**ABSTRACT**

Callify is built using a modern tech stack that enhances its functionality and user experience. The primary framework utilized is **Next.js**, a powerful React-based framework that enables server-side rendering and static site generation, ensuring fast load times and improved SEO. This framework allows developers to create dynamic web applications with ease, making it ideal for a video conferencing platform. To style the application, **Tailwind CSS** is employed, providing a utility-first approach to CSS. This allows for rapid design and customization, enabling developers to create responsive and visually appealing interfaces without the need for extensive custom CSS. Tailwind's flexibility ensures that the UI is not only functional but also aesthetically pleasing, enhancing user engagement.

For user authentication, **Clerk** is integrated into Callify. Clerk simplifies the authentication process by offering features such as social sign-in, multi-factor authentication, and magic links. This ensures a secure and user-friendly experience, allowing users to sign up and log in effortlessly while maintaining high security standards.Lastly, **Get Stream** is utilized for real-time communication features, including video conferencing and chat functionalities. Get Stream's robust API allows for seamless integration of real-time messaging and video capabilities, ensuring that users can communicate effectively during meetings. With its high uptime and reliability, Get Stream enhances the overall performance of Callify, making it a dependable choice for virtual collaboration. Together, these technologies create a powerful and efficient platform that meets the demands of modern remote communication, providing users with a seamless and engaging experience.

CONTENTS

**Sl. No. Chapter Name Page No.**

**i. Abstract i**

**ii. Contents ii**

**iii. List of figures iv**

**iv. List of tables v**

1. **Chapter 1 - Introduction 1**
   1. Definition 1
   2. Technology Stack 2
2. **Chapter 2 – Literature Survey 3**
   1. Definition 3
3. **Chapter 3 – Requirement Specifications 6**
   1. Functional Requirements 6
   2. Non- Functional Requirements 8
   3. Software Requirements 11
   4. Hardware Requirements 13
4. **Chapter 4 – System Design 15**
   1. Definition 15
   2. Overview of Callify 15
   3. Key Features 16
   4. Technology Stack 17
   5. User Interface Design 17
   6. Security Measures 18
   7. Development and Deployment 18
   8. Future Enhancements 18
5. **Chapter 5- Project Implementation 19**
   1. Definition 19
   2. Project Setup 19
   3. Environment Configuration 19
   4. User Authentication with Clerk 20
   5. Integrating Get Stream for Video Calls 20
   6. Building The User Interface 21
   7. Implementing Meeting Functionality 22
   8. Styling with Tailwind CSS 22
   9. Deployment 23
6. **Chapter 6- Testing 24**
   1. Definition 24
   2. Testing Goals and User Needs: 24
   3. Testing Environment and Equipment: 24
   4. Testing Process: 25
   5. Assumptions and Limitations: 25
   6. Key Findings 25
7. **Chapter 7 – Results 26**
   1. Definition 26
   2. Key Features and Benefits 26
   3. Applications Across Industries 26
   4. Challenges and Considerations 27
   5. Future Trends in Video Conferencing 27
8. **Chapter 8- Conclusion & Future scope 28**
   1. Conclusion 28
   2. Future Scope 28

**References 30**

**LIST OF FIGURES**

**Figure. No Figure Name Page. No**

**Figure 1.2.1**  Next Js 02

**Figure 1.2.2** Tailwind CSS 02

**Figure 1.2.3.**  Clerk 02

**Figure 1.2.4.**  Get Stream 02

**Figure 4.2.1**  Dataflow Diagram 15

**Figure 4.3.1**  ER Diagram 16

**Figure 5.4.1** Clerk Authentication 20

**Figure 5.5.1** Get Stream Chat 21

**Figure 5.6.1.** Create Link Model 22

**Figure 5.7.1.** Create and Joining Model 22

**Figure 5.7.2.** Joining Call 22

**LIST OF TABLES**

**Table.No Table Name Page. No**

**Table 2.1.1 : Literature Review**  03-05