

MINI PROJECT REPORT

On

“TALKIFY – A Chat WebApp”

Submitted by

Piyush Sharma

(191500547)

Anmol Pandey

(191500126)

Shashank Jain

(191500748)

Shantnu Maheshwari

(191500745)

Ankit

(191500119)

Department of Computer Engineering & Applications
Institute of Engineering & Technology



GLA University
Mathura- 281406, INDIA 2021-2022



**Department of computer Engineering and
Applications GLA University, Mathura**
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,

Declaration

We hereby declare that the work which is being presented in the MINI Project “TALKIFY – A Chat WebApp”, in partial fulfillment of the requirements for MINI Project viva voce, is an authentic record of our own work carried by the team members under the supervision of our mentor Mr. Mayank Saxena.

Group Members: Shashank Jain	(191500748)
Piyush Sharma	(191500547)
Anmol Pandey	(191500126)
Shantnu Maheshwari	(191500745)
Ankit	(191500119)

Course: B.Tech (Computer Science and Engineering)

Year: 3rd

Semester: 6th

Supervised By:
Mr. Mayank Saxena, Technical Trainer

Department of Training And Development, GLA University



**Department of computer Engineering and
Applications GLA University, Mathura**

**17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406**

Certificate

This is to certify that the above statements made by the candidates are correct to the best of my/our knowledge and belief.

Supervisor

Mr. Mayank Saxena

Technical Trainer

Dept. of T&D

Project Coordinator

(Mr. Mayank Saxena)

Program Coordinator

(Mr. Shashi Shekhar)

About the Project

The project aims to build a one-stop solution for all problem solvers. The name Talkify stems from ‘chating’ comprising chat app.

The purpose of this project is to design a chat application, also known as a instant messaging system. The main purpose of the software is to provide users with an instant messaging tool that has the ability to handle millions of users simultaneously when needed and can be easily done.

This project created a web application for users to instantly communicate with each other. An the most important part of building a good chat application is focus on the data flow on web. This project spends a good amount of research to find the most appropriate technology for deliver the data flow, which is REST api for account management, WebSocket for text chat and WebRTC for video and audio and find the result pleasant.

Motivation

Chatting is a method of using technology to bring people and ideas together despite of the geographical barriers. The technology has been available for years but the acceptance was quite recent. Our project is an example of a chat server. It is made up of two applications - the client application, which runs on the user's web browser and server application, runs on any hosting servers on the network. To start chatting client should get connected to server where they can do private and group chat. Security measures were taken during the last one. Social media can help to improve an individual's sense of connectedness with real or online communities and can be an effective communication (or marketing) tool for corporations, entrepreneurs, non-profit organizations, advocacy groups, political parties, and governments. Observers have also seen that there has been a rise in social movements using social media as a tool for communicating and organizing in times of political unrest.

Requirements

a). Software Requirements:

- Technology Implemented: Full Stack Web Development
- Languages/Technologies Used: HTML, TailWind CSS, React, JAVASCRIPT , Firebase.
- IDE Used: Visual Studio Code
- Web Browser: Google Chrome

GitHub: GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere. **GitHub Repository:** A GitHub repository can be used to store a development project. It can contain folders and any type of files (HTML, CSS, React, JavaScript, Documents, Data, Images). A GitHub repository should also include a license file and a README file about the project. A GitHub repository can also be used to store ideas, or any resources that you want to share.

Visual Studio Code: Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. [7] Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. Microsoft has released Visual Studio Code's source code on the VS Code repository of GitHub.com, under the permissive MIT License, while the compiled binaries are freeware.

b). Hardware Requirements:

- Processor Required: Intel i5
- Operating System: Windows 10, 8,7
- RAM: 8GB
- Hardware Devices: Computer System
- Hard Disk: 256GB

Acknowledgement

We thank the almighty for giving us the courage and perseverance in completing the project. This project itself is an acknowledgement for all those people who have given us their heartfelt co-operation in making this project a grand success. We extend our sincere thanks to Mr. Mayank Saxena, Technical Trainer at “GLA University, Mathura” for providing his valuable guidance at every stage of this project work. We are profoundly grateful towards the unmatched services rendered by him. And last but not least, we would like to express our deep sense of gratitude and earnest thanks giving to our dear parents for their moral support and heartfelt cooperation in doing the main project.

Talkify – A Chat Webapp

Abstract

This project created a web application for users to instantly communicate with each other. An the most important part of building a good chat application is focus on the data flow on web. This project spends a good amount of research to find the most appropriate technology for deliver the data flow, which is REST api for account management, WebSocket for text chat and WebRTC for video and audio and find the result pleasant.

Future Scope

We have analyzed that as we are move further in the technology people have less time to present physically everywhere.

So it's a high time where we can create something which can beneficial in future usage.

And What can be better than a messaging app which connect various of people without physical presence.

Contents

- **Abstract**

1. Introduction	13
▪ Introduction	13
▪ Pre-requisites.....	17
2. Implementation Detail	18
▪ HTML.....	18
▪ CSS.....	19
▪ Java Script.....	20
▪ React JS.....	21
▪ Material UI.....	22
3. Design & Planning	23
4. Advantages	25
5. List of Figures	26
6. Testing	32
7. Conclusion	36

Abstract

This project created a web application for users to instantly communicate with each other. An the most important part of building a good chat application is focus on the data flow on web. This project spends a good amount of research to find the most appropriate technology for deliver the data flow, which is REST api for account management, WebSocket for text chat and find the result pleasant.

Chapter 1

Introduction

Messaging apps now have more global users than traditional social networks—which means they will play an increasingly important role in the distribution of digital journalism in the future. Drawing upon our interviews and case studies, we identify a number of opportunities and challenges for organizations using—or hoping to use—messaging apps for news. We argue that to devise a successful messaging app strategy, publishers must understand regional strongholds, user demographics, and popular features of each app. As happened after the early days of social media, before which a proliferation of services (some with regional strengths) led to intense competition for user attention, we expect to see some eventual consolidation among chat apps. Elsewhere, we conclude that issues around information, privacy, personal security, and mobile data penetration will unfold in different ways around the world; apps like Telegram and FireChat are among those at the forefront of addressing and solving these problems. In developing editorial strategies for some of these wide-ranging messaging platforms, news organizations are not just helping to future-proof themselves, they are also venturing into online spaces that could enable them to reach hundreds of millions of (often young) people with whom they have never engaged before.

AIM

The main aim of this project is to have multiple option for a user which provide full security to provide a user a better experience.

Existing System

There are already many chatting or messaging webapp like whatsapp , telegram etc. But they are highly advance for a new user who has just started using these messaging app. We are tryig to capture the newly user who want a simple and a clean user interface to message.

Feasibility Study

A feasibility study is a high-level capsule version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it's worth doing. Once an acceptance problem definition has been generated, the analyst develops a logical model of the system. A search for alternatives is analyzed carefully. There are 3 parts in feasibility study.

1) Operational Feasibility

2) Technical Feasibility

3) Economical Feasibility

Operational feasibility

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture and existing business processes. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability,

supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases.

Technical Feasibility

This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on outline design of system requirements in terms of input, processes, output, fields, programs and procedures. This can be qualified in terms of volume of data, trends, frequency of updating in order to give an introduction to the technical system. The application is the fact that it has been developed on windows XP platform and a high configuration of 1GB RAM on Intel Pentium Dual core processor. **ECONOMICAL FEASIBILITY-** Establishing the cost-effectiveness of the proposed system i.e. if the benefits do not outweigh the costs then it is not worth going ahead. In the fast paced world today there is a great need of online social networking facilities. Thus the benefits of this project in the current scenario make it economically feasible. The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification-

and identification of all the benefits expected. This assessment typically involves a cost/benefits analysis.

Pre-requisite

Hands-on knowledge of JavaScript, HTML and CSS, React, Firebase is essential before working on the concepts for making of webpages. Make sure that you have the browser or chrome installed and running before opening web.

Chapter 2

Implementation Details

HTML

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as `` and `<input />` directly introduce content into the page. Other tags such as `<p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

Tailwind CSS

Tailwind is a form of CSS only.

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

CSS information can be provided from various sources. These sources can be the web browser, the user and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule

order, inheritance and property definition. CSS style information can be in a separate document or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately for each medium. The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. The process is called cascading.

One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes.

JavaScript

JavaScript is a high-level, interpreted scripting language that conforms to the ECMAScript specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it, and major web browsers have a dedicated JavaScript engine to execute it. As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has APIs for working with text, arrays, dates, regular

expressions, and the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities. It relies upon the host environment in which it is embedded to provide these features.

Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

The terms Vanilla JavaScript and Vanilla JS refer to JavaScript not extended by any frameworks or additional libraries. Scripts written in Vanilla JS are plain JavaScript code. Google's Chrome extensions, Opera's extensions, Apple's Safari 5 extensions, Apple's Dashboard Widgets, Microsoft's Gadgets, Yahoo! Widgets, Google Desktop Gadgets, and Serence Klipfolio are implemented using JavaScript.

React JS

ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library responsible only for the view layer of the application. It was created by Jordan Walke, who was a software engineer at Facebook. It was initially developed and maintained by Facebook and was later used in its products like WhatsApp & Instagram. Facebook developed ReactJS

A ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code. The components are the heart of all React applications. These Components can be nested with other components to allow complex applications to be built of simple building blocks. ReactJS uses virtual DOM based mechanism to fill data in HTML DOM.

React came into the market and gained popularity amongst them. The previous frameworks follow the traditional data flow structure, which uses the DOM (Document Object Model). DOM is an object which is created by the browser each time a web page is loaded. It dynamically adds or removes the data at the back end and when any modifications were done, then each time a new DOM is created for the same page. This repeated creation of DOM makes unnecessary memory wastage and reduces the performance of the application.

Material UI

Material UI is an open-source, front-end framework for React components that has 60,500 plus stars on github. It is built using Less. Less (stands for Leaner Style Sheets), is a backward-compatible language extension for CSS. Material UI is based on Google's Material Design to provide a high-quality, digital experience while developing front-end graphics. Material Design focuses on providing bold and crisp designs – it builds textures by focusing on how the components cast shadows and reflect light.

Chapter-3

Design & Planning

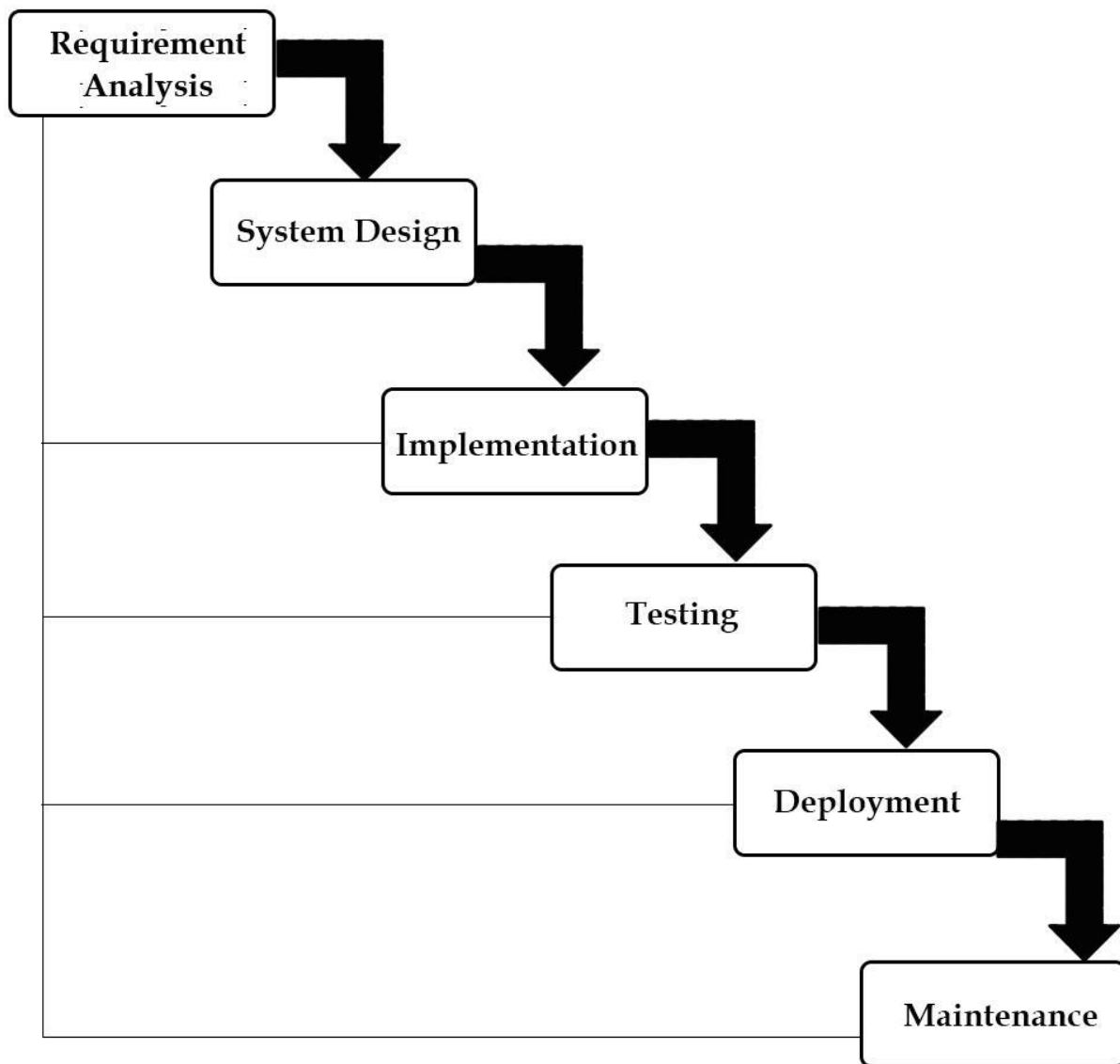
Waterfall Model

The waterfall model was selected as the SDLC model due to the following reasons:

- Requirements were very well documented, clear and fixed.
- Technology was adequately understood.
- Simple and easy to understand and use.
- There were no ambiguous requirements.

Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

Well understood milestones. Easy to arrange tasks.



Chapter- 4

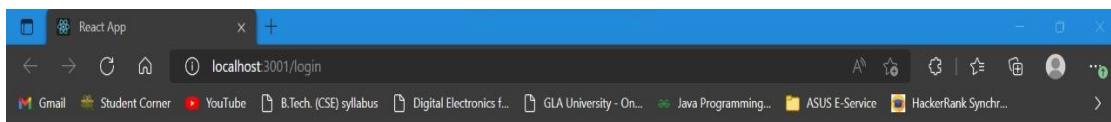
Advantages:

- It overcomes all the problems of existing system.
- Clean and a simple user interface.

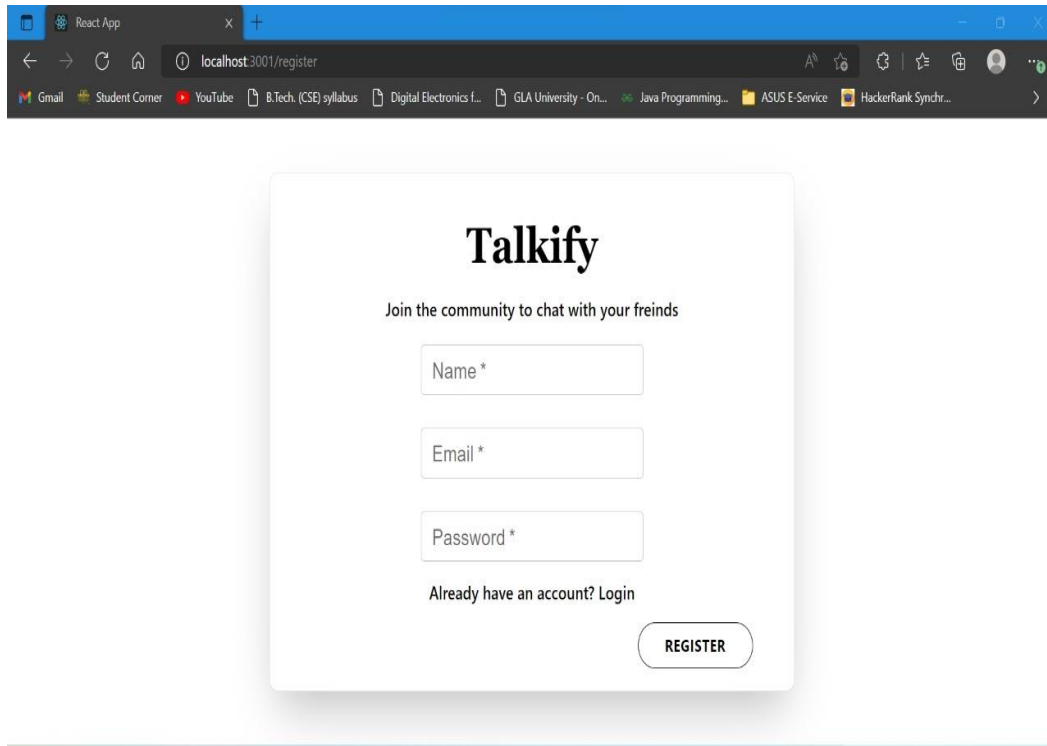
Chapter- 5

List of Figures

1. Login Page



2. Register



The image shows a web browser window with the address bar displaying 'localhost:3001/register'. The browser's tab is labeled 'React App'. The page content is a registration form for 'Talkify'. The form has a title 'Talkify' in a large, bold, serif font. Below the title is a subtitle 'Join the community to chat with your freinds'. There are three input fields: 'Name *', 'Email *', and 'Password *'. Below these fields is a link 'Already have an account? Login'. At the bottom right of the form is a button labeled 'REGISTER'. The browser's address bar shows several bookmarks: Gmail, Student Corner, YouTube, B.Tech. (CSE) syllabus, Digital Electronics f..., GLA University - On..., Java Programming..., ASUS E-Service, and HackerRank Synchr... A horizontal line is visible below the form.

React App

localhost:3001/register

Gmail Student Corner YouTube B.Tech. (CSE) syllabus Digital Electronics f... GLA University - On... Java Programming... ASUS E-Service HackerRank Synchr...

Talkify

Join the community to chat with your freinds

Name *

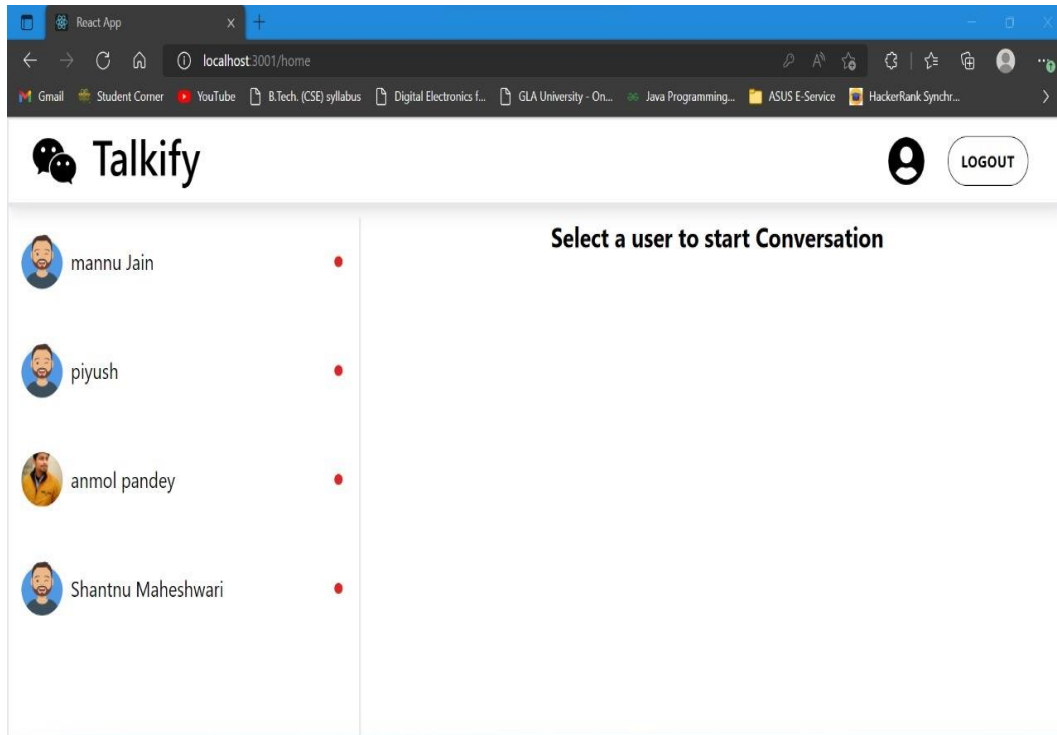
Email *

Password *

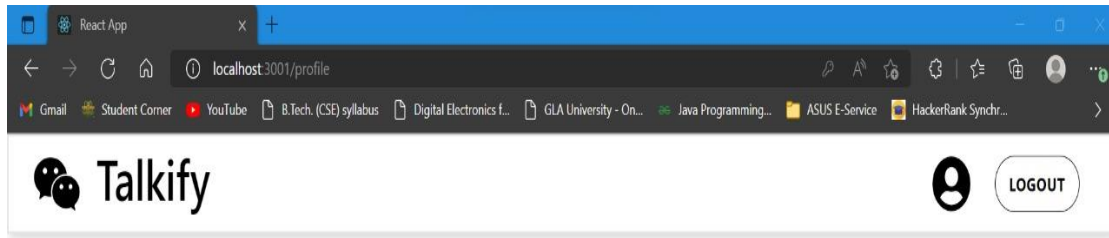
Already have an account? [Login](#)

REGISTER

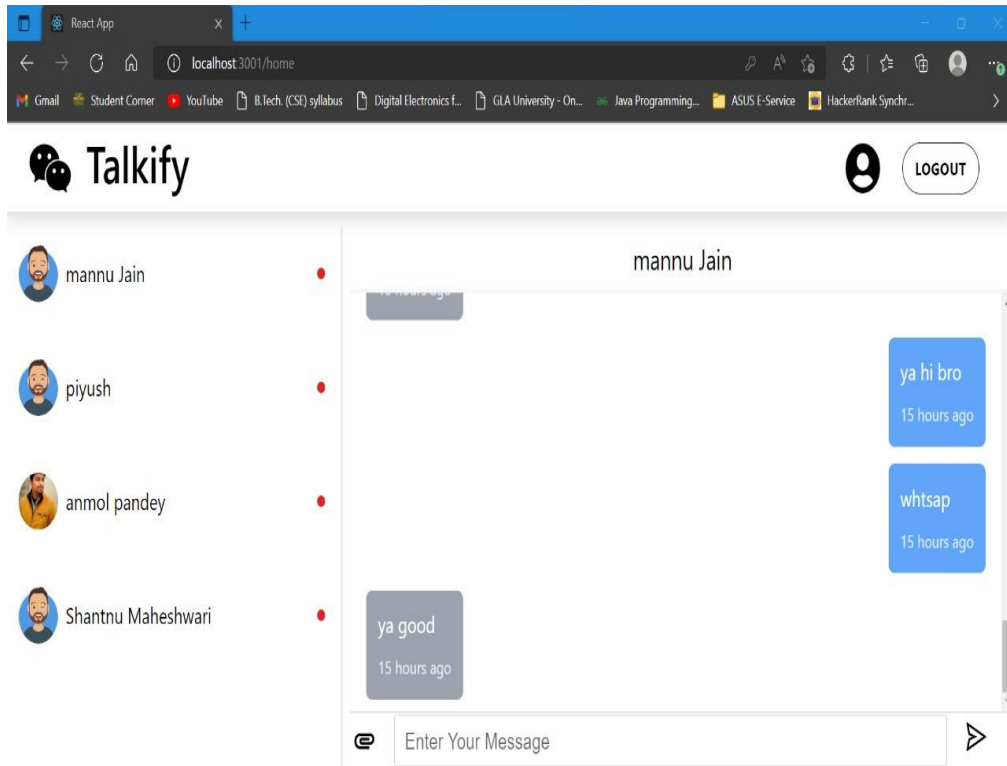
3. Home page



4. Profile Page



5. Messaging Page



Chapter- 6

Software Testing

Once source code has been generated, software must be tested to uncover as many errors as possible before delivery. It is very important to work the system successfully and achieve high quality of software. Testing include designing a series of test cases that have a high likelihood of finding errors by applying software-testing techniques. System testing makes logical assumptions that if all the parts of the system are correct, the goal will be successfully achieved. The system should be checked logically. Validations and cross checks should be there. Avoid duplications of record that cause redundancy of data. In other Words, Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily verified, and so that debugging, testing and modifications are eased. This goal can be achieved by making the source code as clear and straightforward as possible. Simplicity, clarity and elegance are the hallmark of good programs, obscurity, cleverness, and complexity are indications of inadequate design and misdirected thinking. Source code clarity is enhanced by structured coding techniques, by good coding style, by, appropriate supporting documents, by good internal comments,

and by feature provided in modern programming languages. The implementation team should be provided with a well-defined set of software requirement, an architectural design specification, and a detailed design description. Each team member must understand the objectives of implementation.

4.1 TERMINOLOGY

Error The term error is used in two ways. It refers to the difference between the actual output of software and the correct output, in this interpretation, error is essential a measure of the difference between actual and ideal. Error is also to used to refer to human action that result in software containing a defect or fault.

Fault is a condition that causes to fail in performing its required function. A fault is a basic reason for software malfunction and is synonymous with the commonly used term Bug.

Failure is the inability of a system or component to perform a required function according to its specifications. A software failure occurs if the behavior of the software is the different from the specified behavior. Failure may be caused due to functional or performance reasons.

4.2 TYPES OF TESTING

a. Unit Testing The term unit testing comprises the sets of tests performed by an individual programmer prior to integration of the unit into a larger system. A

program unit is usually small enough that the programmer who developed it can test it in great detail, and certainly in greater detail than will be possible when the unit is integrated into an evolving software product. In the unit testing the programs are tested separately, independent of each other. Since the check is done at the program level, it is also called program teasing.

b. Module Testing: A module and encapsulates related component. So can be tested without other system module.

c. Subsystem Testing: Subsystem testing may be independently design and implemented common problems are sub-system interface mistake in this checking we concenton it. There are four categories of tests that a programmer will typically perform on a program unit.

i Functional test

ii Performance test

iii Stress test

iv Structure test

Functional Test: Functional test cases involve exercising the code with Nominal input values for which expected results are known; as well as boundary values (minimum values, maximum values and values on and just outside the functional boundaries) and special values.

Performance Test: Performance testing determines the amount of execution time spent in various parts of the unit, program throughput, response time, and device utilization by the program unit. A certain amount of avoid expending too much effort on fine-tuning of a program unit that contributes little to the overall

performance of the entire system. Performance testing is most productive at the subsystem and system levels.

Stress Test: Stress test are those designed to intentionally break the unit. A great deal can be learned about the strengths and limitations of a program by examining the manner in which a program unit breaks.

Structure Test: Structure tests are concerned with exercising the internal logic of a program and traversing particular execution paths. Some authors refer collectively to functional performance and stress testing as “black box” testing. While structure testing is referred to as “white box” or “glass box” testing. The major activities in structural testing are deciding which path to exercise, deriving test data to exercise those paths, determining the test coverage criterion to be used, executing the test, and measuring the test coverage achieved when the test cases are exercised.

Chapter- 7

Conclusion

We have completed our project within time limit with the coordination of our team members under the supervision of our mentor Mr. Mayank Saxena.

Our project repository is available at

<https://github.com/CodeWithShashankJain/Talkify>

ReSources

www.google.com

[**www.geeksforgeeks.org**](http://www.geeksforgeeks.org)

www.youtube.com

[**www.w3schools.com**](http://www.w3schools.com)

[**www.beta-labs.in**](http://www.beta-labs.in)

