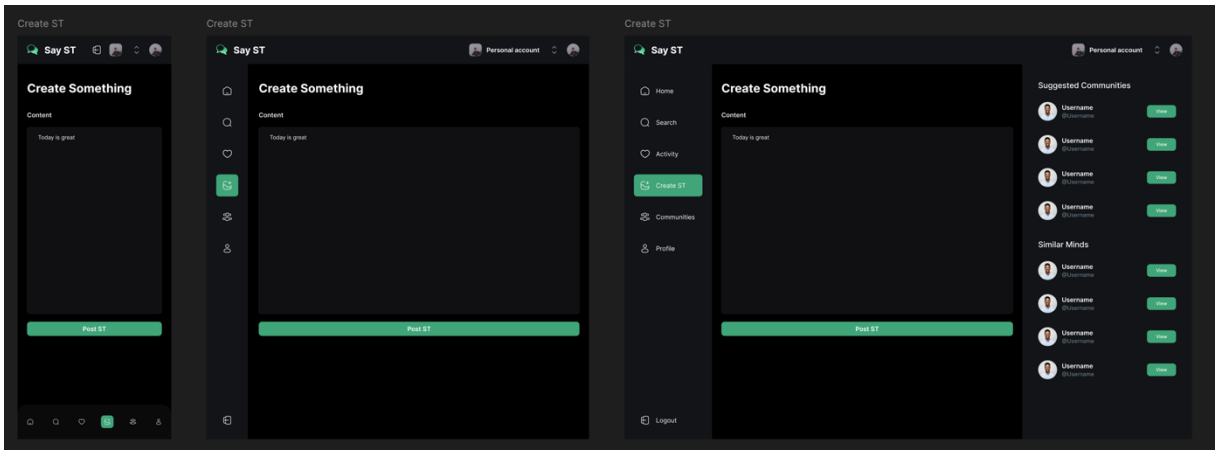
Search



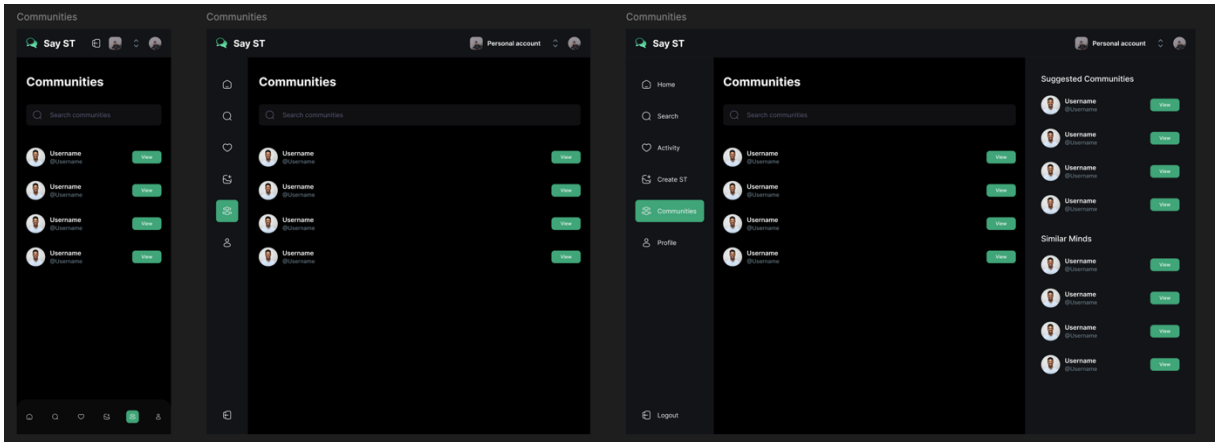
Activity



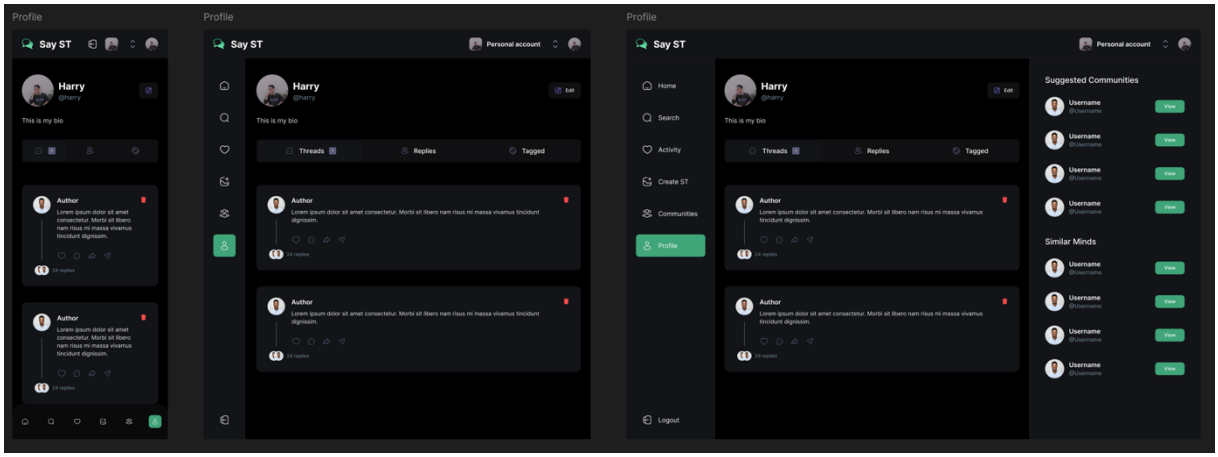
Create Something



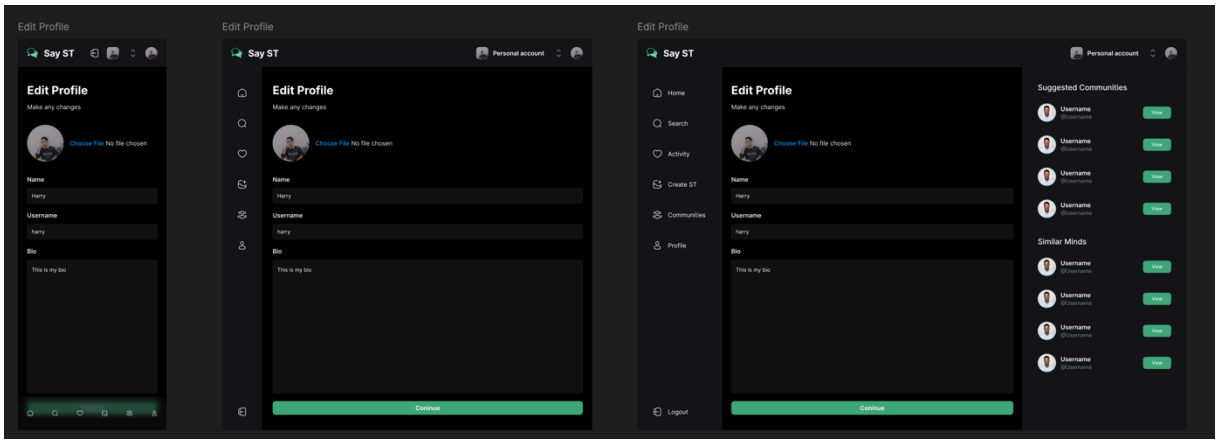
Communities



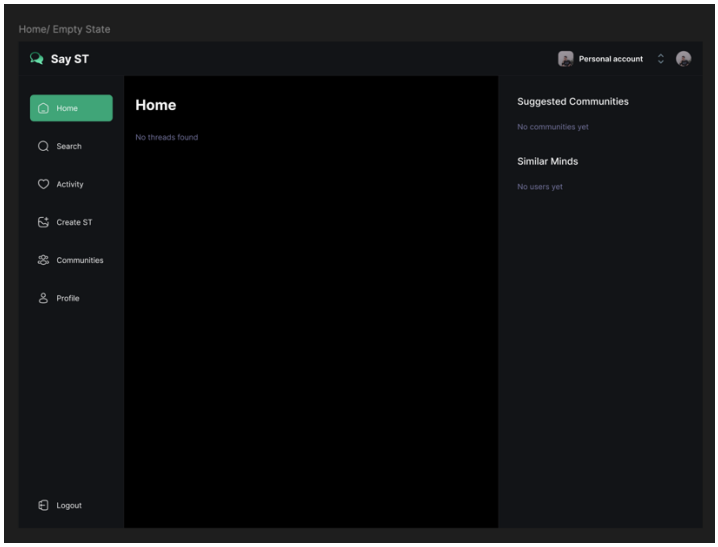
Profile



Edit Profile



Empty State



CHAPTER 4

IMPLEMENT AND RESULTS

4.1. Implement

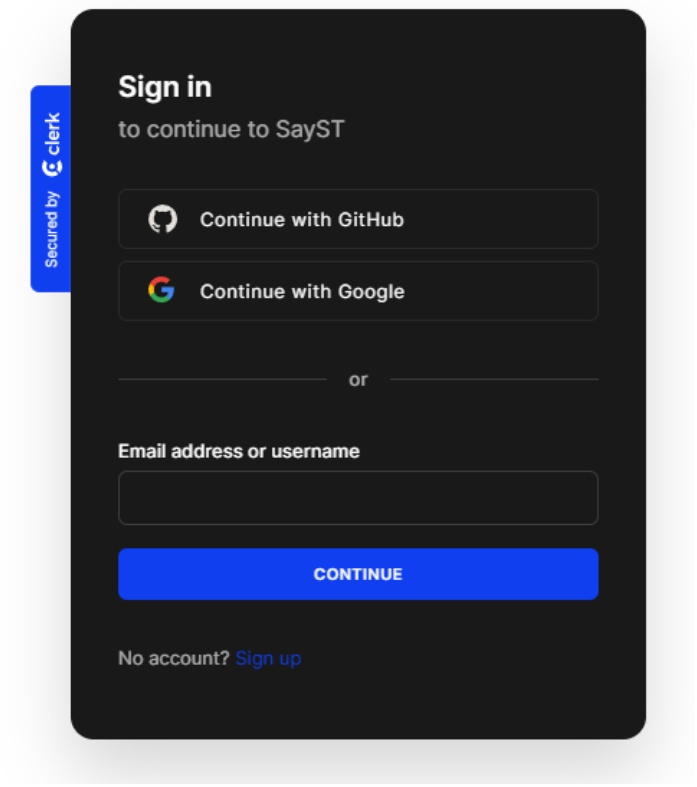
The development of SayST was guided by a focus on simplicity, performance, and scalability. The architecture was designed to handle a growing user base, with Next.js enabling efficient server-side rendering and static site generation to maintain performance as the platform scales.

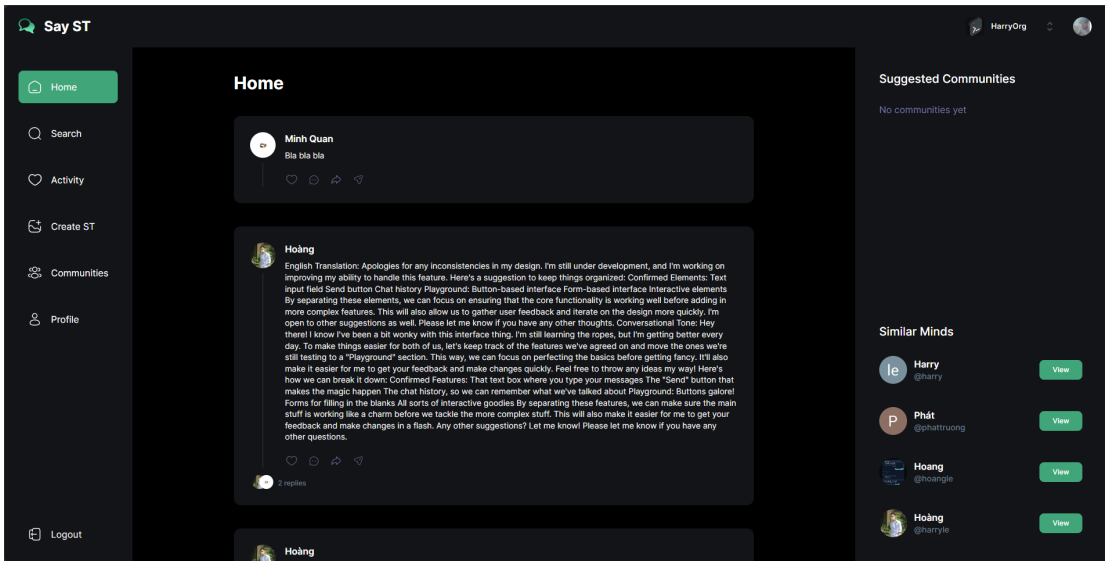
4.2. Results

The platform's core features—public posts, comments, and likes—were implemented with an emphasis on ease of use and efficient functionality.

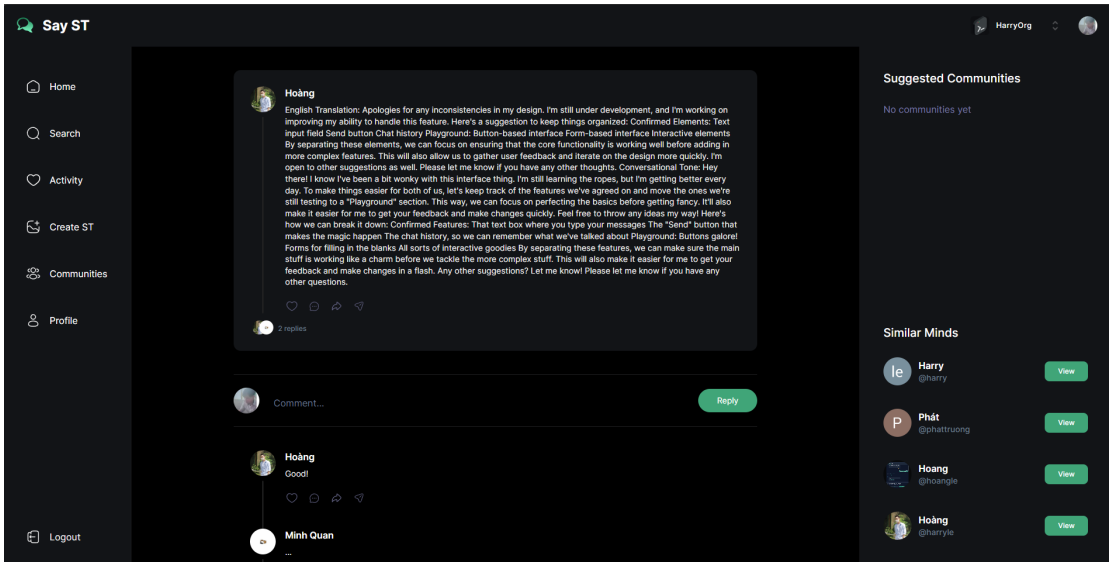
- Public Posts: Users can create, view, and interact with posts without navigating complex privacy settings.
- Comments: A streamlined commenting system was developed to facilitate straightforward user interactions.
- Likes: A simple liking mechanism was integrated to allow quick content engagement.

4.2.1. Result 1




Say ST
HarryOrg
Home
Search
Activity
Create ST
Communities
Profile
Logout
Home
Minh Quan
Bla bla bla
Hoàng
English Translation: Apologies for any inconsistencies in my design. I'm still under development, and I'm working on improving my ability to handle this feature. Here's a suggestion to keep things organized: Confirmed Elements: Text input field Send button Chat history Playground: Button-based interface Form-based interface Interactive elements By separating these elements, we can focus on ensuring that the core functionality is working well before adding in more complex features. This will also allow us to gather user feedback and iterate on the design more quickly. I'm open to other suggestions as well. Please let me know if you have any other thoughts. Conversational Tone: Hey there! I know I've been a bit wonky with this interface thing. I'm still learning the ropes, but I'm getting better every day. To make things easier for both of us, let's keep track of the features we've agreed on and move the ones we're still testing to a "Playground" section. This way, we can focus on perfecting the basics before getting fancy. It'll also make it easier for me to get your feedback and make changes quickly. Feel free to throw any ideas my way! Here's how we can break it down: Confirmed Features: That text box where you type your messages The "Send" button that makes the magic happen The chat history, so we can remember what we've talked about Playground: Buttons galore! Forms for filling in the blanks All sorts of interactive goodies By separating these features, we can make sure the main stuff is working like a charm before we tackle the more complex stuff. This will also make it easier for me to get your feedback and make changes in a flash. Any other suggestions? Let me know! Please let me know if you have any other questions.
2 replies
Hoàng
Suggested Communities
No communities yet
Similar Minds
Harry
@harry
Phát
@phattruong
Hoang
@hoangle
Hoàng
@harryle
View

4.2.2. Result 2


Say ST
HarryOrg
Home
Search
Activity
Create ST
Communities
Profile
Logout
Hoàng
English Translation: Apologies for any inconsistencies in my design. I'm still under development, and I'm working on improving my ability to handle this feature. Here's a suggestion to keep things organized: Confirmed Elements: Text input field Send button Chat history Playground: Button-based interface Form-based interface Interactive elements By separating these elements, we can focus on ensuring that the core functionality is working well before adding in more complex features. This will also allow us to gather user feedback and iterate on the design more quickly. I'm open to other suggestions as well. Please let me know if you have any other thoughts. Conversational Tone: Hey there! I know I've been a bit wonky with this interface thing. I'm still learning the ropes, but I'm getting better every day. To make things easier for both of us, let's keep track of the features we've agreed on and move the ones we're still testing to a "Playground" section. This way, we can focus on perfecting the basics before getting fancy. It'll also make it easier for me to get your feedback and make changes quickly. Feel free to throw any ideas my way! Here's how we can break it down: Confirmed Features: That text box where you type your messages The "Send" button that makes the magic happen The chat history, so we can remember what we've talked about Playground: Buttons galore! Forms for filling in the blanks All sorts of interactive goodies By separating these features, we can make sure the main stuff is working like a charm before we tackle the more complex stuff. This will also make it easier for me to get your feedback and make changes in a flash. Any other suggestions? Let me know! Please let me know if you have any other questions.
2 replies
Comment...
Reply
Hoàng
Good!
Minh Quan
...
Suggested Communities
No communities yet
Similar Minds
Harry
@harry
Phát
@phattruong
Hoang
@hoangle
Hoàng
@harryle
View

CHAPTER 5

DISCUSSION AND EVALUATION

5.1. Discussion

Several specific functionalities were implemented successfully using Next.js. Server-side rendering and static site generation were particularly beneficial, significantly improving performance and scalability, which are critical for maintaining a seamless user experience on SayST. These functionalities ensured faster load times and better handling of a larger user base, aligning perfectly with the project's goals of simplicity and efficiency. Additionally, the integration of Next.js facilitated a streamlined development process, allowing for quick iterations and a robust codebase. The choice of technologies also provided a strong foundation for potential future enhancements, such as user customization options.

However, the development process was not without its difficulties. One significant challenge was managing the state across various components, which sometimes led to unexpected behaviors and required careful debugging. Additionally, while Next.js is powerful, it presented a steep learning curve for developers unfamiliar with its nuances, slowing down initial development stages. Another limitation was the lack of built-in support for some advanced customization features, necessitating additional effort to implement these functionalities. Areas for improvement identified include optimizing the commenting system for better performance and enhancing the user interface to make it more intuitive. These aspects will require further development to fully realize the project's vision.

5.2. Comparison

SayST and Thread both aim to offer a minimalist social media experience but differ significantly in their approach and feature sets. Thread includes functionalities like direct messaging and private profiles, aiming to provide a comprehensive social media platform. In contrast, SayST focuses on simplicity and speed by limiting its features to public posts, comments, and likes. This streamlined approach ensures a faster and more straightforward user experience. Additionally, SayST leverages Next.js for server-side rendering and static site generation, enhancing its scalability and performance for a potentially larger user base. While Thread provides a robust set of features catering to diverse social interactions, SayST prioritizes a clean, efficient interface with potential for future customization options, setting it apart as a focused alternative in the social media landscape.

5.3. Evaluation

SayST, as a minimal social platform focusing on viewing, liking, and commenting on posts, has shown significant promise in delivering a streamlined user experience. The implementation of core functionalities using Next.js has been highly successful, particularly in terms of performance and scalability. Users benefit from fast load times and a responsive interface, which are critical for maintaining engagement on the platform. The simplicity of SayST's feature set ensures that users can easily navigate and interact with the content, fulfilling the project's goal of providing a straightforward social media experience. However, the development process also highlighted some areas for improvement. For instance, the commenting system could be further optimized for better performance, and the user interface, while intuitive, might benefit from additional refinements to enhance usability. Overall, SayST effectively meets its objective of offering a minimalist yet engaging social platform, with clear opportunities for future enhancements to elevate the user experience further.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1. Conclusion

In conclusion, the development of SayST has demonstrated the potential of creating a minimalist social media platform focused on core functionalities such as viewing, liking, and commenting on posts. By leveraging Next.js for server-side rendering and static site generation, the project successfully achieved its objectives of simplicity, performance, and scalability. Through careful implementation and iterative development, SayST has established a clean and efficient user experience. While challenges were encountered, particularly in state management and advanced customization features, these areas also present opportunities for future enhancements. Overall, SayST has laid a solid foundation for a focused social media experience, with a clear path for continued growth and improvement.

6.2. Future work

- Light/Dark mode
- Optimize performance

REFERENCES

1. Thread. "Thread: A Minimalist Social Media Application." Accessed June 11, 2024. https://www.threadapp.com.

APPENDIX