UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

INFORMATICS INSTITUTE OF TECHNOLOGY

Department of Computing

(B.Eng.) in Software Engineering

5COSC021C2: Software Development Group Project

Project SupAut: Teaching Using Effective Assistive Technology To Support Language And Communication Development In Middle School Autistic Students

Module Leader: Mr. Banuka Athuraliya

This project is partial fulfillment of the requirements for the Software Development Group Project

Team: SEMICOLON

IIT ID	UoW ID	Student Name
20210482	w1867602	Mohamed Fahim Shafna Zainab
20200730	w1833551	Munasinghe Arachchige Sandil Methmin Munasinghe
20210443	w1870513	Muhammed Azahim Muhammed Adheeb
20210729	w19128867	Mallawa Witharanage Minoli Kamwinie Dayarathne
20210129	w1870597	Melewwe Thanthrige Mewan Manodhya Amarasinghe

DECLARATION

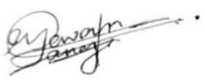
We hereby declare that the content of this project report and all related artifacts are original work and have not been previously submitted or are currently being submitted for any academic program.

Student Full Name : Mohamed Fahim Shafna Zainab
Registration No : 20210482 | w1867602
Signature :

Student Full Name : Munasinghe Arachchige Sandil Methmin Munasinghe
Registration No : 20200730 | w1833551
Signature :

Student Full Name : Muhammed Azahim Muhammed Adheeb
Registration No : 20210443 | w1870513
Signature :

Student Full Name : Melewwe Thanthrige Henry Mewan Manodhya Amarasinghe
Registration No : 20210129 | w1870597
Signature :



Student Full Name : Mallawa Witharanage Minoli Kamwinie Dayarathne
Registration No : 20210729 | w19128867
Signature :

ABSTRACT

This research study aims to investigate and identify effective strategies and tools for tracking and documenting the progress of autistic students in traditional inclusive classroom settings. The primary objective of the study is to address the challenges faced by educators in terms of enhancing the social, creative, and communication skills of autistic students. A mixed-methods approach was employed, including literature review, case studies, interviews, and surveys, to gather comprehensive data on the most pressing challenges faced by educators in teaching autistic students. This approach enabled the research team to gain a deeper understanding of the situation and identify the key areas of concern. Subsequently, a variety of research, design, development, and evaluation methodologies were implemented to filter the most efficient solutions. These solutions were developed and implemented with the goal of supporting educators in their efforts to enhance the social, creative, and communication skills of autistic students. Furthermore, the study explores the potential of assistive technologies, such as conversational AI, AI art generators, NLP techniques, and transformer-based language models, to support the language and communication development of 7-12-year-old autistic children. These solutions are developed to improve the classroom experience for the students and teachers alike. The findings of this study furnish insights on ways to systematically monitor and document the progress of autistic students and bolster their language and communication skills through assistive technologies in traditional classroom settings, thus bridging the gap in current research in this area.

Keywords: Autism, Inclusive Education, Teaching, Assistive Technology, Social, Creative, Communication Skills, Language and Communication development, conversational AI, AI art generator, NLP, Transformer-based Language model.

ACKNOWLEDGEMENT

We extend our deep appreciation to Mr. Kushan Bhareti, our mentor, and Mr. Banu Athuraliya, the leader of the Software Development Group, for their guidance and support throughout this project. We also extend our gratitude to the other instructors who provided invaluable instruction on how to compose this report. Our thanks also goes to our classmates who participated in the questionnaire and offered assistance in various ways, as well as those who provided feedback on the report's composition. Our team members deserve special recognition for their diligent efforts in putting together this report. Finally, we are grateful to our friends, family, and loved ones for their support throughout this endeavour.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Explanation
AI	Artificial Intelligence
APA	American Psychiatric Association
API	Application Programming Interface
ASD	Autism Spectrum Disorder
BCS	British Computer Society
BERT	Bidirectional Encoder Representations from Transformers
CDC	Centers for Disease Control and Prevention
DL	Deep Learning
DSM	Diagnostic and Statistical Manual of Mental Disorders
FAST	Families and Schools Together
GE	General Education
GLIDE	Guided Language to Image Diffusion for Generation and Editing
GPT	Generative Pre-trained Transformer
HTTP	Hyper Text Transfer Protocol
IEP	Individualised Educational Plans
JS	JavaScript
KPMG	Klynveld Peat Marwick Goerdeler

LR	Literature Review
ML	Machine Learning
NASS	National Autism Spectrum Disorder Surveillance System,
NGO	Non-governmental Organisations
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OOADM	Object Oriented Analysis Design Method
PHA	Public Health Agency
RNN	Recurrent Neural Network
RoBERT	Robustly Optimized BERT
SEN	Special Educational Needs
SLEP	Social, Legal, Ethical and Professional Issues
SMS	Short Message Service
SRS	System Requirements Specification
SSADM	Structured System Analysis Design Method
UI	User Interface
UNCRPD	United Nations Convention on the Rights of Persons with Disabilities
UNESCO	United Nations Educational, Scientific and Cultural Organization
UX	User Experience

VR	Virtual Reality
WBS	Work Breakdown Structure

ACKNOWLEDGEMENT

We extend our deep appreciation to Mr. Kushan Bhareti, our mentor, and Mr. Banu Athuraliya, the leader of the Software Development Group Project, for their guidance and support throughout this project. We also extend our gratitude to the other instructors who provided invaluable instruction on how to compose this report. Our thanks also goes to our classmates who participated in the questionnaire and offered assistance in various ways, as well as those who provided feedback on the report's composition. Our team members deserve special recognition for their diligent efforts in putting together this report. Finally, we are grateful to our friends, family, and loved ones for their support throughout this endeavour.

CHAPTER 1: IMPLEMENTATION

1.1 Chapter Overview

This chapter goes into great detail on how the prototype structure that was suggested in the initial research was implemented. This report provides a breakdown of the technologies, programming languages, frameworks, libraries, APIs, and documentations used to construct this project. The Project SupAut prototype's components are also thoroughly investigated, including how each feature was created, its primary purpose, relevant screenshots, and the challenges and solutions encountered during implementation.

1.2 Overview of the Prototype

- The front-end framework for this application is Reactjs. The optimum user experience has been provided using UI elements and templates. As a result, the Reactjs-built user interface can be interacted with by users.

- Back-end of this application is powered by Nodejs, Twillio serverless functions and Python.

- Open source platform Google collab have been used in creation of ML models.

- Firebase realtime database has been used for project’s database implementation.

- Finally, for both front and backend deployment Firebase hosting was used.

1.3 Technology Selections

Following is a list of the technologies selected for the Project SupAut application, including the languages selected as well as the implementation libraries and frameworks.

1.3.2 Programming Language Selection

1.3.2.1 Node JS

Node.js is a powerful, open-source, cross-platform server-side runtime environment that is based on the V8 JavaScript engine from Chrome. By utilizing JavaScript on the server-side, it enables programmers to develop scalable, rapid, and efficient network applications. In our project, Node.JS was used to create Twillio Serverless Function programs as well as database queries.

1.3.2.2 Liquid Template Language

For the purpose of creating dynamic and customised messages for SMS, email, or chat channels, Twilio uses Liquid Template Language. In Twillio Studio, queries were written in Liquid Template Language, and dynamic content was produced there. The integration of the chatbot with the use of Twillio used both output and tag markup kinds in liquid template language.

1.3.2.3 React JS

After looking over a number of programming languages that are accessible for front-end development we chose JavaScript as our programming language.React.js, an open-source JavaScript framework, is used to build front-end web applications and user interfaces (UIs). When it comes to handling testing, routing, and application state, React has a wide range of effective capabilities and tools. Reactjs was a great choice for the project since it made the use of the React Router component, which is necessary to construct dynamic pages for each student profile, seamless. From the perspective of the teacher, Project SupAut also depends on fluid navigation for a better user experience. All of the project's front-end pages were also made using Reactjs. Hence, Reactjs is the best solution for our project.

1.3.2.4 Python

High-level, interpreted programming languages like Python are popular in many fields, including web development, data analysis, artificial intelligence, scientific computing, and others. The machine learning model for SupAut's back-end development was created using Python. Python was utilized since SupAut's machine learning model is a fairly straightforward model. Python was therefore the best option for creating the machine learning model in SupAut.

1.3.3 Libraries/Frameworks Selection

1.3.3.1 Core - UI library

The free and open-source front-end web development toolkit CoreUI offers a selection of pre-designed and customizable UI components and templates for developing cutting-edge and responsive online programs. Admin Dashboard components, which are easily accessible in the CoreUI framework, are also used by Project SupAut. The user interfaces for Project SupAut were made using the Core - UI components.

1.3.3.2 Tensor Flow

TensorFlow, a free and open-source software framework, is used for dataflow and differentiable programming across a range of tasks. It allows programmers to design and train machine learning models using a variety of techniques, such as neural networks, decision trees, and regression models. Tensor Flow was used to build and train the Project SupAut machine learning model.

1.3.3.3 GPT - 3 - API

The GPT-3 API (Generative Pre-trained Transformer 3 Application Programming Interface), developed by OpenAI, is a machine learning-based technique for natural language processing that generates writing that mimics that of a human. GPT-3-API was utilized in the framework of Project SupAut to respond to a student's questions and demonstrate a genuine teacher-student interaction. Also, the conversation served as a kind of knowledge source. The Project SupAut chatbot generates statements using the GPT-3 API.

1.3.3.4 Rapid API Library

RapidAPI is a platform that offers developers a method to use one interface to access APIs from numerous sources. Developers may quickly connect with the RapidAPI platform and use its APIs by using the RapidAPI library, a Python library. Rapid API library is a great fit for SupAut because it's straightforward to integrate and has simple functionality.

1.3.3.5 Twilio REST API

With the Twilio REST API, developers can integrate voice, message, and video communication capabilities into their applications. By using simple HTTP requests, developers may easily add communication functionality to their web or mobile applications using the Twilio REST API. GET, POST, PUT, and DELETE are examples of standard HTTP operations that can be used to access the API endpoints. The chatbot component of SupAut was constructed with the aid of Twillio REST API for effective student interaction.

1.3.3.6 Twillio

A set of APIs from the cloud communications provider Twilio let programmers include messaging, audio, and video functions in their apps. With the help of Twilio's platform, companies can connect with their clients via SMS, phone, and messaging services like Facebook Messenger and WhatsApp. Twillio provides the chatapp user interface and SupAut is using the WhatsApp API since the developers found that it was the most popular.

1.3.3.7 DALL - E

To evaluate the textual descriptions and produce corresponding visuals, it combines neural networks and deep learning algorithms. DALL-E represents a major advance in artificial intelligence since it shows that machines are capable of understanding and interpreting language in a way that was previously believed to be unique to humans. DALL - E is used for visualisation in the chatbot of Project SupAut which will provide a better learning experience to the students.

1.3.4 IDE Selection

Visual Studio Code is the main IDE utilized to construct Project SupAut.

1.3.4.1 Visual Studio Code

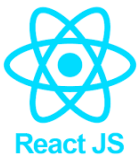
Microsoft's Visual Studio Code is a free, open-source text editor. The popularity of VS Code, one of the most well-liked programming environments in recent years, has decreased due to the editor's numerous useful features. VS Code was mostly used to code Project SupAut's front end.

1.3.5 Summary of Research Technologies and Tools

Component	Tool/Technology
Programming language	NodeJS, ReactJS,Python
ML library	TensorFlow
UI framework	React JS
IDE	Visual Studio Code, Google Collab

Table 1: Summary of research technologies and tools

1.3.6 Technology Stack

Front-End	  
Back-End	 

NLP Platform	 Google Cloud Natural Language  Open AI GPT-3  Open AI DALL-E 
Infrastructure	
Version Control	 
Services	 

Table 2: Technology stack of Project SupAut

1.5 Implementation of the Backend Component

The backend development of Project SupAut can be summarized into the following steps.

1.5.1 Introduction

1.5.1.1 Functionality

The backend of Project Suput focuses on building an automated conversational system that can act as a teacher’s assistant by helping students with any queries and ensuring they complete assigned assignments. Throughout the conversation, the chatbot gauges the student's proficiency level by posing a series of questions. The chatbot leverages advanced natural language generation and image generation capabilities, using the API provided by OpenAI, including their GPT-3.5 and DALLE models, to assess logical and visual skills, and respond to students' answers by providing constructive feedback. The Sentino API can be utilized to evaluate the creativity and quality of the student's responses, while the Twinword-Text-Similarity API can be harnessed to estimate the similarity between the student's answer and an expected answer, resulting in an assessment score. All chat logs and student answers can be persistently stored in Firebase, facilitating tracking of the student's progress through the dashboard and allowing instructors to provide additional assistance where needed.

1.5.1.2 Architecture

SupAut-Chatbot is a cloud-based chatbot developed on the Twilio platform. It uses Twilio Studio for chatbot flows and Twilio Serverless for deployment. The chatbot is designed with various widgets, such as trigger, function, set variable, split, subflow, and message widgets.

The trigger widget activates a function that retrieves the student's response from the Firebase database and evaluates it using OpenAI GPT-3.5 and DALL-E APIs to generate feedback and creative responses. Several flows within the chatbot are dedicated to specific tasks, which run function calls to execute the chatbot's logic and dynamic effects.

To evaluate creativity level and tone analysis, Sentino API is employed, while Twinword-Text-Similarity API is used for text similarity evaluation and assigning marks based

on the student's answer and model answer. The chatbot is coded in Node.js, built on a serverless architecture and uses the Firebase Admin SDK to read and write data to the database.

Overall, the SupAut-Chatbot provides an automated solution to evaluate student responses, offer feedback, and aid in learning.

1.5.2 Design and Implementation of the Database

With careful consideration it was decided that the use of Firebase Admin SDK in the SupAut project was the best way forward. The Firebase Admin SDK was employed to read and write data to the database, including retrieving students' responses and storing feedback. Firebase is a NoSQL database that offers real-time data synchronization and offline data persistence, making it highly suitable for integration into a ReactJs project. This study finds Firebase to be highly effective and easy to integrate into the SupAut project.

1.5.1.1 Firebase Realtime Database Implementation

Connecting firebase project to project SupAut and accessing various services like hosting, cloud function, cloud storage, authentication and database.

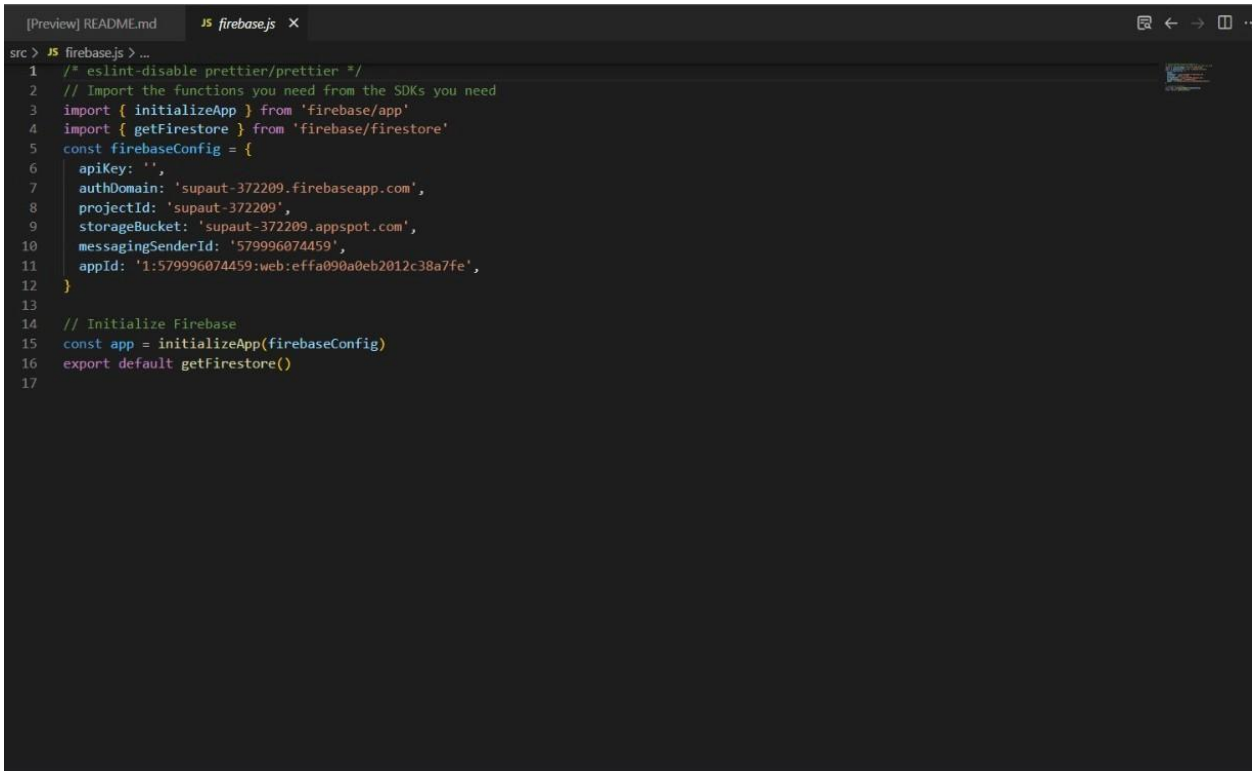
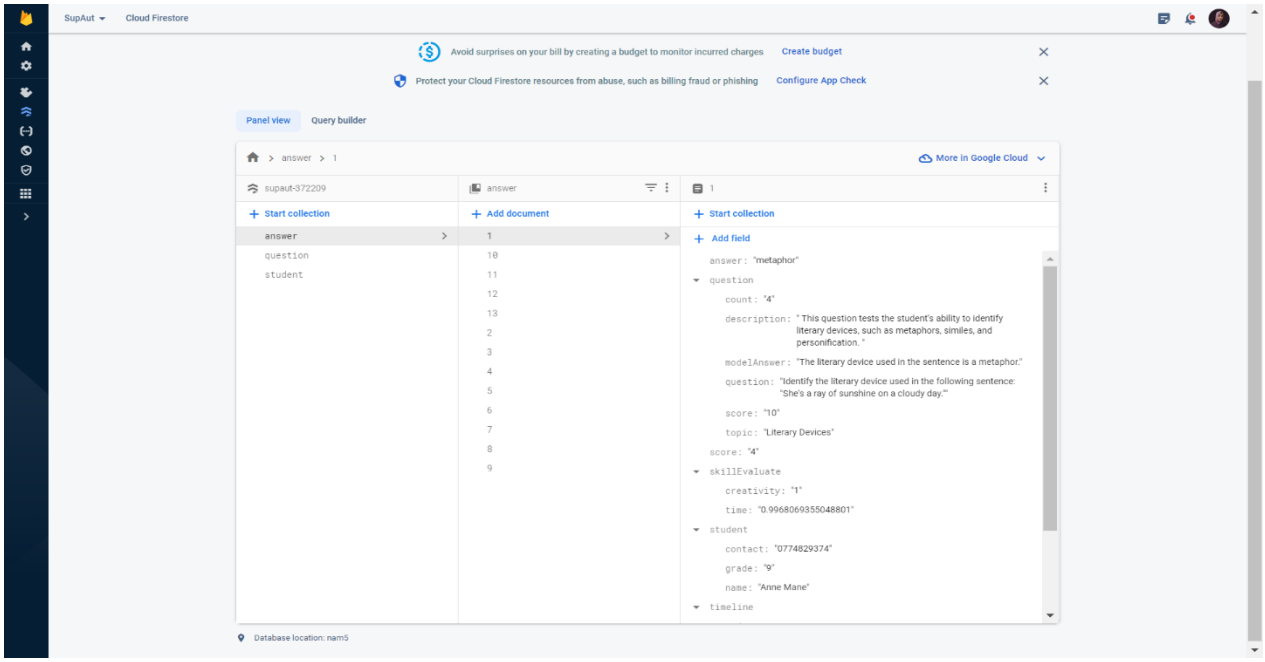


Figure 1: Firebase connection and accessing other services

Firebase database structure shown below



1.5.3 Getting Initial Endpoints and Implementation of API

Initially 4 API endpoints has been designed using Axios API service for the application and those are listed below in the table:

Method	Endpoints	Description
POST	https://sentino.p.rapidapi.com/score/facts	Used to get the Big 5 traits from a text in a scoring format
GET	https://twinword-text-similarity-v1.p.rapidapi.com/similarity/	Used to get the score of text similarity
POST	https://api.openai.com/v1/completions	Understand as well as generate natural language reposes
POST	https://api.openai.com/v1/images/generations	A model that can generate and edit images given a natural language

		prompt

Table 3: List of APIs and its details

The Axios library facilitates HTTP requests between the Chatbot and Database in Twilio serverless functions.

The Chatbot uses several APIs to evaluate student responses, including a scoring matrix, mean, standard deviation, z-score, and error function to ensure a fair evaluation.

The OpenAI GPT-3 text-davinci-003 model and DALL-E API generate feedback and creative responses while also tracking the frequency of requests for assistance. The Sentino API calculates creativity levels and tone analysis, and the Twinword-Text-Similarity API calculates text similarity and assigns marks based on the student's answer and the model answer.

The team successfully made efficient API calls by modifying them to improve the natural conversation process, smoothness, and lifelike patterns, while addressing primary concerns around performance and latency.

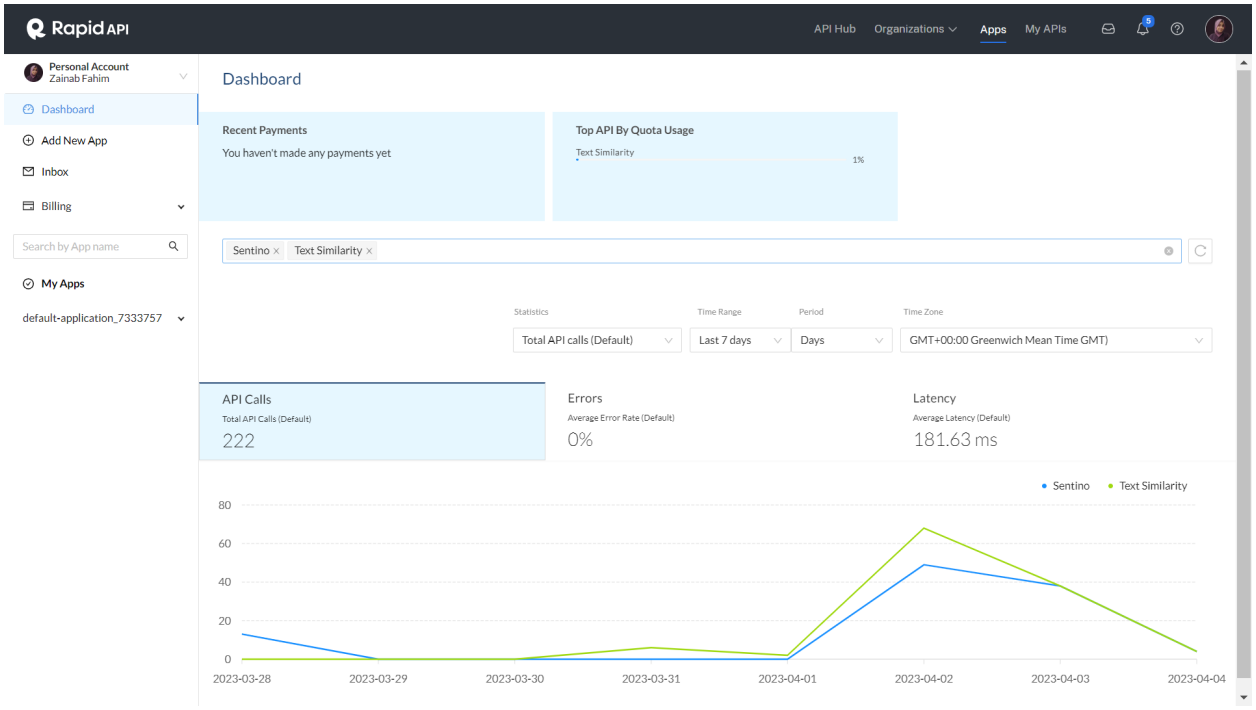


Figure 3: Rapid API performance dashboard

1.5.3 Implementation of Communication Workflows

The chatbot flow was created using Twilio Studio and includes various widgets such as webhook, function, and message widgets. When a message is received, the webhook widget triggers the function responsible for retrieving the student's response from the database, which is evaluated through various APIs, and feedback is then provided to the student via the message widget.

The Twilio Studio flow is divided into four main sections: Main SupAut Flow, SupAut Assignment Flow, SupAut Breakdown Flow, and SupAut Visualize Flow. The Main SupAut Flow is the primary chatbot flow that initiates by greeting and verifying the student, navigates through other flows using options prompted by the student, and records the student's progress. The SupAut Breakdown Flow simplifies complex instructions or concepts, while the SupAut Visualize Flow assists with visualizing concepts (Anoyiannakis, 2013). Finally, the SupAut Assignment Flow handles unanswered assigned questions. These sections are particularly

relevant when educating a student in the autism spectrum (Ghanouni et al., 2019; Gómez-Marí et al., 2022).

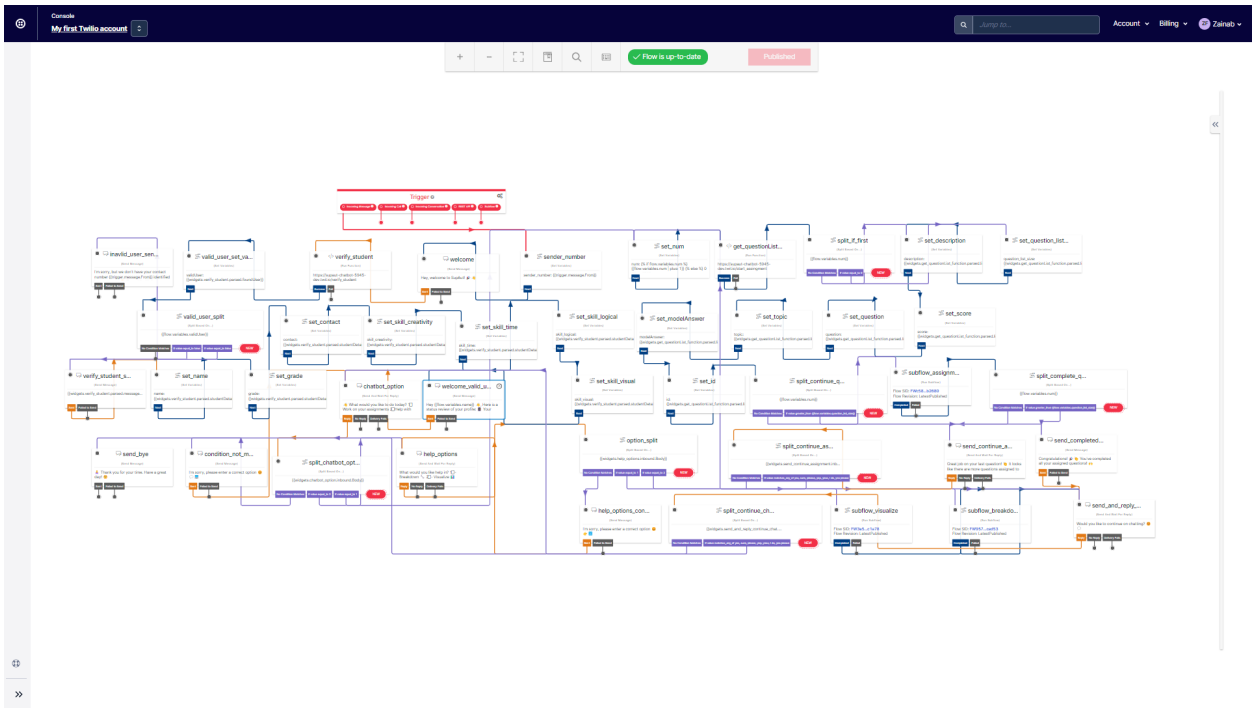


Figure 4: Main SupAut Flow (Full)

For a clearer view of a zoomed in version of the Main SupAut flow check Appendix Section A.1

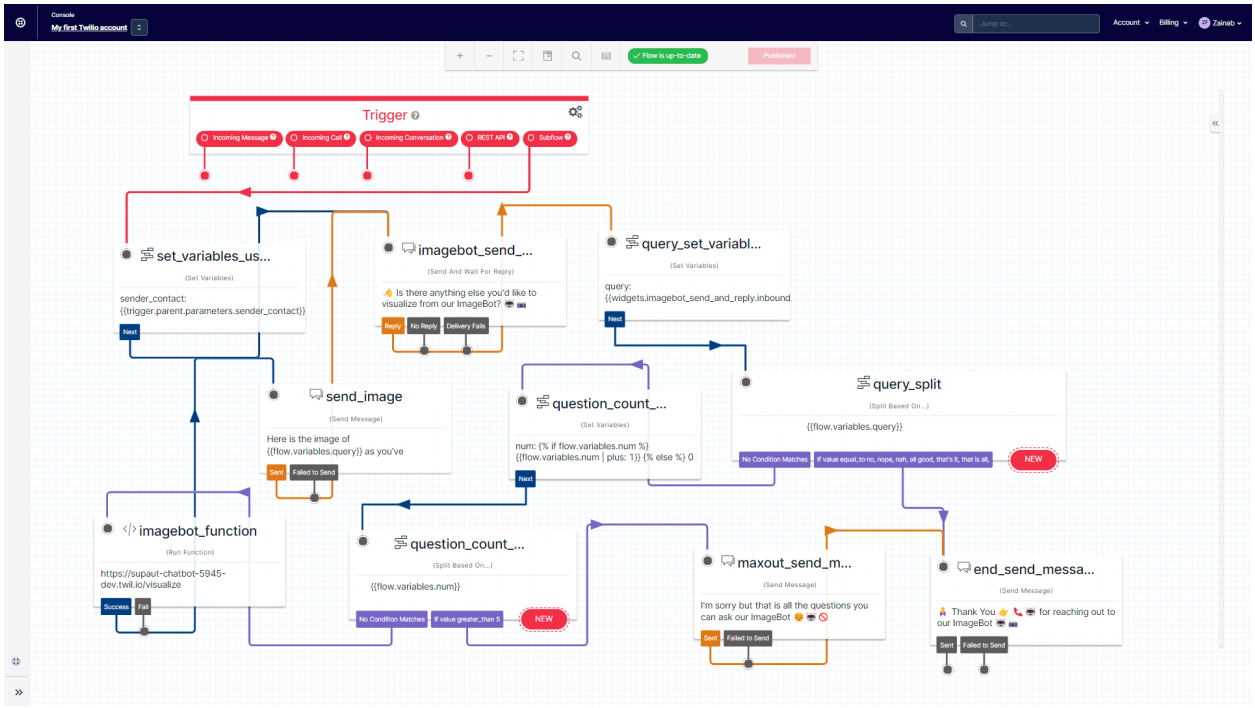


Figure 5: Visualize SupAut Flow

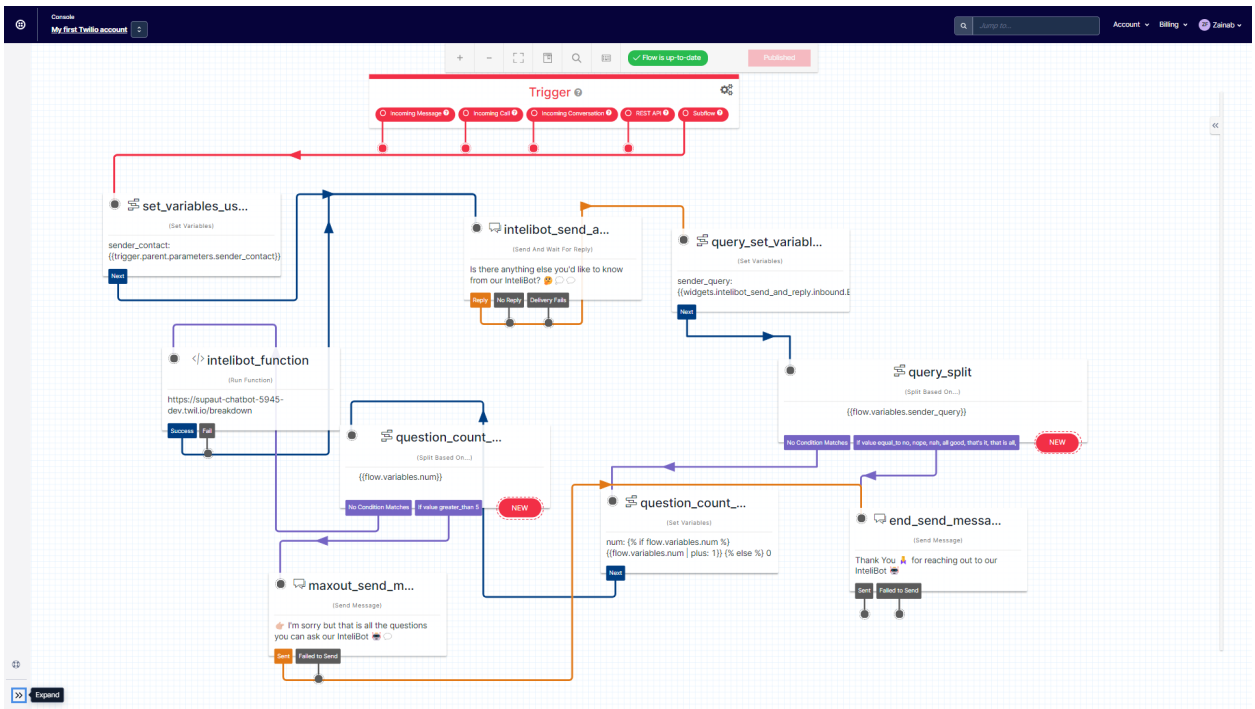


Figure 6: Breakdown SupAut Flow

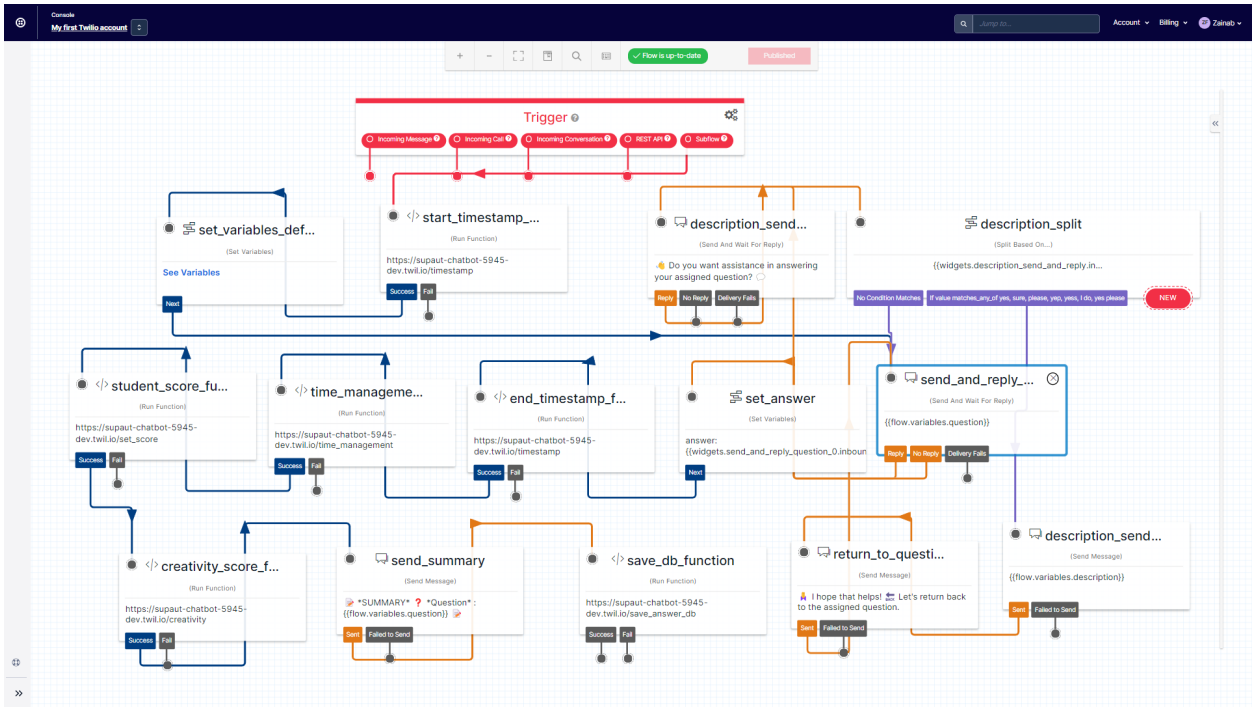


Figure 7: Assignment SupAut Flow

1.5.4 Deploying the Backend

The SupAut-Chatbot is deployed using Twilio Serverless, which provides a serverless environment for creating, deploying, and managing Twilio applications (Twilio, 2022). The chatbot's code is written in Node.js and its dependencies are listed in the package.json file. To deploy the chatbot, users need a Twilio account and a Twilio WhatsApp Sandbox. They must also install Node.js and npm, and set up the Twilio CLI (Twilio, 2022).

To ensure security and privacy, sensitive data is stored in environmental variables to restrict access. Additionally, we've used .protected extensions on the functions to secure access to function data

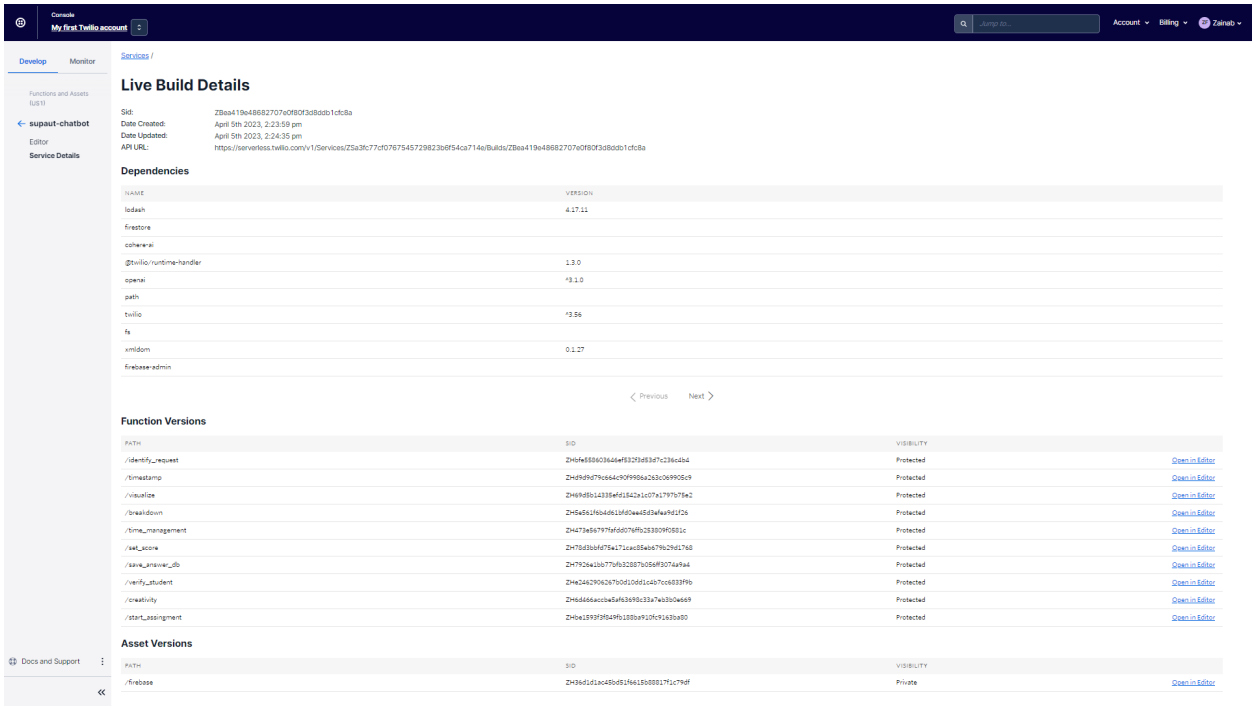


Figure 8: SupAut service build details

The deployment process requires the installation of Twilio CLI and Firebase CLI, initialization of the Firebase project, execution of twilio serverless:deploy to deploy Serverless functions, and configuration of chatbot and Firebase environment variables for deployment (Twilio, 2022). To test the chatbot, users need to use the WhatsApp Sandbox phone number. Obtain Rapid API's OpenAI GPT-3.5, DALLE, Sentino, and Twinword-Text-Similarity APIs, and set up environment variables for all API keys and Firebase credentials. The program runs as a single service, with each functionality deployed as an API function calls for the entire Supaut chatbot service, serving distinct parts of its functionality.

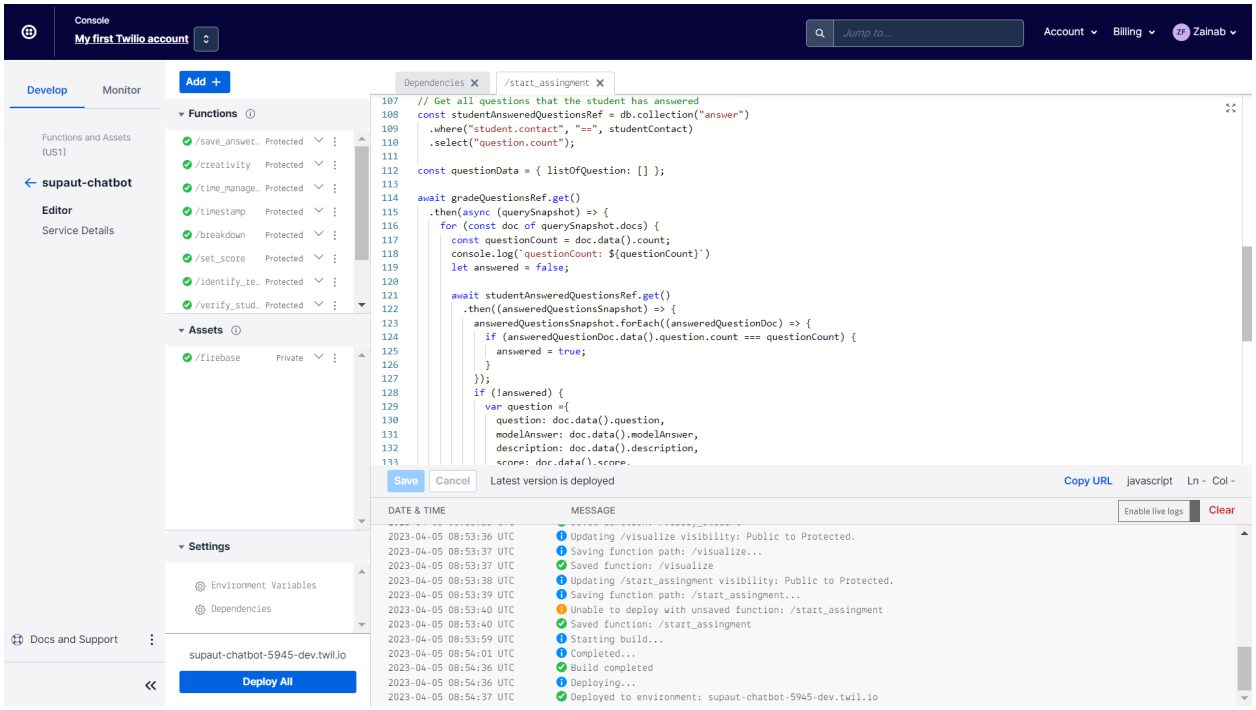
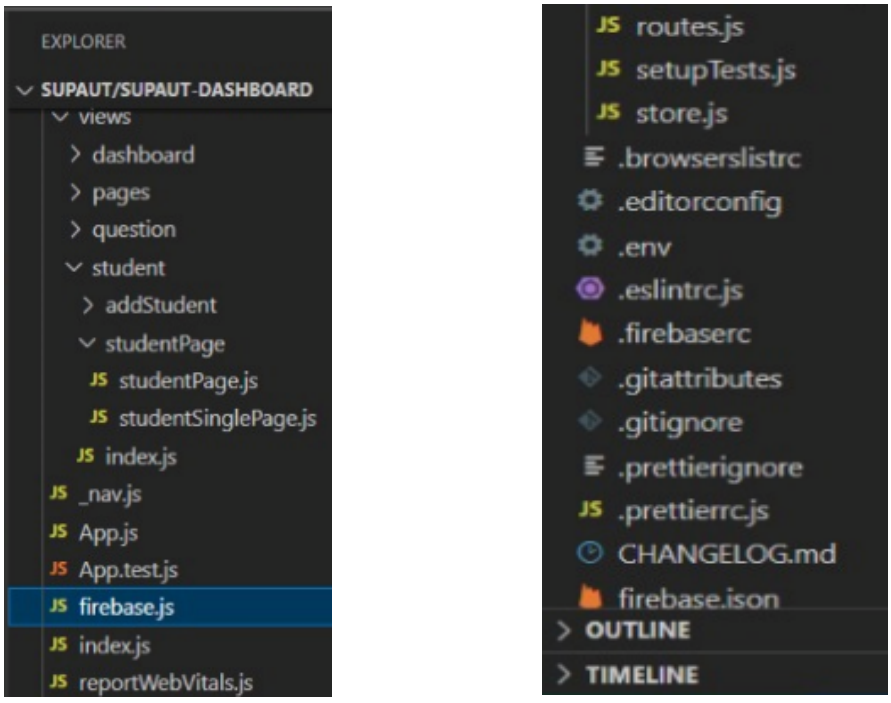


Figure 9: SupAut service deployment console

1.6 Implementation of the Front End Component

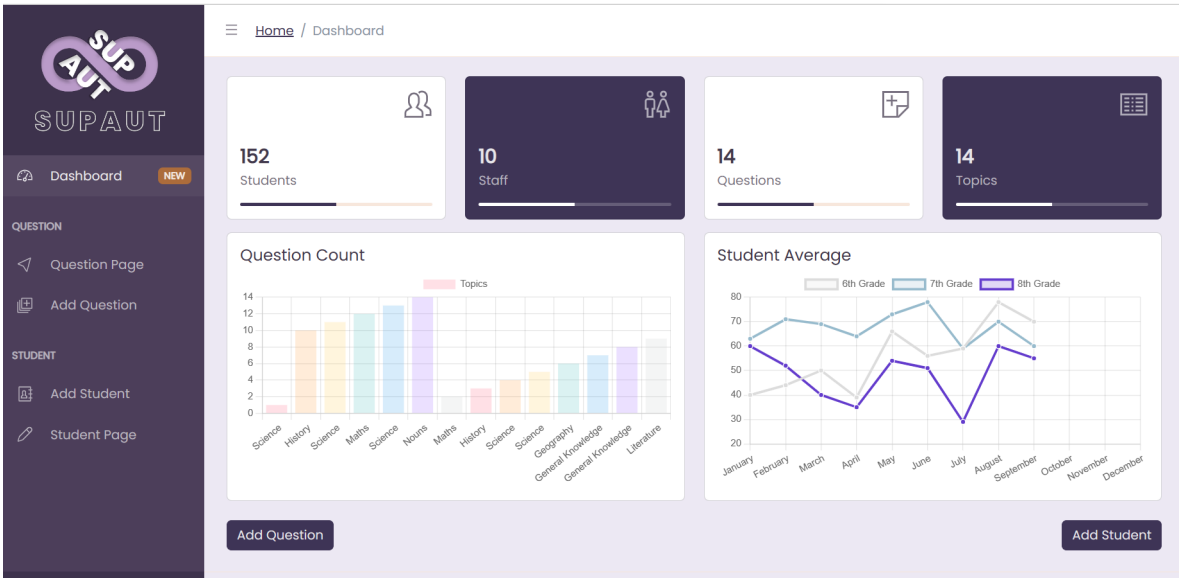
The online application for Project SupAut was created using ReactJS, one of the most well-known frontend frameworks. Another tool employed was the Core UI library. The application was created using the wireframe designs created throughout the research stages, with the necessary design adjustments made to meet the simple user experience and mentor comments. The structure of this progressive web application was entirely derived from the documentation for ReactJS and the Core UI library, and all of the key elements that make up the system have been described here, along with examples and explanations.

Front-end Folder structure is implanted as given below,



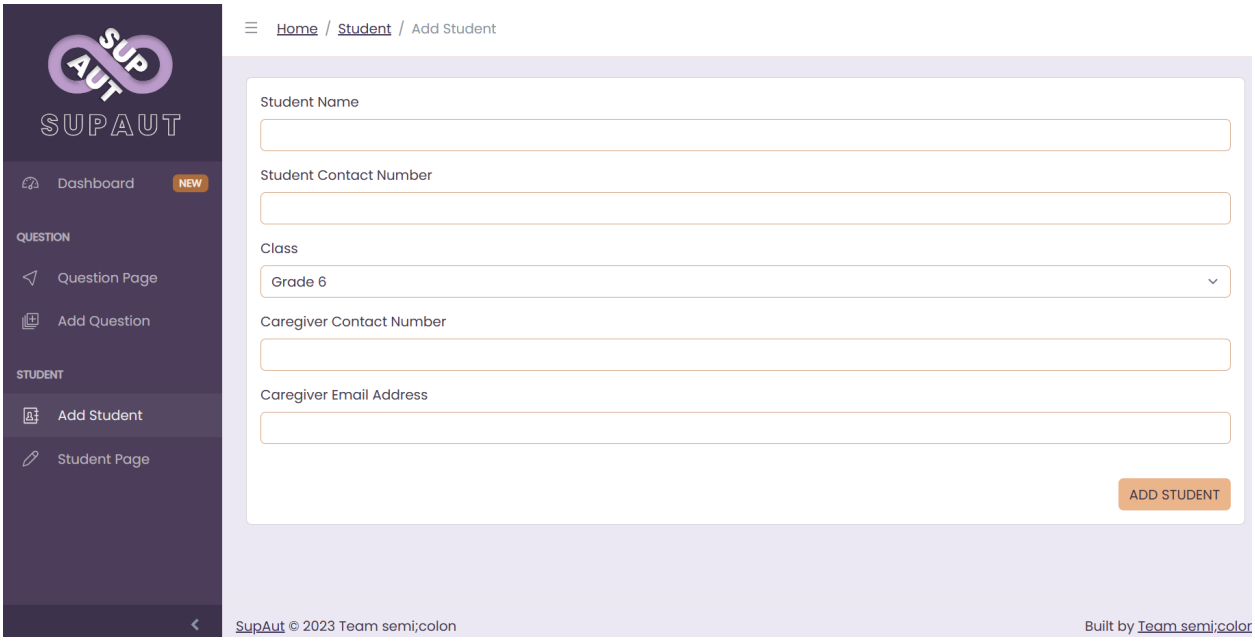
1.6.1 Dashboard Page

The dashboard will display information on the total number of students, newly hired employees, new question themes, total number of questions, and average time spent on the dashboard. This website can be used by the middle school teacher to add a pupil or a query.



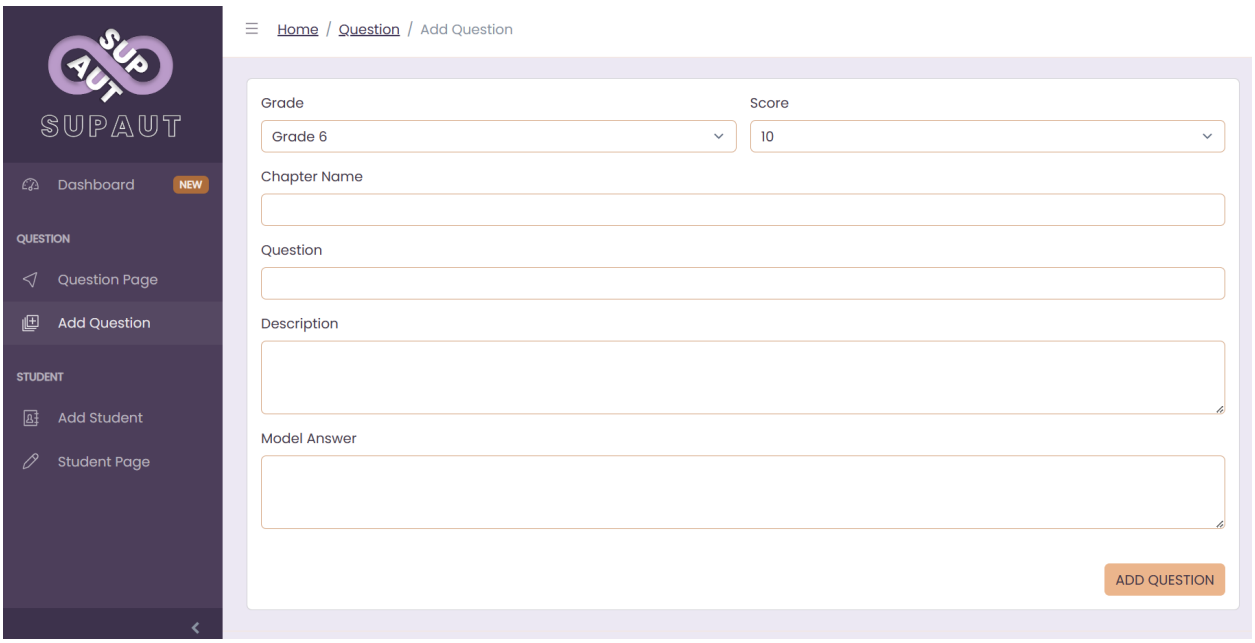
1.6.2Add Student Page

If the teacher clicks the add student button on the dashboard page, they will be directed to the add student page. The name, student contact number, class, caregiver contact number, and caregiver email address are all fields on the add student page. After all these fields have been filled in, a teacher can add a student to the SupAut system by selecting the add student button.



1.6.3 Add Question Page

If the instructor clicks the add question button on the dashboard page, they will be directed to the add question page. There are places on the website where you can enter the grade, chapter name, question, description, and model response. Once all of these fields have been filled in, the middle school teacher can click the save button to save the question's data.



1.6.4 Student Details Page

There are fields for the student's name, contact information, and email address, as well as fields for the caregiver's name, contact information, and email address. An individual student-specific progress graph can be found on the student page. This progress chart demonstrates the unique

student's visual, logical, creative, and time management skills. Based on how the students reacted to the questions, the chart will be created. If the middle school teacher chooses the download IEP report button, the IEP report can be downloaded. If the teacher selects the "make IEP report" button, an individual IEP report can be made.

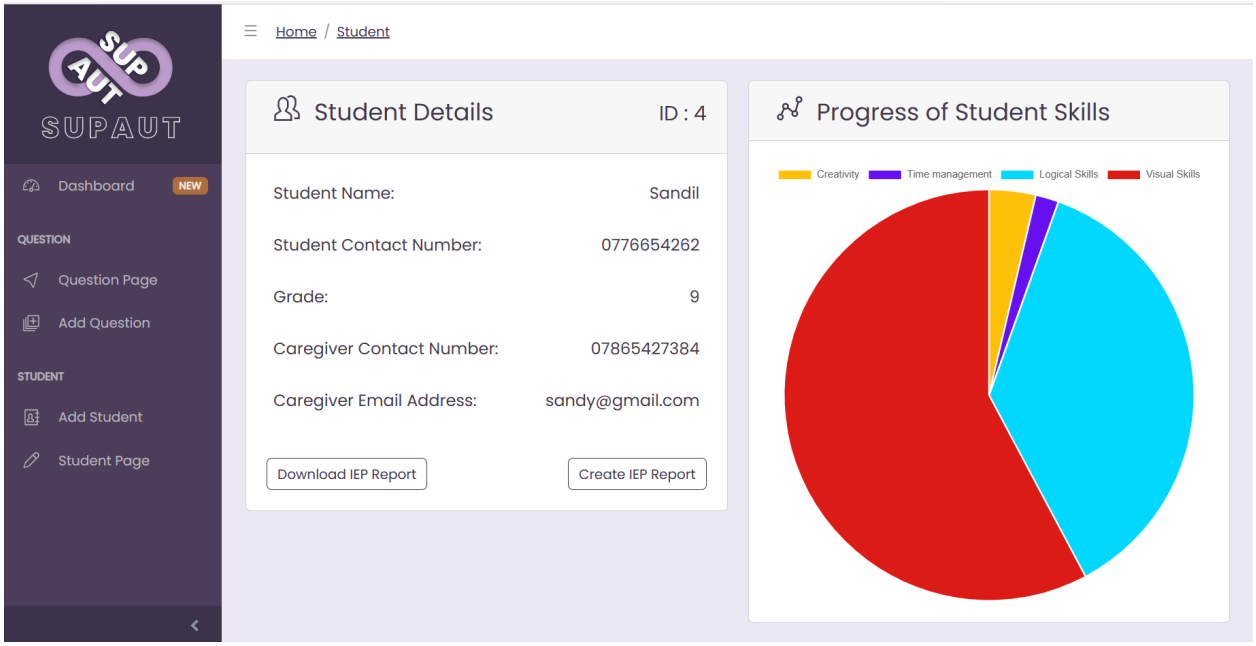


Figure 14: Student Details Page

1.6.5 Question Page

The question page shows the grade for which the question is designated, the question's number, the question, and the question's responses.

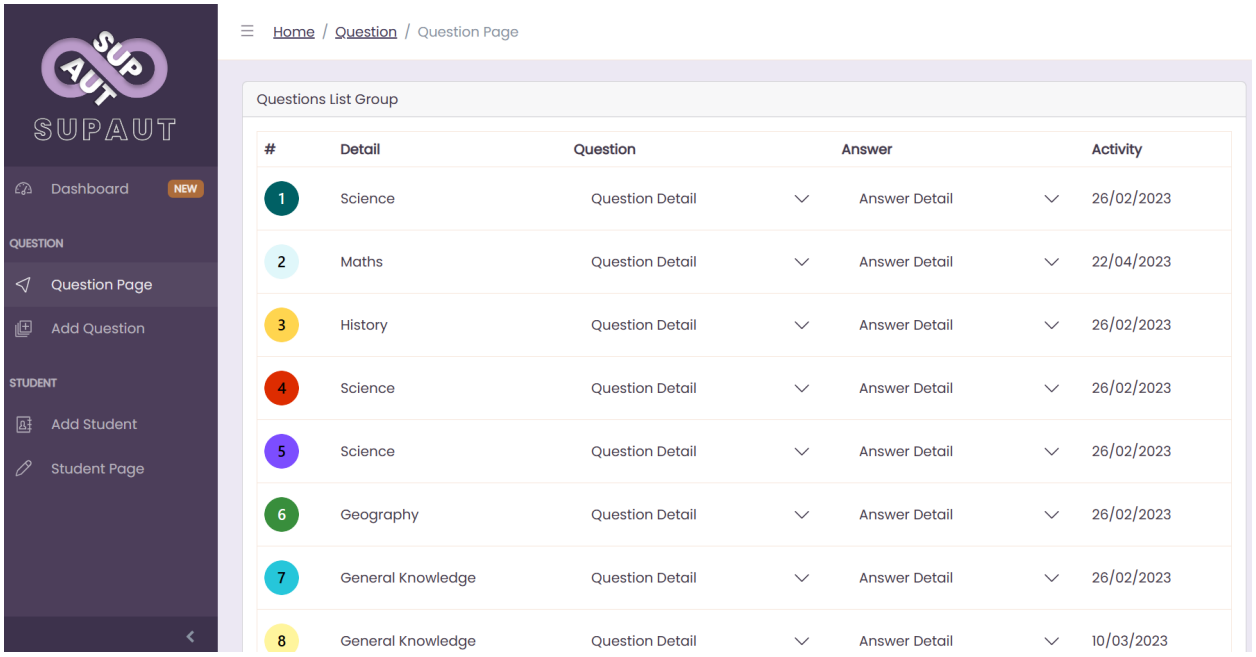
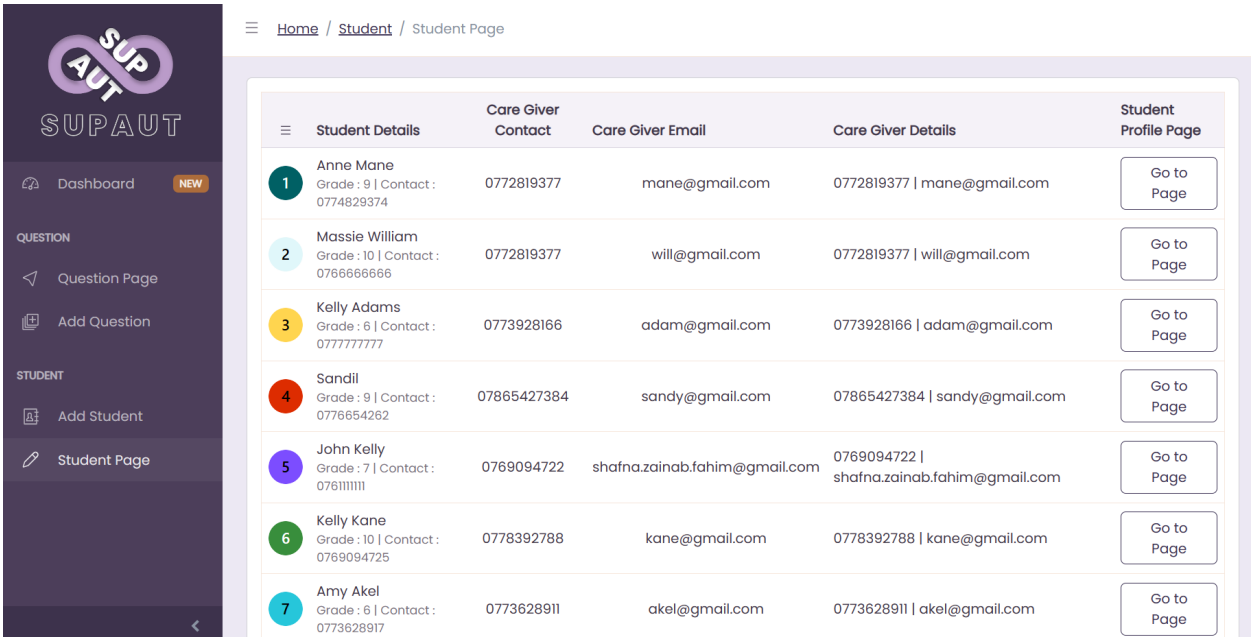


Figure 15: Question page

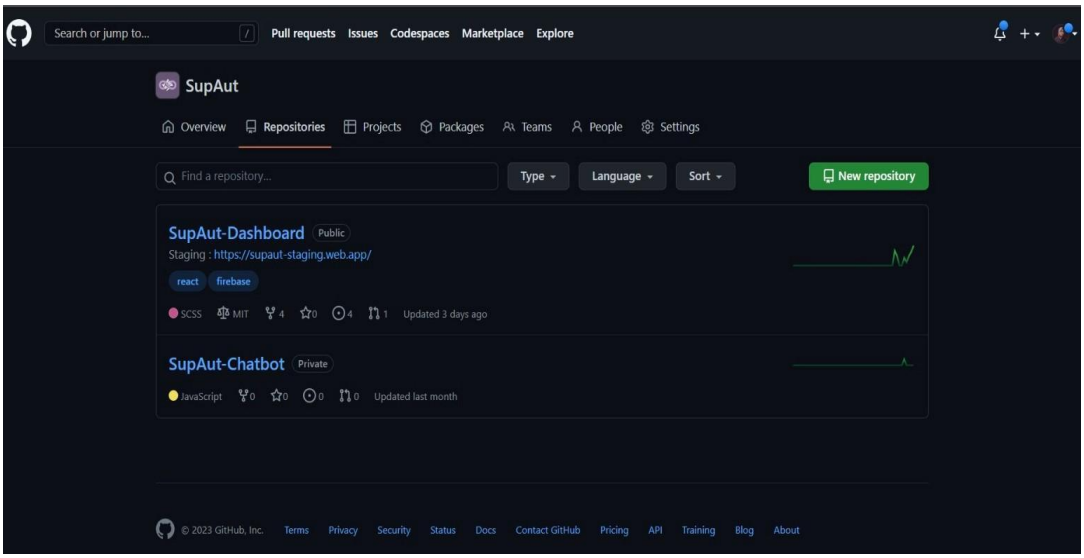
1.6.6 Student List Page

This page consists the lists of the names of the students who are added. We can click the go to page button and go to the single students profile page.



1.7 GIT Repository

Since Git is a version control system, it can keep track of changes made to any collection of files. In this project, the code base's various changes are tracked using Git. Inside the team's github organization called SupAut, Project SupAut has maintained two distinct repositories of the application called SupAut-Dashboard and SupAut-Chatbot.



In the repository on Github, issues were created. Each team member was given a problem to solve.

The issues were resolved once the assignment was finished. A pull request will be sent into the main branch after the team members have pushed the code into their remote branches. The code will be carefully examined before being merged to the main branch. So that it is always prepared to be deployed with the server, the cleanest, bug-free code will be merged with the main branch.

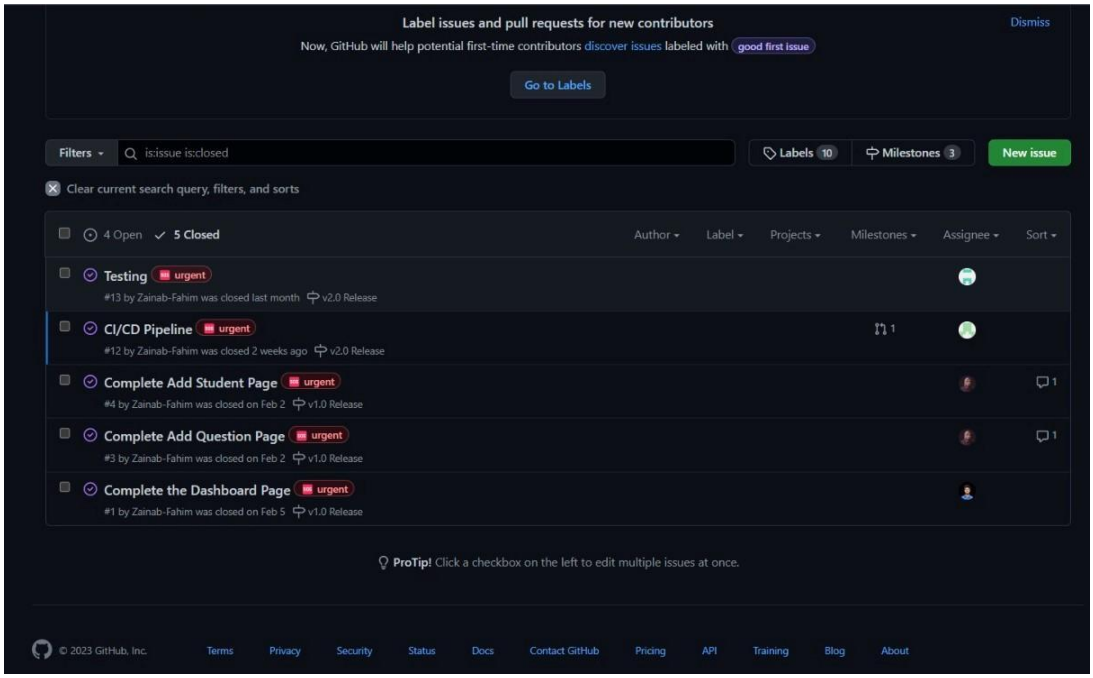


Figure 18: Issues being created in Github

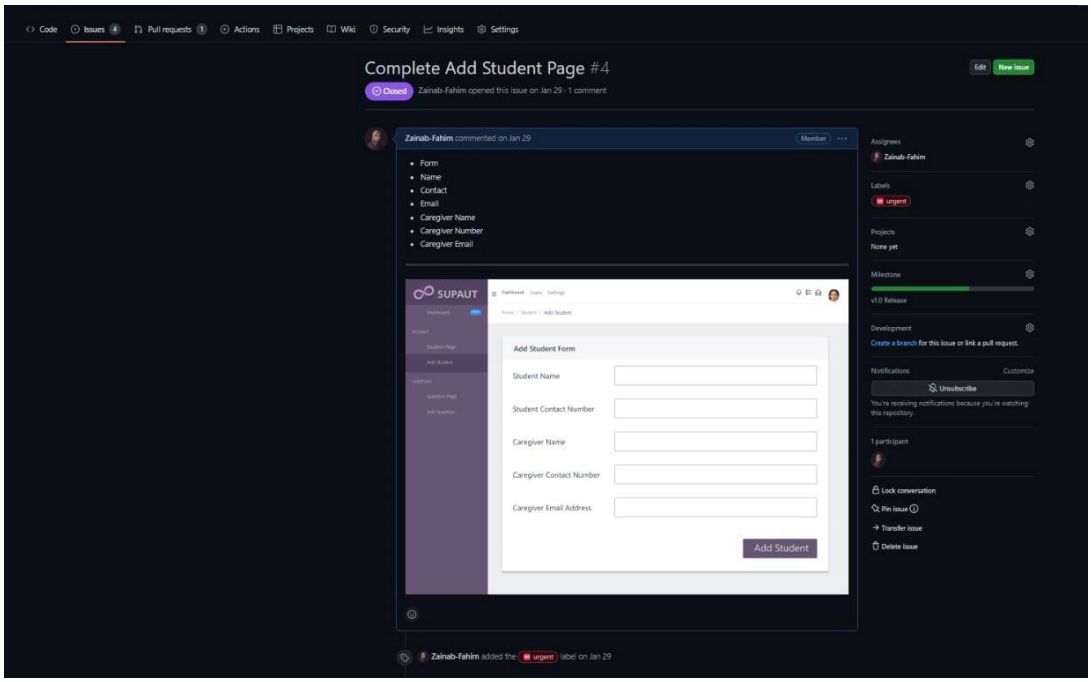


Figure 19: Issues being closed after the tasks are completed

SupAut Frontend

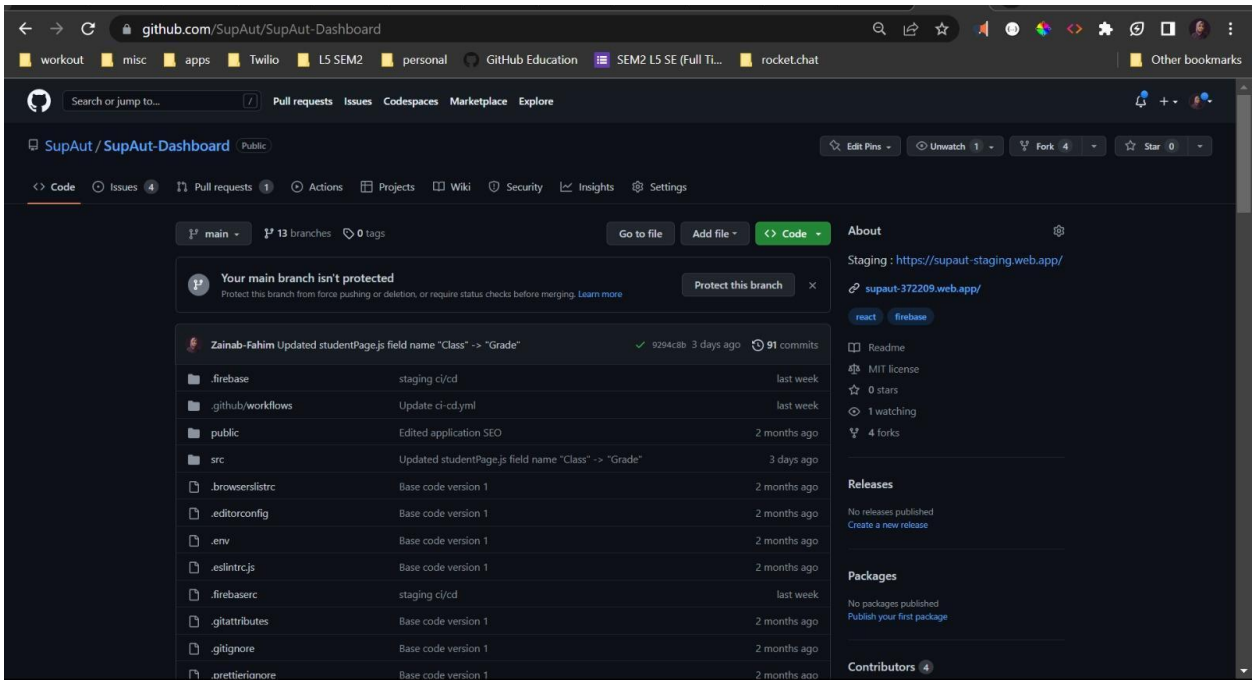


Figure 20: Project SupAut Frontend Repository

1.8 Deployments/CI-CD Pipeline

1.8.1 Deployments

1.8.1.1 Firebase Hosting

The application's front end was hosted by Firebase. Web application hosting for Firebase is quick and safe. The ReactJS-based frontend was therefore chosen to be hosted by Firebase. The major reasons we chose Firebase were that it made it simple to deploy our web application and that we could quickly create a custom domain for it.

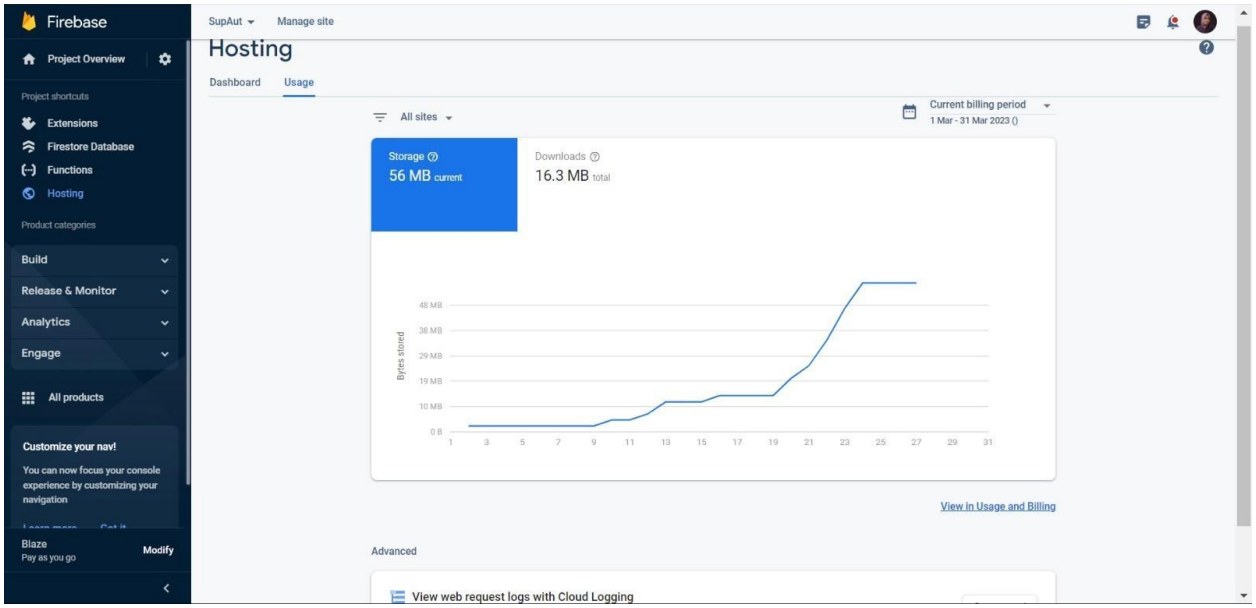


Figure 21: Firebase deployment for the front-end

1.8.2 CI/CD Pipelines

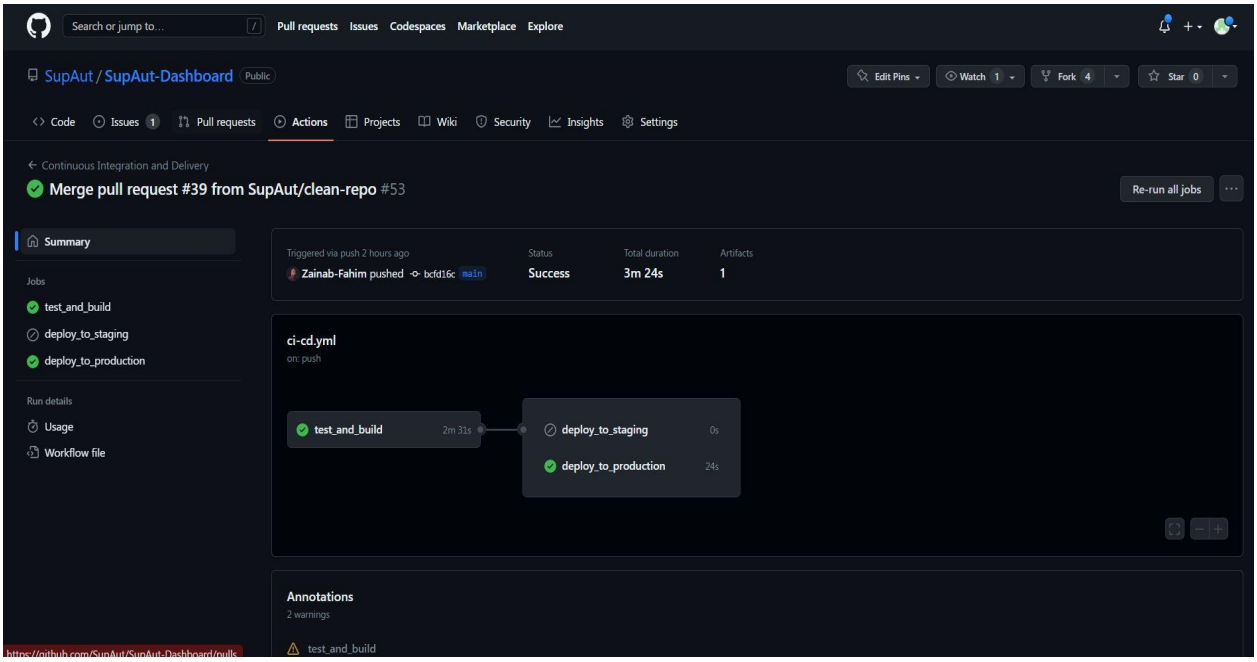
Two current methods of software development are continuous integration and continuous deployment. Two CI/CD pipelines are interconnected for the frontend and backend of Project SupAut. These CI/CD pipelines included frontend and backend integrations. GitHub Actions are mostly used to implement these CI/CD processes.

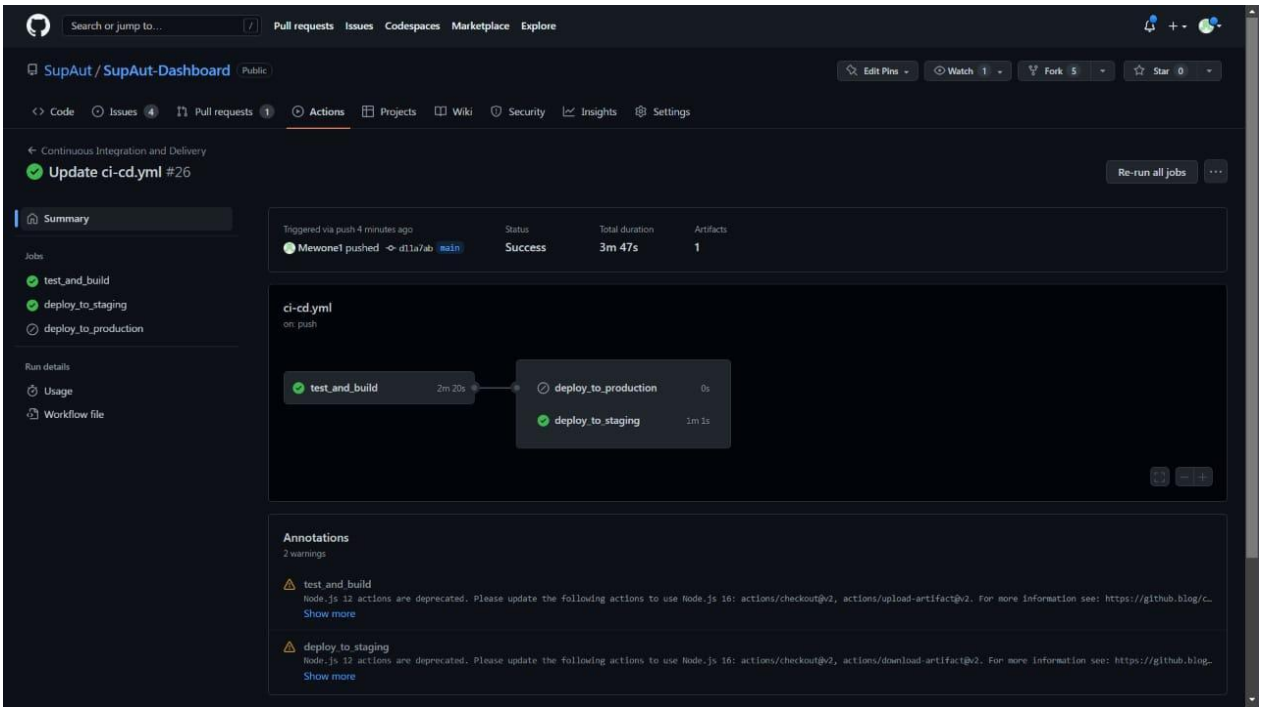
1.8.2.1 Front-End

CI - Github actions continuous integration

CD - Firebase auto-continous deployment

All pipeline codes are attached in the Appendix section


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
SupAut / SupAut-Dashboard
Public
Edit Pins
Watch 1
Fork 4
Star 0
Code
Issues 1
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Continuous Integration and Delivery
Merge pull request #39 from SupAut/clean-repo #53
Re-run all jobs
Summary
Jobs
test_and_build
deploy_to_staging
deploy_to_production
Run details
Usage
Workflow file
Triggered via push 2 hours ago
Zainab-Fahim pushed bcfd16c main
Status
Success
Total duration
3m 24s
Artifacts
1
ci-cd.yml
on: push
test_and_build 2m 31s
deploy_to_staging 0s
deploy_to_production 24s
Annotations
2 warnings
test_and_build


Search or jump to...
Pull requests
Issues
Codespaces
Marketplace
Explore
SupAut / SupAut-Dashboard
Public
Edit Pins
Watch 1
Fork 5
Star 0
Code
Issues 4
Pull requests 1
Actions
Projects
Wiki
Security
Insights
Settings
Continuous Integration and Delivery
Update ci-cd.yml #26
Re-run all jobs
Summary
Jobs
test_and_build
deploy_to_staging
deploy_to_production
Run details
Usage
Workflow file
Triggered via push 4 minutes ago
Mewone1 pushed d11a7ab main
Status
Success
Total duration
3m 47s
Artifacts
1
ci-cd.yml
on: push
test_and_build 2m 20s
deploy_to_production 0s
deploy_to_staging 1m 1s
Annotations
2 warnings
test_and_build
Node.js 12 actions are deprecated. Please update the following actions to use Node.js 16: actions/checkout@v2, actions/upload-artifact@v2. For more information see: https://github.blog/c...
Show more
deploy_to_staging
Node.js 12 actions are deprecated. Please update the following actions to use Node.js 16: actions/checkout@v2, actions/download-artifact@v2. For more information see: https://github.blog...
Show more

1.9 Chapter Summary

The creation of the Twillio chatbot's backend, a prototype, and Technology Picks (Data Picks, Programming Language Picks, Framework Picks, and IDE Picks). The implementation of the frontend component, the deployments, the GIT Repository, and the backend component are all briefly covered in this part.

CHAPTER 2: TESTING

2.1 Chapter Introduction

This chapter pertains to the testing phase of SupAut web application. The aim of this phase is to validate and review all the functional and non-functional requirements established during the research stage. The testing process includes unit testing, which ensures that each component of the application functions as intended. Additionally, performance testing is conducted to guarantee that the application does not crash or become unresponsive while in use. Usability and compatibility tests are also performed to improve the user experience. The results of these tests will be thoroughly reviewed to identify any issues and enhance the application accordingly.

2.2 Testing Criteria

In order to ensure that the SupAut education system meets its intended objectives, it is necessary to test both its functional and non-functional requirements against the business and quality requirements of the system. Functional testing evaluates the application's features and capabilities, while non-functional testing examines the application's overall quality. For instance, performance testing and accuracy testing are types of non-functional testing that assess how well the application performs and how precise its output is. It is crucial to conduct both types of testing to ensure that the SupAut education system delivers a high-quality user experience and meets the established objectives.

2.3 Testing Functional Requirements

The testing of functional requirements is a critical phase in the software development life cycle. It involves thoroughly validating the software application against the specified functional requirements to ensure that it performs as expected. This process includes conducting various types of testing, such as unit testing, integration testing, system testing, and acceptance testing, to verify that the application functions correctly, meets user needs, and adheres to business rules. Testers meticulously create test cases, execute them, and analyze the results to identify any

discrepancies or defects. Proper documentation of the testing process, including test plans, test cases, and test results, is essential for traceability and audit purposes.

FR No:	Test case description	Test case condition	Expected Result	Actual result	Status
FR1	User Registration and Login	Login or registration information of the user.	If a registered user is trying to log into the system, the user should be able to log successfully and go to the home screen without any issue.	After registering or logging in, the user can successfully navigate to the home page.	**Pass**

FR2	Allow users to view SupAut Dashboard details (Students, Staff, Questions, Topics, Question Count Graph and Student Average Graph)	Checking student count and staff members enrolls in and checking for questions and topic count that has added Checking Question count graph and student average graph	Successfully fetching data from database and show the student count, staff count, question count, topic count, question count graph and student average graph	Successfully fetched data from database and showed student count, staff count, question count, topic count, question count graph and student average graph	**Pass**
FR3	Allow users to view Add Question tab and allowing to add Questions to any grade and provide a model answer for the question	Checking if grade, score, chapter name, question, description and model answer have been added. Then after Add Question button should be clicked	Successfully fetching data from database and show added question and model answer on question page.	Successfully fetched data from database and showed added question and model answer on question page.	**Pass**

FR4	Application should store user entered added questions and model answers on Question Page	Checking the entered values of question details, answer details and Activity Status added	Successfully fetching data from add student page and analyzing the Activity Status	Successfully fetched data from add student page and analyzed the Activity Status and added.	**Pass**
FR5	Allow users to store student details on Add Student page	Checking the entered values of student details are valid Add Student button should be clicked	Successfully fetching data from database and show added student details on Add Student page.	Successfully fetched data from database and showed added student details on Add Student page.	**Pass**
FR6	Application should store user entered student details on Student Page	Checking the entered student details are added and valid Go to Page button should be clicked to view student profile page	Successfully fetching data from database and show added student details on Add Student page.	Successfully fetched data from database and showed added student details on Add Student page.	**Pass**

ChatBot Activity					
FR7	Teacher should be able to ask a question.	It is required that the middle school teacher asks a question from the student.	Successfully fetching data from twilio serveless and show on chatbot	Successfully fetched data from twilio serveless and shows on chatbot	**Pass**
FR8	The question should be visualized.	The student should be able to get a visual aid for the question on the request of him/her.	Successfully fetching data from twilio serveless and show on chatbot	Successfully fetched data from twilio serveless and shows on chatbot	**Pass**
FR9	The question should be broken down into understandable parts.	The student should be able to obtain a broken down version of the question on the request of him/her.	Successfully fetching data from twilio serveless and show on chatbot	Successfully fetched data from twilio serveless and shows on chatbot	**Pass**
FR10	The question/answer should be rephrased according to the social	The middle school teacher should be able to view a rephrased answer that is sent by the student.	Successfully fetching data from twilio serveless and show on chatbot	Successfully fetched data from twilio serveless and shows on chatbot	**Pass**

	conversing norms.				
FR11	The student should be able to post an answer/question	It is required that the student posts a question/answer back to the teacher.	Successfully fetching data from twilio serveless and show on chatbot	Successfully fetched data from twilio serveless and shows on chatbot	**Pass**
FR12	Allow users to view Progress of student skills and Student details Creating a monthly IEP report	It is necessary to evaluate how well kids responded to the questions provided by the teachers in order to compile a monthly IEP report. When creating the IEP report, the use of the features visualisation, breaking down, and rephrasing will also be taken into account.	Successfully fetching data that provided on the chatbot and activity on dashboard to generate iep report	Successfully fetched data that provided on the chatbot and activity on dashboard and generated iep report	**Pass**

Table 4: Tabular description of test case scenarios

2.4 Testing Non-Functional Requirements

Non-functional requirements optimize system usability, efficiency, and maintainability. Testing for accuracy, performance, reliability, usability, and scalability will be conducted below.

Test Case No:	Requirements list	Status
NFR1	Scalability	Pass
NFR2	Usability	Pass
NFR3	Accuracy	Pass
NFR4	Reliability	Pass
NFR5	Performance	Pass

Table 5: Non-Functional requirements of Project SupAut

All non-functional requirements, including accuracy, performance, reliability, usability, and scalability, have been thoroughly tested with multiple test cases. Testing results and explanations are provided below, with additional details in the Appendix.

2.4.1 Scalability

Test No:	Test case description	Test case condition	Expected result	Actual result	Status
1.0	Scalability	The code's reusability, structure, comments, and	Effective implementation of coding conventions and structure	Effective implementation of best coding practices.	**Pass**

		adherence to coding standards were evaluated			
1.1	Scalability	Effective organization of folders and appropriate file naming conventions	Maintaining well-organized folder structures and file namings to ensure efficient code management**.**	The files were effectively organized and maintained to facilitate future enhancements, with proper structure and maintenance practices in place.	**Pass**

Table 6: Testing Non-Functional requirement 1

2.4.2 Performance

Test No:	Test case description	Test case condition	Expected result	Actual result	Status
2.0	Performance	The program was loaded and tested on a computer with an Intel Core i5 CPU and 8GB RAM, as per the system specifications.	The program functions smoothly without any noticeable lag or conflicts.	The program operates smoothly without encountering any issues or performance delays, demonstrating	**Pass**

				successful functionality.	
2.1		The concurrent loading of both ML models and their evaluation using varied input data was effectively executed.	Sustaining optimal performance levels within the application.	The application experienced minor performance lag and gaps during execution, but all tasks were completed successfully.	**Pass**
3.0	Performance	To check the ChatBot Activity Application had to run on two mobile phones with different RAM capacities. First one has 4GB RAM and the second one has 2GB RAM.	Same internet connection has been used for both phones. At the same time, both phones were evaluated.	During testing on these various phones, it was discovered that the ChatBot activity performed better on 4GB phone than the 2GB phone.	**Pass**

Table 7: Testing Non-Functional requirement 2

2.4.3 Reliability

The system exhibits high reliability in meeting non-functional requirements, such as performance, usability, and maintainability, ensuring smooth operation and consistent performance.

2.4.3 Accuracy

Accuracy testing is a crucial aspect of evaluating the precision and correctness of predicted outputs. It involves conducting multiple iterations and using diverse datasets to validate the accuracy of the system's predictions. By testing the accuracy of the outputs against expected results, accuracy testing helps identify any discrepancies and ensures that the system is delivering reliable and precise results. This process can involve using different data sets, varying inputs, and repeated testing to continuously improve the accuracy levels of the system's predictions, making it more reliable and trustworthy for its intended purpose. Accuracy testing plays a vital role in ensuring the quality and dependability of a system's performance, providing valuable insights into the system's accuracy and helping in identifying areas for improvement.

2.5 Unit Testing

In the software development process, unit testing is performed by breaking down the application into smaller components, which are then tested individually and independently. This approach, known as unit testing, ensures that each component functions correctly on its own before they are integrated into the larger application. For the SupAut web application, the following table provides further details on how unit testing is conducted.

Description	**Expected Output**	**Actual Output**	**Status**
User Sign Up Page			
User is able to enter the full name, email, password and again password confirmation.	If the given sign-up information is valid, users can sign up to the SupAut Web application successfully.	Sign-Up Completed	**Pass**

After that the user should click the Register button.			
User Login Page			
User can enter the email and password for logging to the application.	User should be able to go to the Dashboard screen.	Went to the Dashboard screen.	**Pass**
If the given information by the user to log to the application is not correct and presses the login button.	If so, the application is shown a proper error mg to the user.	Error message showed.	**Pass**
SupAut Dashboard Page			
User can go to the Dashboard page after logging in.	Dashboard page can be seen.	Dashboard page can be seen.	**Pass**

User can click the Add Question button and go to the Add Question page.	Add Question page should appear.	Add Question page appears.	**Pass**
User can click the Add Student option and go to the Add Student page.	Add Student page should appear.	Add Student test page appears	**Pass**
User can move the cursor around on Question Count Graph and Student Average Graph.	On Question Count Graph Topics and Question count should appear. On Student Average Graph Month and Grade should appear	On Question Count Graph Topics and Question count appears. On Student Average Graph Month and Grade appears.	**Pass**
QUESTION (Add Question Page, Question Page)			
User can enter the values for add a question and click the Add Question button.	Question should be added and question details and answer details should be appear on Question Page.	Successfully Question added and question details and answer details appeared on Question Page.	**Pass**

If the given information by the user is valid.	If so, Question, Question Detail, Answer Detail and Activity status should appear	Successfully Question, Question Detail, Answer Detail and Activity status appeared.	**Pass**
If the user did not enter the values for the relevant input fields and pressed the Add Question button.	If so, a proper error is displayed to the user separately for the input fields.	Error message showed.	**Pass**
STUDENT (Add Student Page, Student Page)			
After the user presses Add Student Page user can enter the values for add a student and click the Add Student button.	User should be able to see the added student on the Student Page.	Successfully Added student appears on Student Page.	**Pass**
If the given information by the user is correct	If so, Student Details, Care Giver Contact, Care Giver Email, Care Giver Details and Student Profile Page should appear	Successfully Student Details, Care Giver Contact, Care Giver Email, Care Giver Details and	**Pass**

		Student Profile Page appeared.	
If the user did not enter the values for the relevant input fields and pressed the Add Student button.	If so, a proper error is displayed to the user separately for the input fields.	Error message showed.	**Pass**
Student Profile Page (Go to Page)			
After the user presses Go to Page user can view Student Details and Progress of Student Skills	User should be able to see the added student details on the Go to Page and Progress of Students Skills as a Pie chart showing their Creativity, Time Management, Logical Skills and Visual Skills.	Successfully added student details appears on the Go to Page and shows the Progress of Students Skills as a Pie chart showing their Creativity, Time Management, Logical Skills and Visual Skills.	**Pass**
In Student Details user can click the Download IEP Report and Create IEP Report	User should be able to download the automated IEP report that included their skill analyze during their assessments time, activity on the ChatBot and	Successfully downloaded the IEP Report and can Create and edit the IEP Report By Clicking Create IEP Report	**Pass**

	activity through the Dashboard.		

Table 8: Unit testing for SupAut Pages

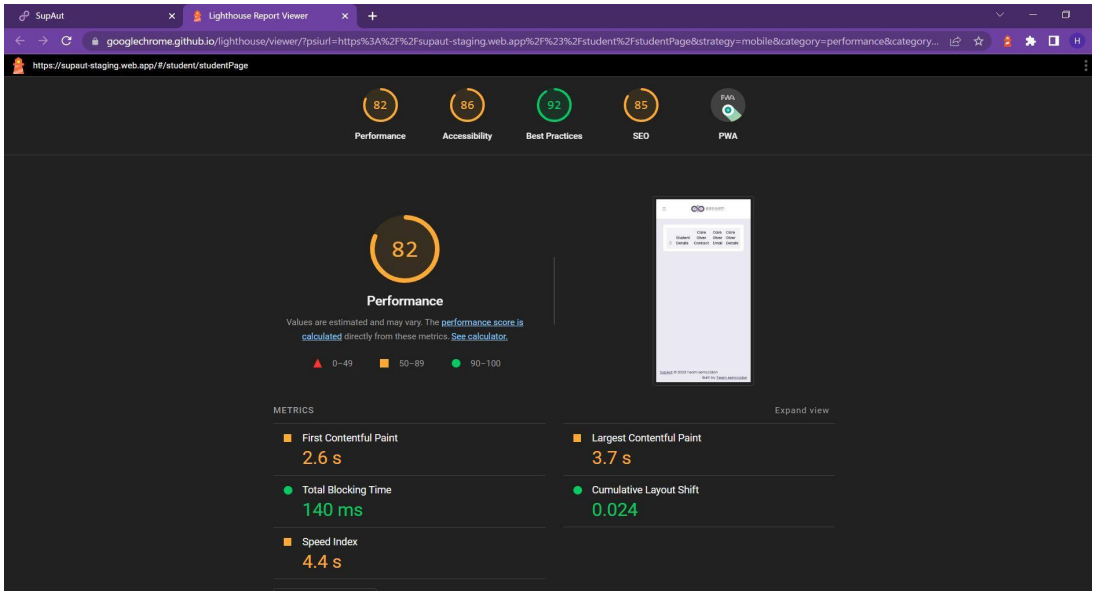
2.6 Performance Testing

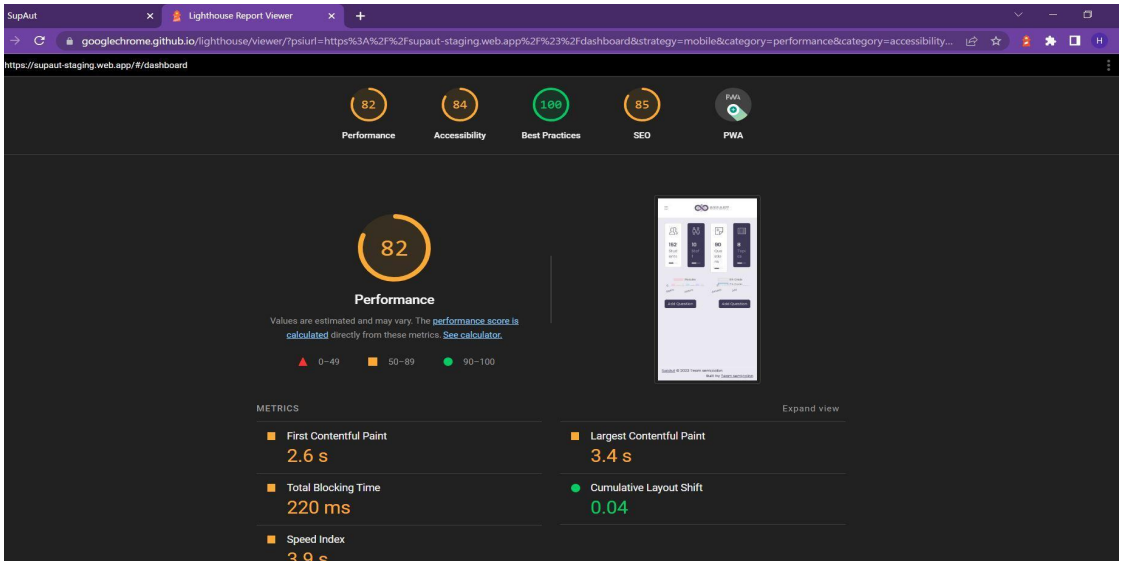
Performance testing is a type of non-functional testing that focuses on evaluating how well a program performs under a specific load, including factors such as speed, scalability, and responsiveness. This testing approach also considers the performance of various hardware modules to provide accurate results. The Project SupAut web application underwent testing on various web browsers such as Google Chrome, Firefox, Microsoft Edge, and iOS. The performance of each page was assessed using the Lighthouse testing Chrome extension, and the results are included in Appendix section B.2.

Test No:	**Test case description**	**Test case condition**	**Expected result**	**Actual result**	**Status**
1	Performance	Performance tested on popular web browsers during the initial load.	Displays optimal performance with fast initial loading.	The program functions effectively, and the user interface (UI) is responsive and operates smoothly.	**Pass**

2		The performance of the ChatBot model was evaluated based on query entries and request speeds.	The ChatBot model performs efficiently with a delay of less than 10 seconds in processing queries and requests.	The ML model demonstrates good performance in detecting with minimal delays.	**Pass**

Table 9: Testing Performance of the application





2.7 Usability Testing

Usability testing evaluates the user-friendliness and effectiveness of a system, including the UI design and user experience. The SupAut progressive web application was designed with a minimal yet effective GUI, catering to users of all ages. The application is fully responsive, functioning seamlessly across standard mobiles and tablets. This information is detailed in the report to highlight the user-centric approach of the SupAut project.

Test No:	Test case description	Test case condition	Expected result	Actual result	Status

1	Usability	User friendly Design	Project SupAut's UI/UX is simple, accessible, and user-friendly. It caters to users of all ages, is fully responsive, and maintains effective GUI design for seamless interaction.	The UI and UX of the application are commendable, providing a user-friendly experience for all users. Users find the application easy to use and navigate.	**Pass**
2		Responsive Design	Responsive for multiple devices	Project SupAut was designed to be responsive across various devices and screen sizes, ensuring optimal performance and user experience.	**Pass**

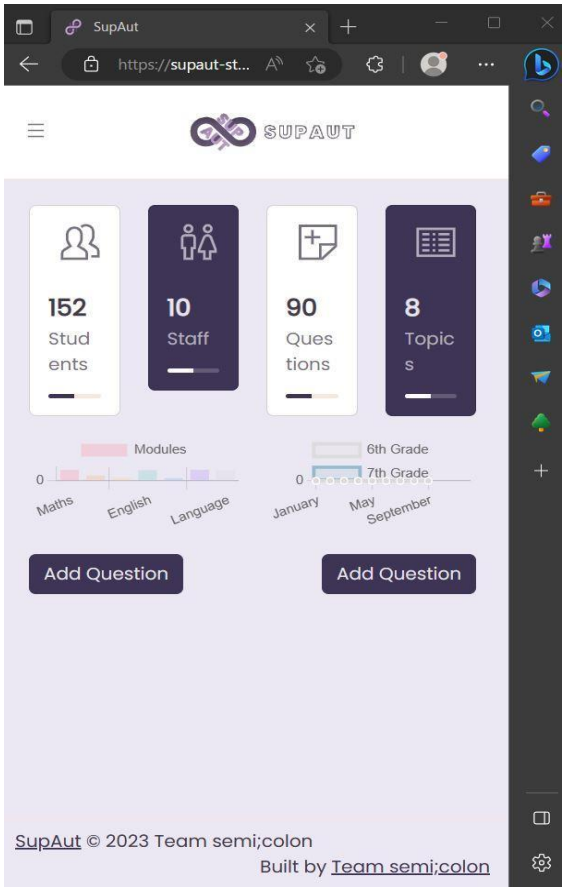
Table 10: Testing Usability of the application

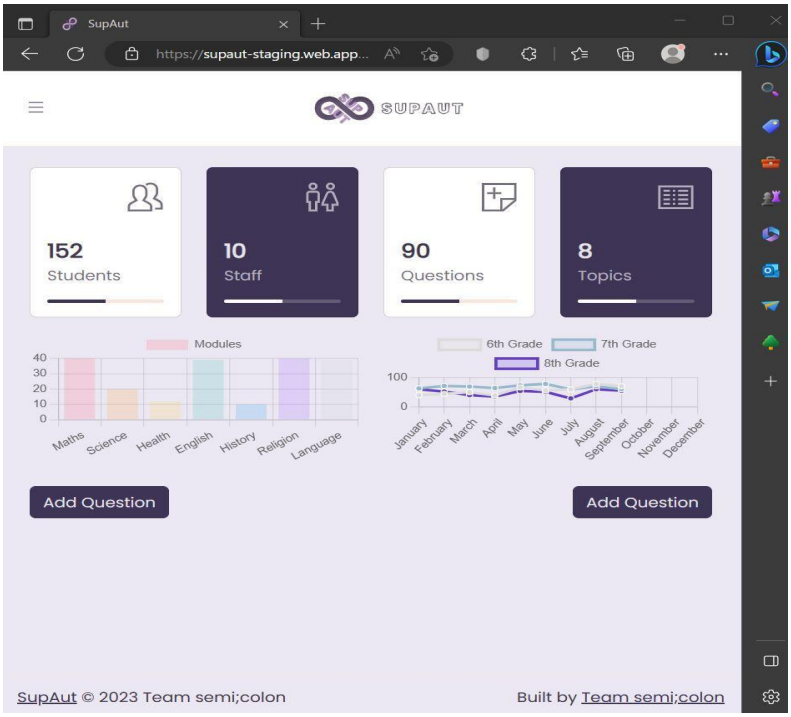
2.7.1 User Friendly UI/UX - Usability Testing

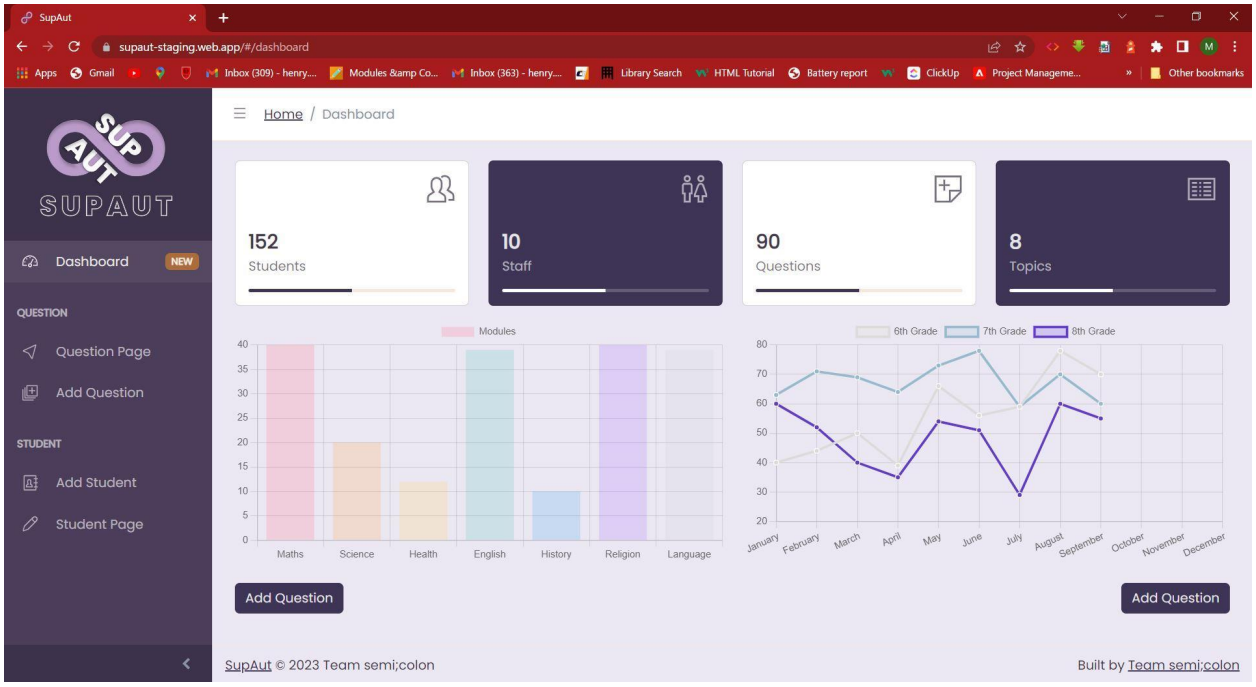
Usability testing, a non-functional testing approach, was conducted to assess the effectiveness of the UI design and UX of the SupAut application. This testing focused on evaluating how easily users could interact with the application, ensuring a seamless user experience. With a target audience of students and teachers, the application was designed with simplicity in mind, incorporating light purple color variants and white color schemes to make pages visually appealing. Additionally, a chatbot was included to enhance user-friendliness. The application performed well on desktop computers.

2.7.2 Responsiveness - Usability Testing

The performance of Project SupAut application on various devices with different screen sizes is depicted in the following results.







2.8 Compatibility Testing

Compatibility testing is an important non-functional testing approach used to ensure that the SupAut web application functions properly across different devices and networks. This testing helps to verify the application's compatibility with various browsers, such as Chrome, Firefox, Microsoft Edge, Brave, and mobile browsers like Samsung Internet Browser and iOS. It ensures that all functionalities of the application work well across these browsers, ensuring the system's compatibility. The responsive design ensures smooth functionality across different screen sizes, as confirmed in Appendix Section B.3. The system's compatibility has been thoroughly verified.

2.9 Chapter Summary

The purpose of this chapter is to provide an overview of the testing phase of the SupAut application. It includes discussions on testing functional and non-functional requirements, with specific focus on performance testing, usability testing, and compatibility testing. Additionally, detailed information about unit testing is also provided in this chapter.

CHAPTER 3: EVALUATION

3.1 Chapter Overview

The previous chapter covered all the testing methods, techniques and requirements of the SupAut project. This chapter will guide you through the evaluation process for this particular project. The use and importance of quantitative and qualitative data, outcome-based evaluation and self-evaluation are discussed in detail.

3.2 Evaluation Methods

For this particular project, the team considered a qualitative evaluation by domain experts as an option. as it is important to study the project with the experts to know if the project would really affect a specific area. As a result, several domain experts were contacted and asked for their opinion and feedback on the SupAut project by submitting a video demonstration of the software. In addition, a dummy demonstration was sent to computer science students to get their feedback on SupAut. Meanwhile, team members conducted a self-evaluation.

3.3 Quantitative Evaluation

Quantitative Evaluations were conducted for the Implementation of Chatbot in Twillio. Below Charts show the Deliverability of the Messages. Kin of an evaluation if the chatbot conversation is proper and stable.

Figure 27: Messages by Status

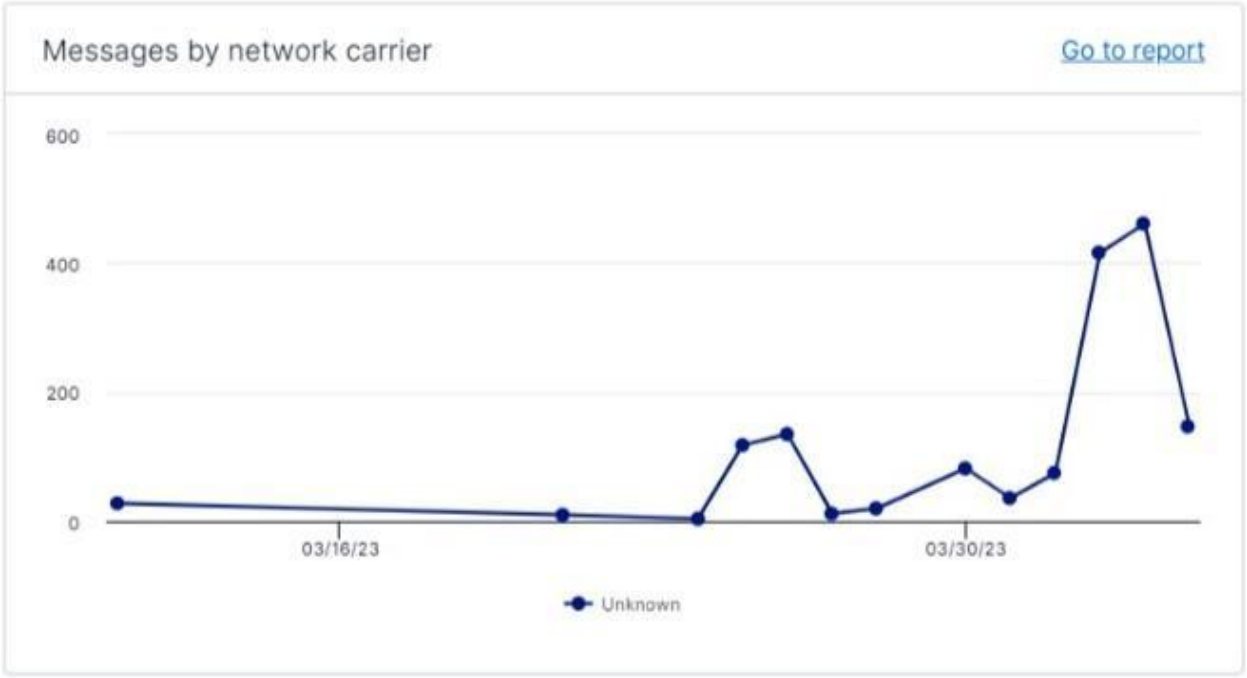


*Figure SEQ Figure * ARABIC 28: Messages by Network Carrier*

3.4 Qualitative Evaluation

The prototype was shown to end-users and domain experts. They went through it and had provided us with feedback.

3.4.1 Feedback from End Users

A Survey/Questionaire was made by the team and was released to the end users who are undergraduates and gotten feedback for the project SupAut. The responses of the Survey is mentioned under Appendix (relevant appendix title) for better clarity. But overall the feedbacks were adapted in good ways. and the implemented ideas and how they were implemented convinced the end users. For others They don't like some components that can be implemented even better and more precisely.

3.4.2 Feedback from Domain Experts

Ms. Shahada Rifkhan SEN Teacher, Psychology Undergraduate, SEN teacher at Alethea International School

“I really like what you've done to include SEN students into your project SupAut. I find it very organised, and would make it very convenient for the teachers as well as some students. However I'd like to tell ask you about, what about students who can't write, nor spell complete sentences, because those are certain things that SEN students struggle with? Is there maybe a microphone that would translate audio into text, so you could be completely inclusive. That was my only concern, other than that, I felt everything else to be very well done.”

3.5 Self Evaluation

Zainab Fahim SupAut Contributor

The project was implemented with consideration to minimize security and privacy risks while ensuring a great user experience. The chatbot's flow and usability were well-executed, and the deployment was successful. The dashboard has basic functional features that work well, although there could be some improvements to make the question-making process easier. As suggested by the domain expert, implementing voice functionality would improve the chatbot's usability even

further. The GitHub repository was well-maintained, though there could have been more discussions within GitHub to improve the project's quality.

3.6 Chapter Summary

The evaluation chapter covers different methods of evaluating the effectiveness and impact of a product or service. Both quantitative and qualitative approaches will be explored, with a quantitative evaluation involving the analysis of numerical data and a qualitative evaluation using non-numerical data to gain a deeper understanding of user experiences. Feedback from experts and users is also seen as a valuable tool for improvement, and author self-evaluation is highlighted as an essential part of the process. Overall, this chapter highlights the importance of evaluation in product development and provides an overview of effective evaluation strategies.

CHAPTER 4: CONCLUSION

4.1 Chapter Overview

An evaluation of the entire project is included in this section, which also acts as its last chapter. The projects advantages and disadvantages are discussed with compared to other systems.The project's goals and objectives, new knowledge and abilities acquired during the project, and potential improvements are all mentioned in the report.

4.2 Achievements of Aims and Objectives

To design and develop an effective method for tracking and documenting the progress of autistic students in traditional classroom settings, with a focus on social, creative, and communication skills,to test and evaluate the potential of assistive technology to support language and communication development in autistic children.

The research aim had two main components. Identifying effective methods and tools for tracking and documenting the progress of Autistic students and the other one which is exploring the potential of assistive technology to help autistic children with language and communication challenges to complete homework assignments independently. Fortunately the development team was able to make both these aims a success.

Features accomplished	Features unable to accomplish
Prepare IEP report	Automate IEP report.
Organise child progress in a dashboard	Teach social conversing standards.
Organise question and answer reviews systematically	
Conversation assignment helper	

Breaking down instructions	
Visualizer	

Table 11: Achievements of aims and objectives

Objective	Status	Description	Evidence
OBJ1	Completed	The problem's background, the goal and objectives, the project's scope, the necessary resources, and the project prototype's features are all explained.	Chapter 1 – Introduction
OBJ2	Completed	In this section, existing functions are discussed. There is a chapter that looks at the advantages and disadvantages of each and recommends the best option for this project.	Chapter 2 – Literature Review
OBJ3	Completed	The project management chapter describes the methods used for this project. This chapter also includes a work breakdown structure (WBS), a Gantt chart, and an activity timeline.	Chapter 3 - Methodology
OBJ4	Completed	In addition to outputs like use case diagrams, use case descriptions, context diagrams, domain models, and functional and non-functional needs and their priority levels, this chapter covers a wide range of requirements and approaches.	Chapter 4 – SRS
OBJ5	Completed	The social, legal, moral, and professional difficulties facing Project SupAut are described.	Chapter 5 – SLEP
OBJ6	Completed	The architecture and design of Project SupAut are discussed in this chapter. High-level architectural designs, class diagrams, sequence diagrams, activity diagrams, and wireframes are all included in this	Chapter 6 – Design

		chapter.	
OBJ7	Completed	The implementation of Project SupAut is covered in this chapter, along with the challenges that were faced and how they were resolved, as well as technical information regarding the prototype and the various construction methods that were employed.	Chapter 7 - Implementation
OBJ8	Completed	This chapter describes Project SupAut's testing. The functional and nonfunctional requirements were clearly laid forth. This chapter covers unit testing, performance testing, usability testing, and compatibility testing.	Chapter 8 – Testing
OBJ9	Completed	This project's evaluation procedure is now over. Qualitative, quantitative, and self-evaluation are all discussed.	Chapter 9 – Evaluation
OBJ10	Completed	It outlines the difficulties encountered when creating this product, as well as potential areas for development.	Chapter 10 – Conclusion

Table 12: Objectives Table

4.3 Limitations of the Research

From the original description of the problem through the completion of the final report, numerous restrictions and challenges were faced throughout the project. While most of them were successfully completed, some others required more work. The following are some of the main problems and restrictions that the project ran across.

- Limitation on the number of references

- Lack of resources in text classification ML researchers and models
- Disability to perform the automation of the IEP report

Lack of study materials and datasets for the ML model, as well as the developers' lack of knowledge in this area, led to little progress being made in the machine learning part despite many tries.

4.4 Future enhancements

The system has been created to the desired standard, although some of the parts could still use improvement. Although it might be more customized, the system performs as well as or better than comparable systems currently on the market.The developers can complete the ML aspect of the project in the future and also add the features like the automation of the IEP report and teaching social conversing standards

4.5 Extra work (Competitions, research papers, etc)

4.6 Concluding remarks

This research project presents a solution to the problem of autistic students facing different challenges in classroom settings in the areas of social,creative, and communication skills. Also the challenges faced by teachers in documenting the progress of these students. The solutions, technologies, techniques, and methods used by Project SupAut are also addressed in-depth with clear explanations. Some of the most crucial elements of the project were completed using the design. The prototype was thoroughly tested to ensure that it satisfied the necessary requirements. While working on this project, a number of novel techniques, theories, technologies, and procedures were found. Even though there were many restrictions and challenges, the team SemiColon was able to complete the majority of the crucial aspects of Project SupAut’s implementation.

REFERENCES

Anoyiannakis, K. (2013). Using technology to support individuals with ASD: A review of the literature. ScholarWorks@GVSU. Available from https://scholarworks.gvsu.edu/honorsprojects/203/?utm_source=scholarworks.gvsu.edu%2Fhonorsprojects%2F203&utm_medium=PDF&utm_campaign=PDFCoverPages [Accessed 10 January 2023].

Colab.research.google.com. n.d. Google Colaboratory. [online] Available from: Welcome To Colaboratory - Colaboratory (google.com)[Accessed 3 March 2023].

Gómez-Marí, I., Sanz-Cervera, P. and Tárraga-Mínguez, R. (2022). Teachers' attitudes toward autism spectrum disorder: A systematic review. Education Sciences, 12 (2), 138. Available from https://doi.org/10.3390/educsci12020138 [Accessed 9 January 2023].

Ghanouni, P. et al. (2019). The use of technologies among individuals with autism spectrum disorders: Barriers and challenges. Journal of Special Education Technology, 35 (4), 286–294. Available from https://doi.org/10.1177/0162643419888765 [Accessed 9 January 2023].

Node.js (n.d.) Getting Started Guide. [online] Available from : https://nodejs.org/en/docs/guides/getting-started-guide/ [Accessed 23 Februaryl 2023].

React (n.d.) Getting Started. [online] Available from : https://legacy.reactjs.org/docs/getting-started.html [Accessed 4 February 2023].

WhatsApp Business API (no date). Twilio. Available from https://www.twilio.com/whatsapp [Accessed 9 January 2023].

APPENDIX

Appendix - Section A - Implementation

Appendix Section A.1 - Backend Implementation

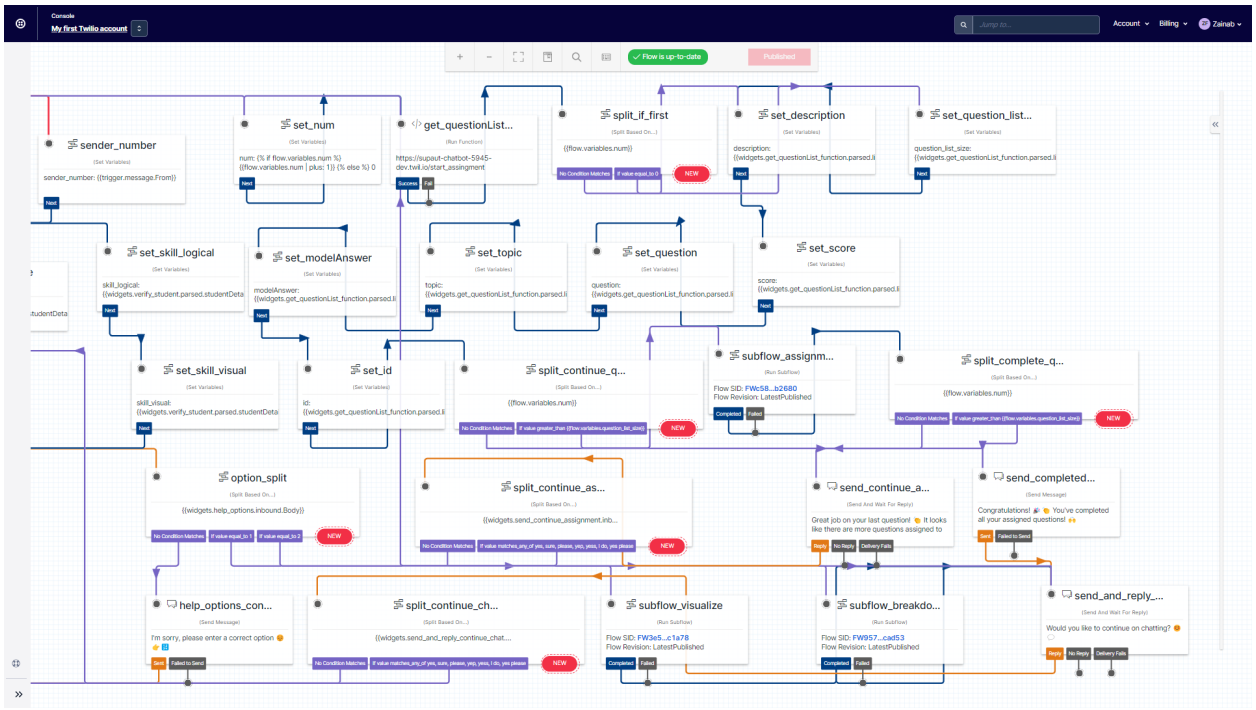


Figure 29: Main SupAut flow (Right Zoomed)

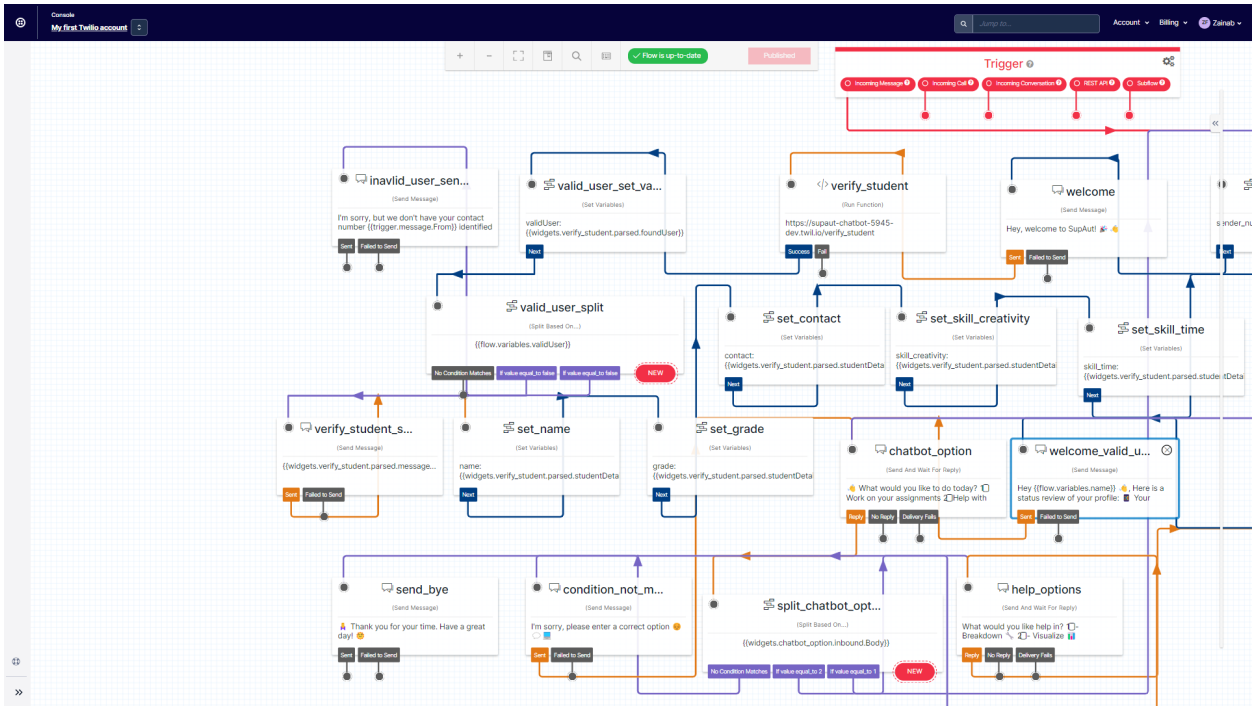
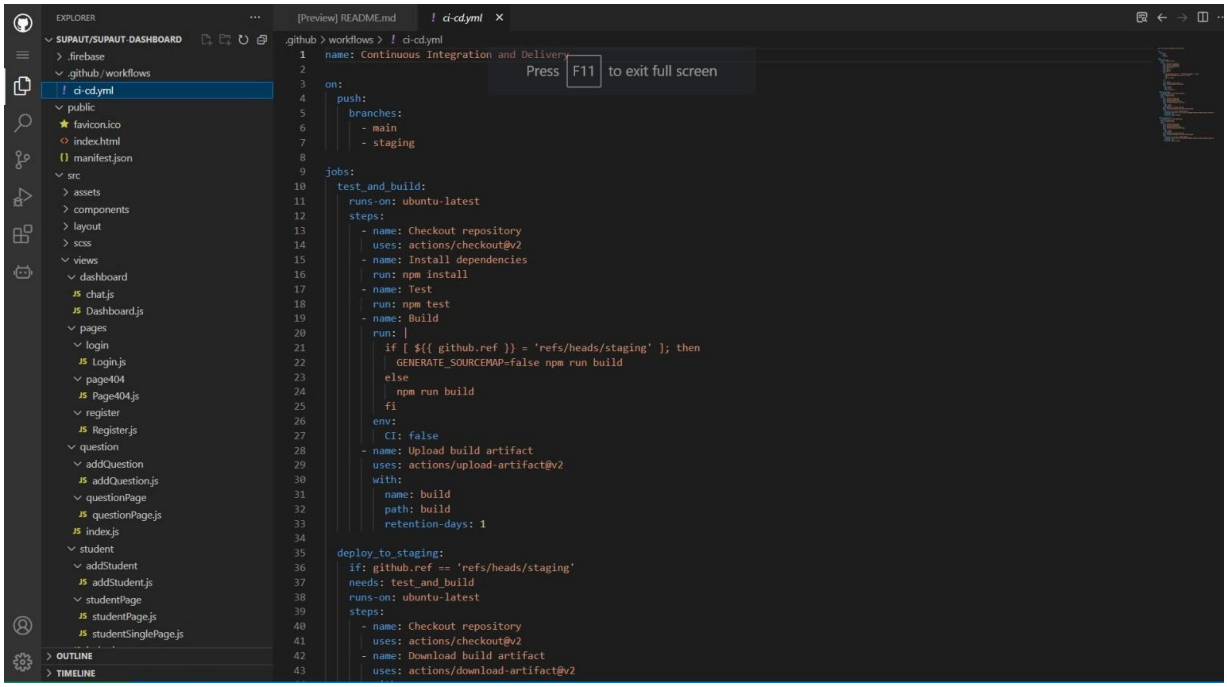
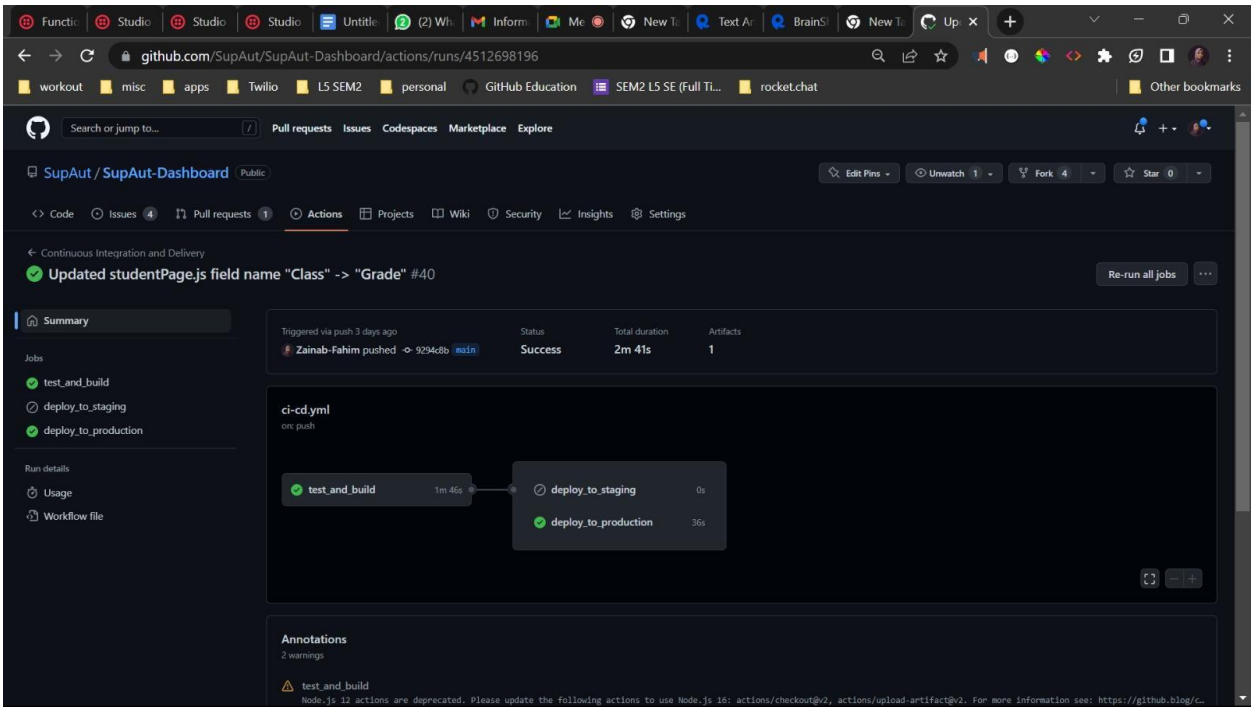
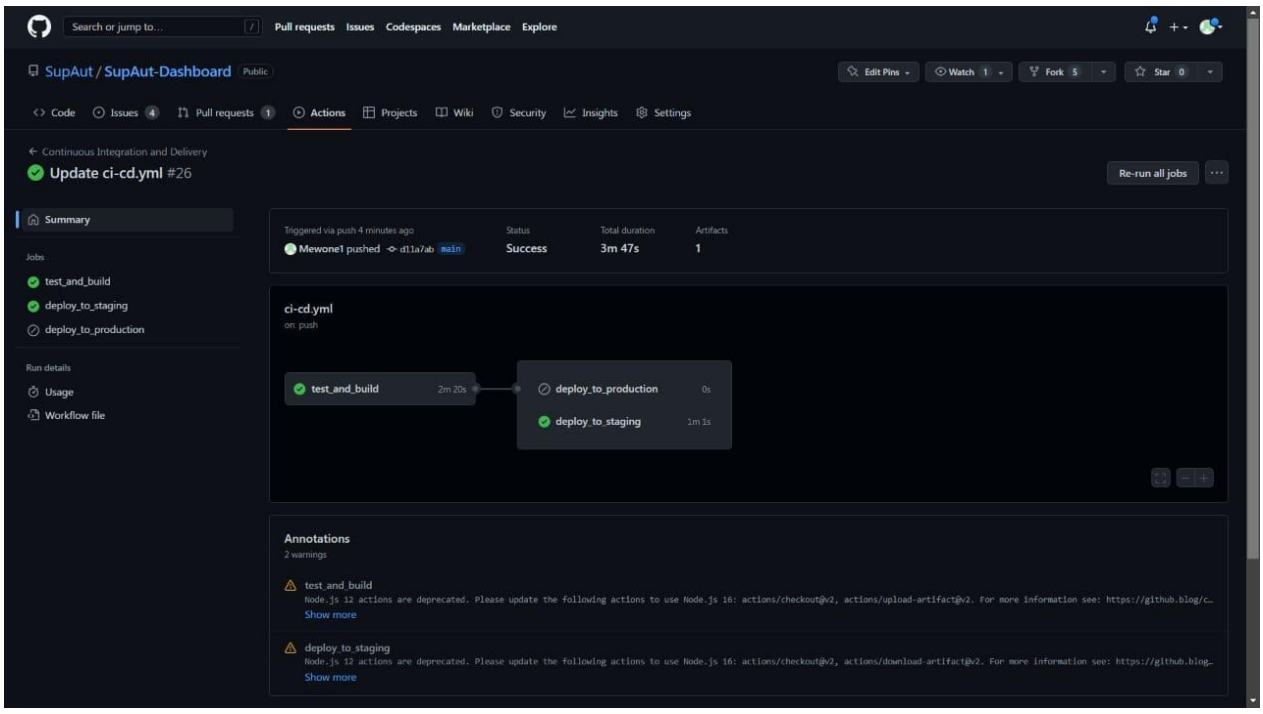


Figure 30: Main SupAut flow (Left Zoomed)

Appendix Section A.2 - CI/CD pipeline codes




github.com/SupAut/SupAut-Dashboard/actions/runs/4512698196
SupAut / SupAut-Dashboard
Public
Code
Issues 4
Pull requests 1
Actions
Projects
Wiki
Security
Insights
Settings
Continuous Integration and Delivery
Updated studentPage.js field name "Class" -> "Grade" #40
Re-run all jobs
Summary
Jobs
test_and_build
deploy_to_staging
deploy_to_production
Run details
Usage
Workflow file
Triggered via push 3 days ago
Zainab-Fahim pushed 9294c8b main
Status
Success
Total duration
2m 41s
Artifacts
1
ci-cd.yml
on: push
test_and_build 1m 46s
deploy_to_staging 0s
deploy_to_production 36s
Annotations
2 warnings
test_and_build


SupAut / SupAut-Dashboard
Public
Edit Pins
Watch 1
Fork 5
Star 0
Code
Issues 4
Pull requests 1
Actions
Projects
Wiki
Security
Insights
Settings
Continuous Integration and Delivery
Update ci-cd.yml #26
Re-run all jobs
Summary
Jobs
test_and_build
deploy_to_staging
deploy_to_production
Run details
Usage
Workflow file
Triggered via push 4 minutes ago
Mewone1 pushed d11a7ab main
Status
Success
Total duration
3m 47s
Artifacts
1
ci-cd.yml
on: push
test_and_build 2m 20s
deploy_to_production 0s
deploy_to_staging 1m 1s
Annotations
2 warnings
test_and_build
Show more
deploy_to_staging
Show more

Appendix section A.3 - Firebase hosting

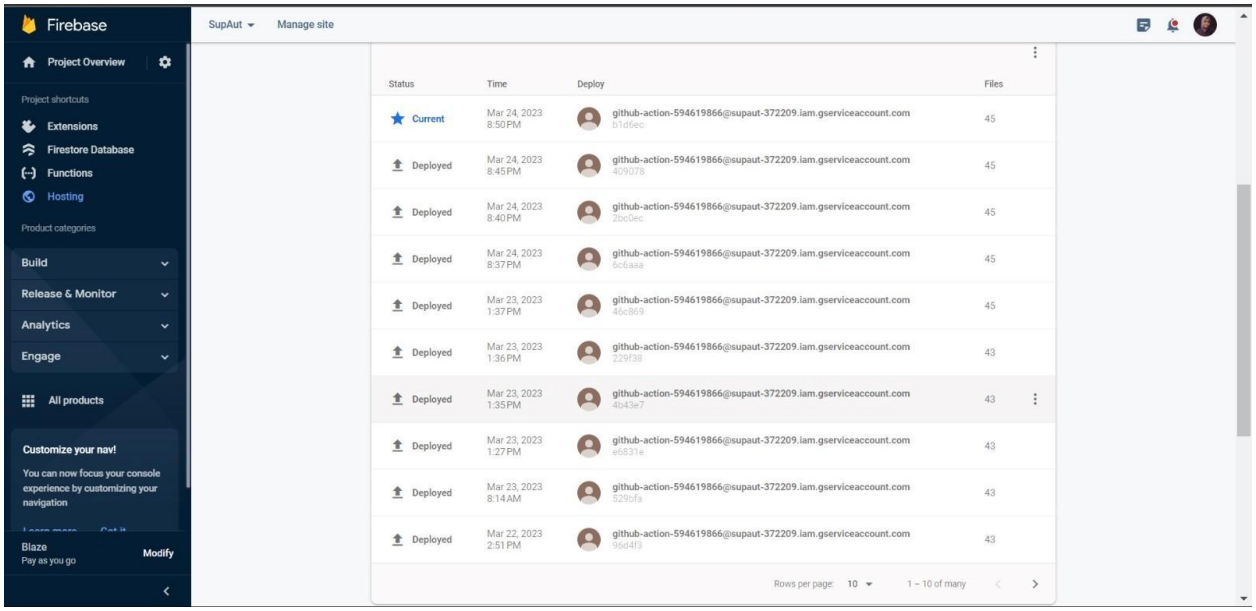
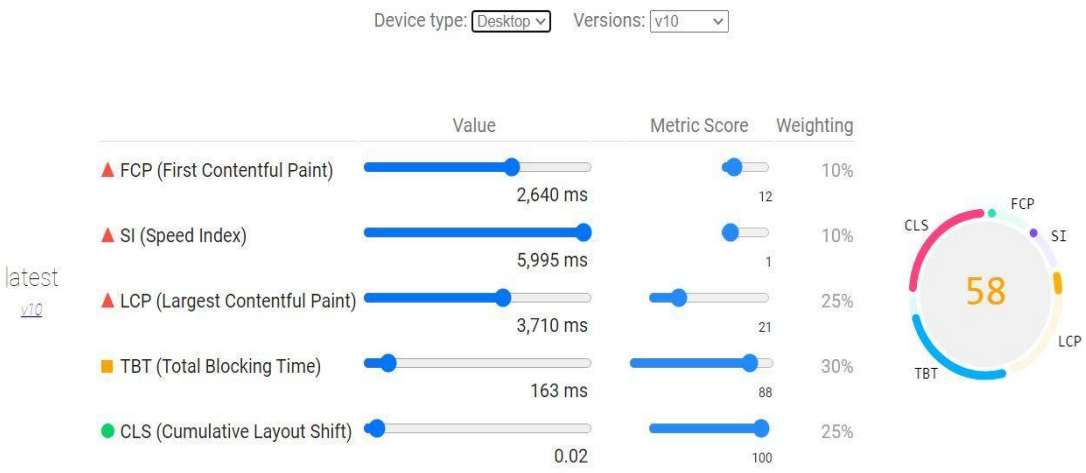
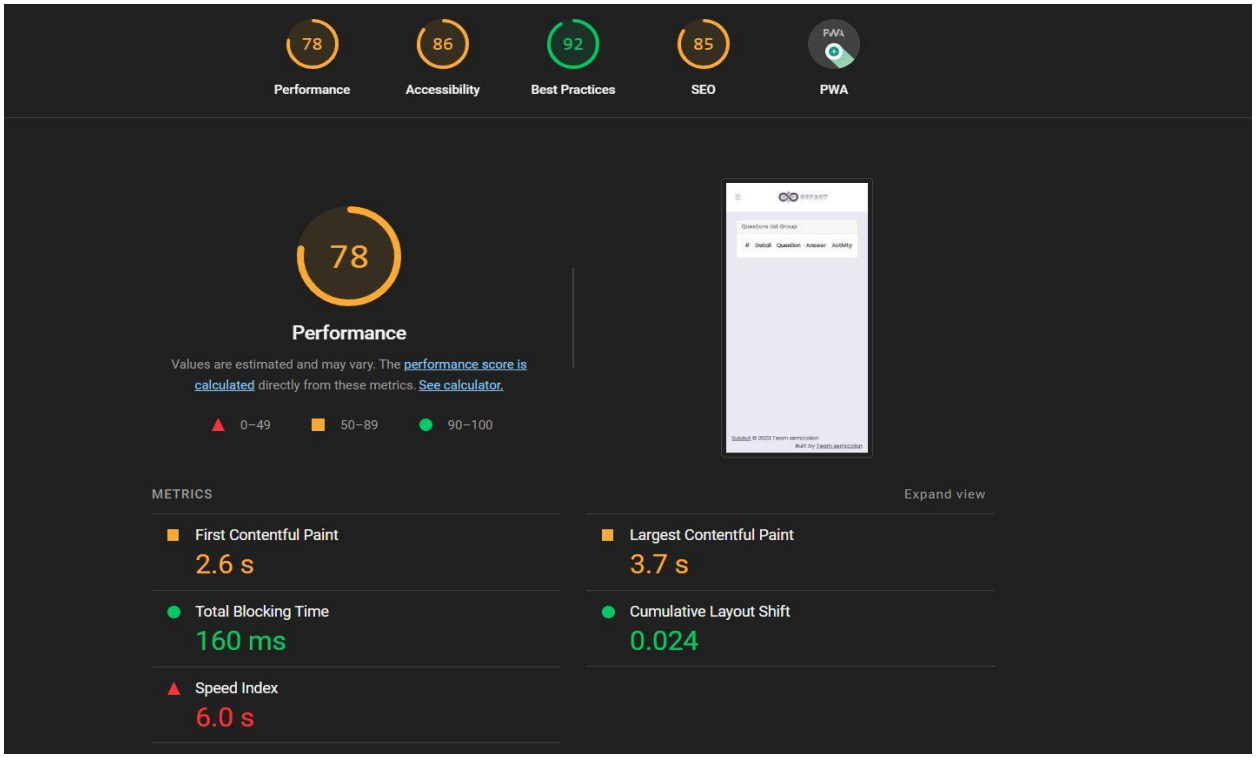


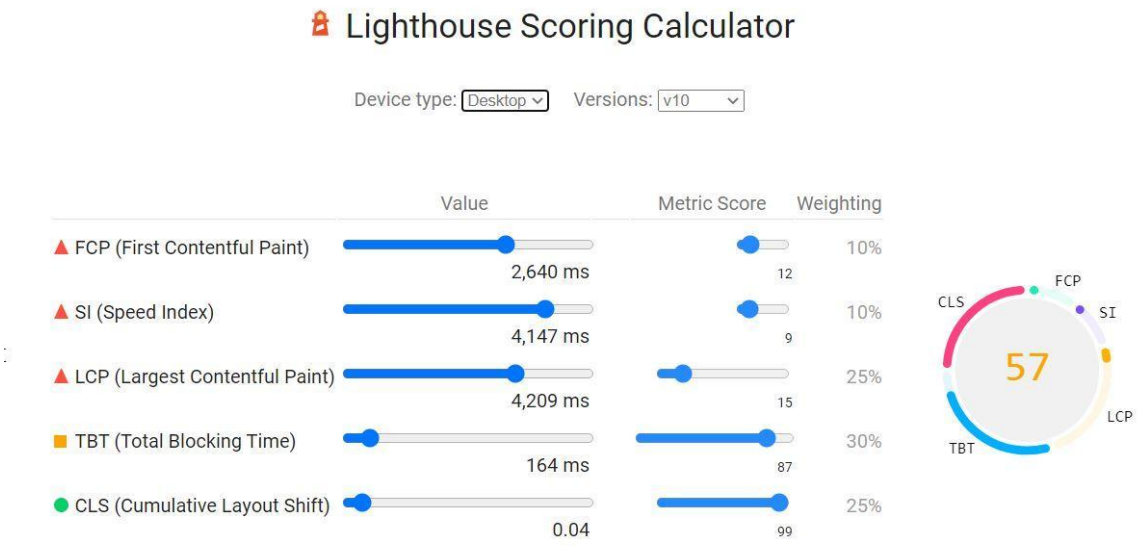
Figure: Successful deployment of the web application in Firebase.

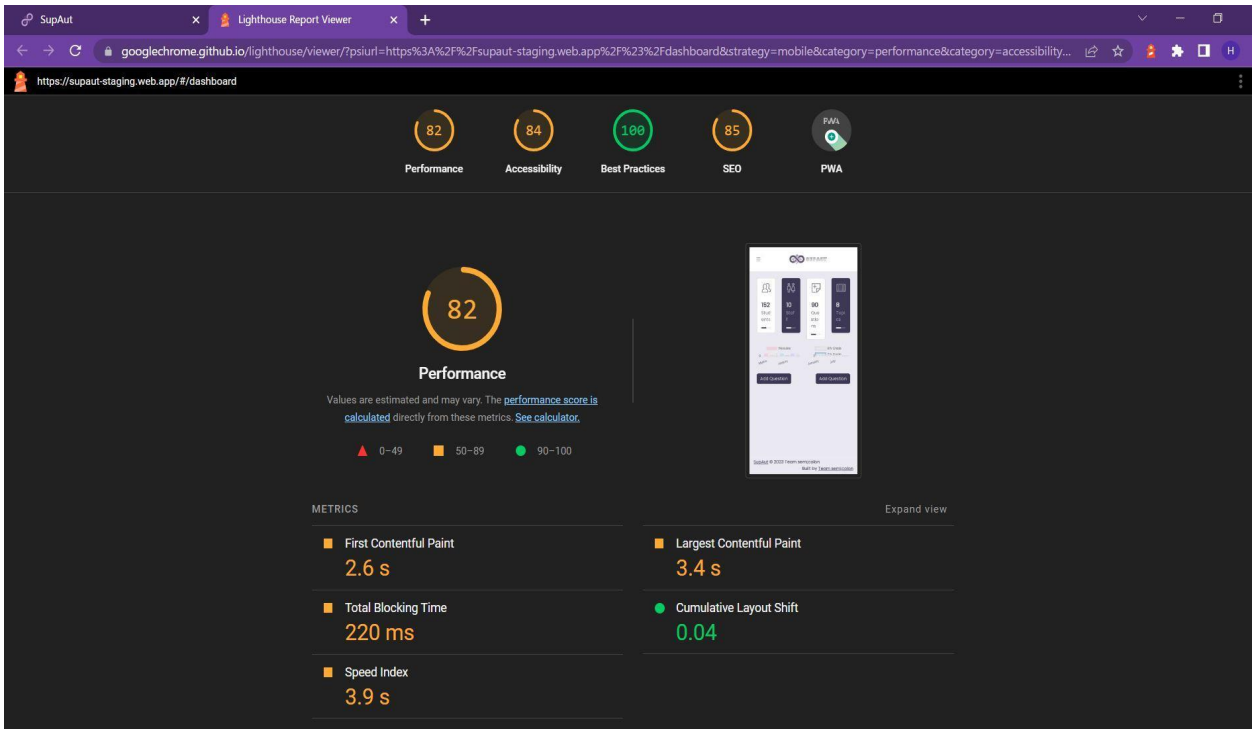
Appendix - Section B - Testing

Appendix Section B.1 - Performance Testing




78
Performance
86
Accessibility
92
Best Practices
85
SEO
PWA
78
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49
50–89
90–100
METRICS
Expand view
First Contentful Paint
2.6 s
Largest Contentful Paint
3.7 s
Total Blocking Time
160 ms
Cumulative Layout Shift
0.024
Speed Index
6.0 s


Lighthouse Scoring Calculator
Device type: Desktop
Versions: v10
Value
Metric Score
Weighting
FCP (First Contentful Paint)
2,640 ms
12
10%
SI (Speed Index)
4,147 ms
9
10%
LCP (Largest Contentful Paint)
4,209 ms
15
25%
TBT (Total Blocking Time)
164 ms
87
30%
CLS (Cumulative Layout Shift)
0.04
99
25%
FCP
SI
CLS
57
LCP
TBT


SupAut
Lighthouse Report Viewer
https://supaut-staging.web.app/#/dashboard
82
Performance
84
Accessibility
100
Best Practices
85
SEO
PWA
82
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49
50–89
90–100
METRICS
Expand view
First Contentful Paint
2.6 s
Largest Contentful Paint
3.4 s
Total Blocking Time
220 ms
Cumulative Layout Shift
0.04
Speed Index
3.9 s

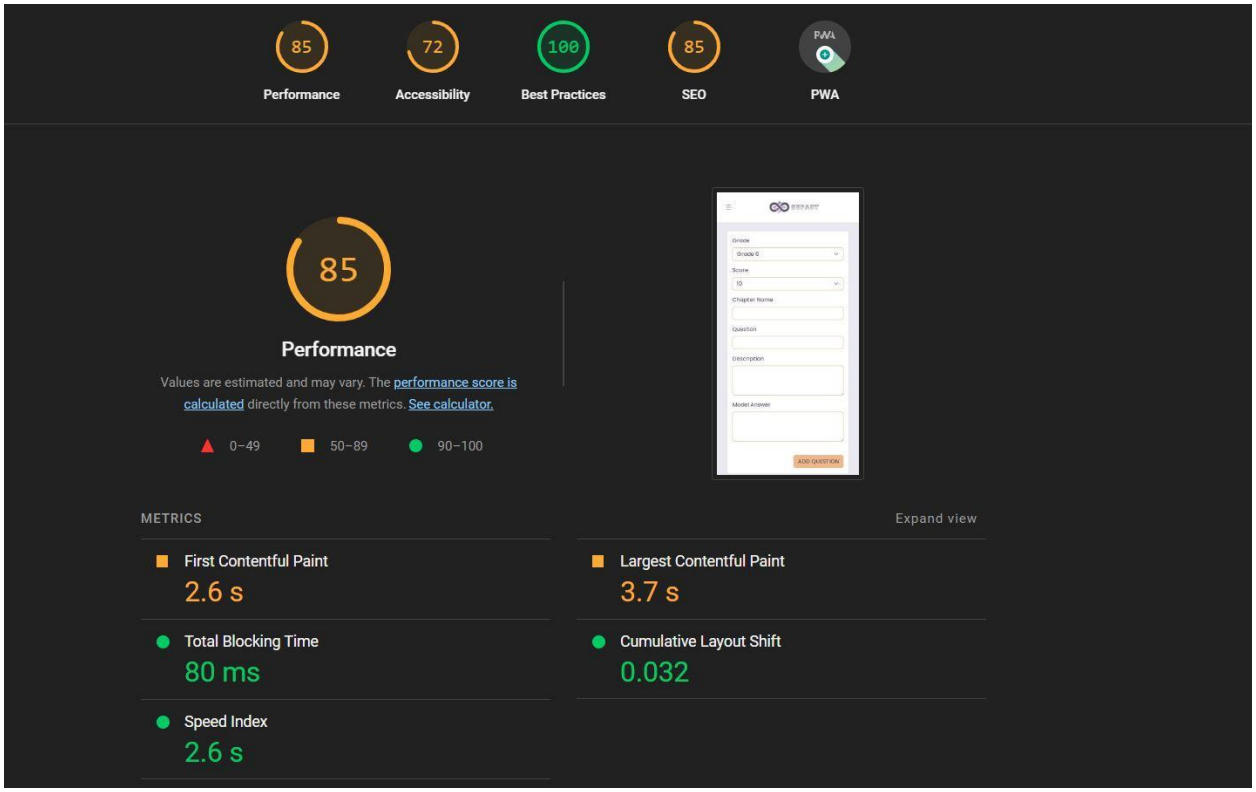


Figure 39: Performance testing of Add Question page

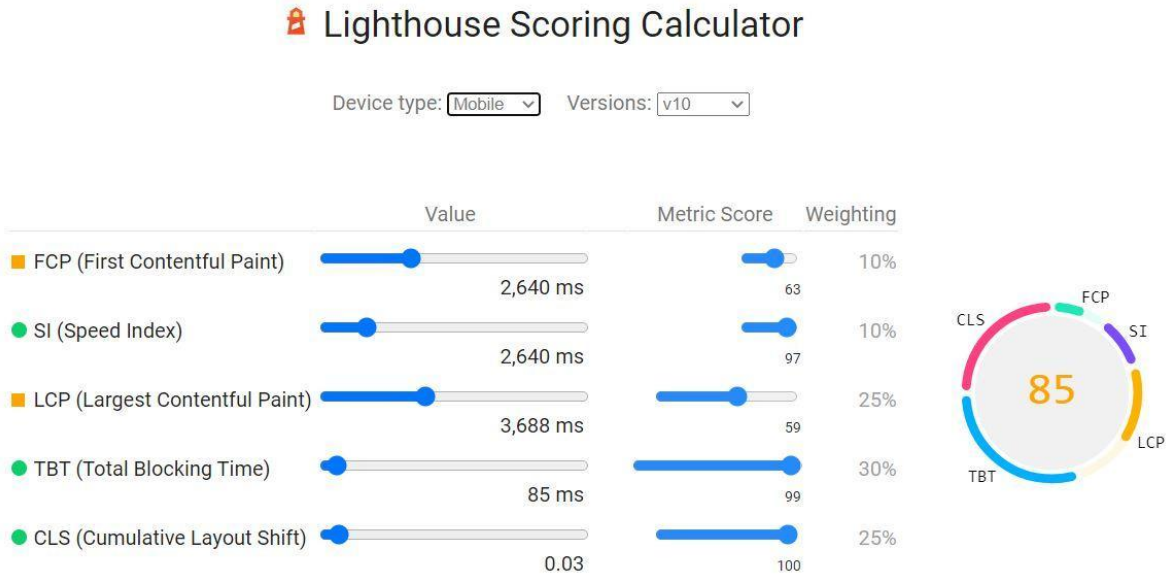


Figure 40: Performance Calculation of Dashboard page in Mobile Version

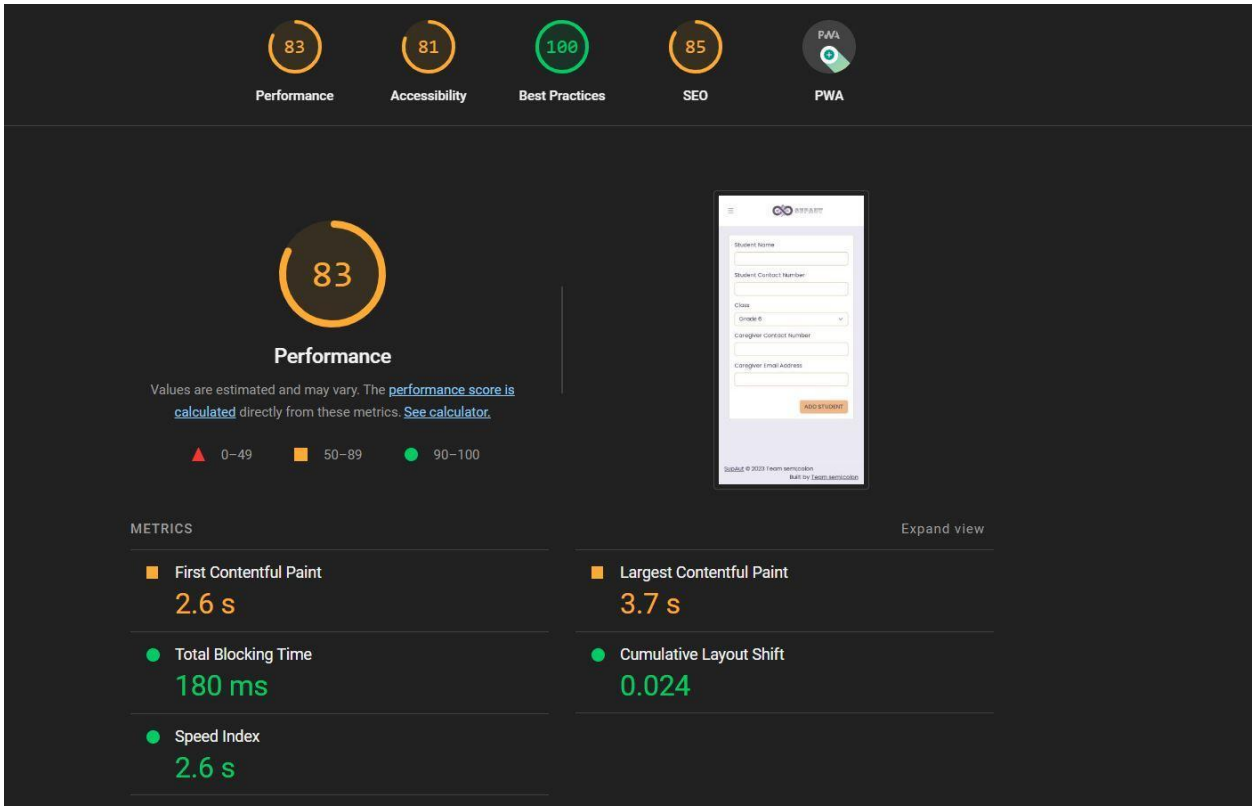


Figure 41: Performance testing of Add Student page

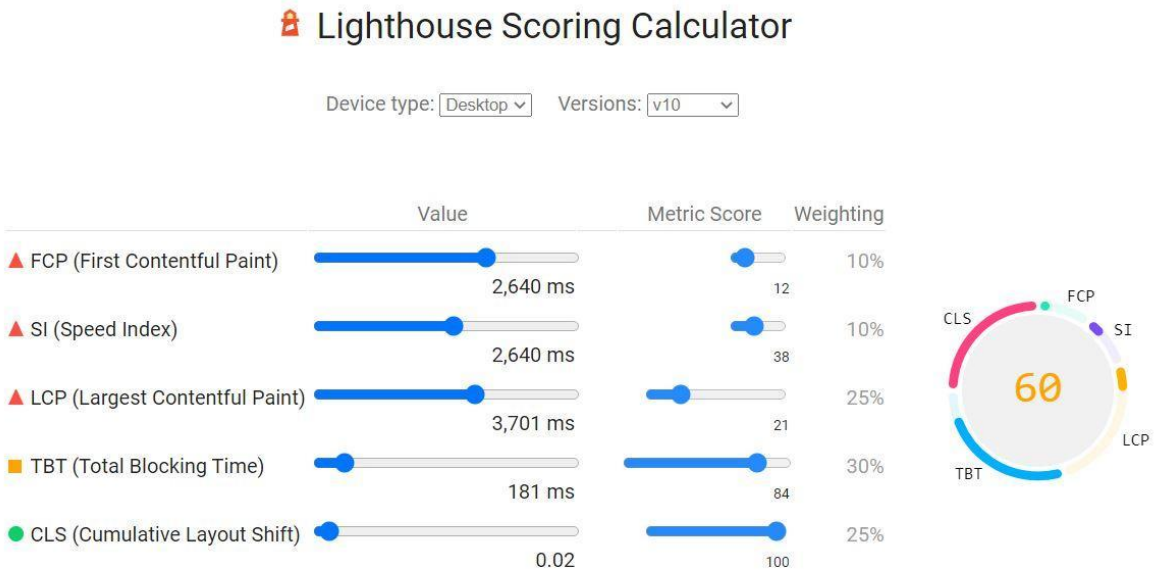


Figure 42: Performance Calculation of Add Student page in Dekstop Version

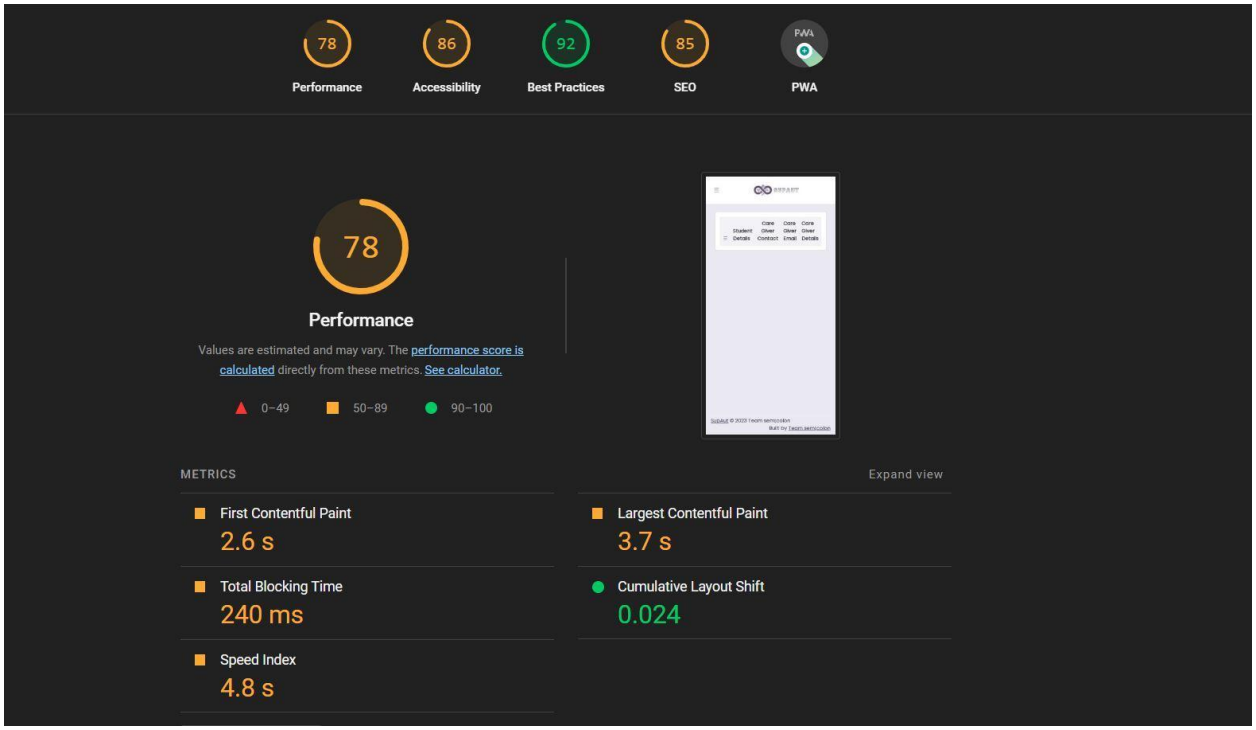


Figure 43: Performance testing of Student page

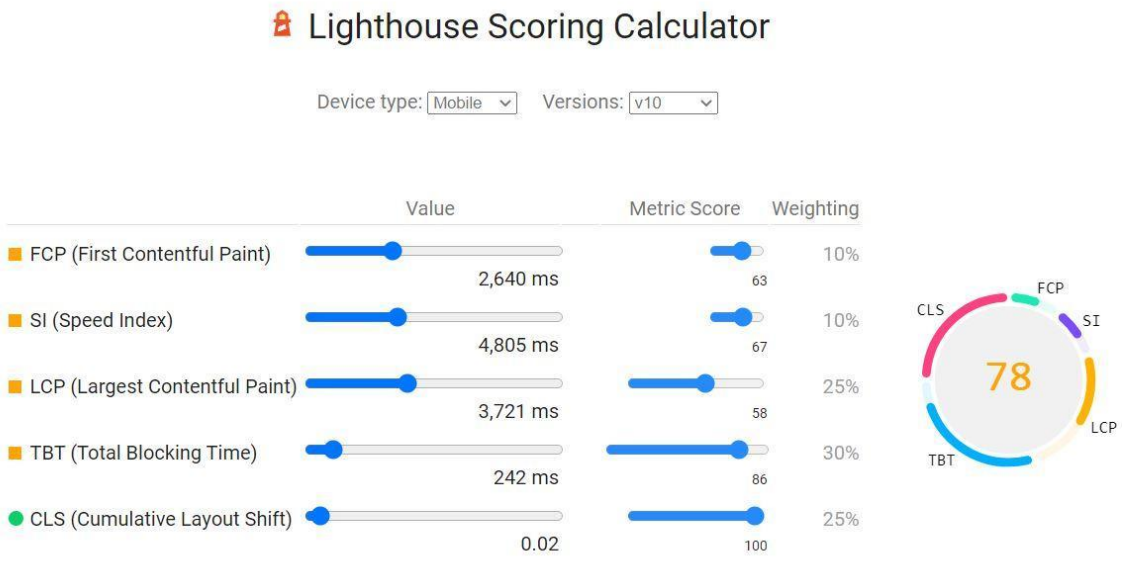


Figure 44: Performance Calculation of Student page in Mobile Version

Appendix Section C - Evaluation

Appendix Section C.1 - Questionnaire feedback response from the end users

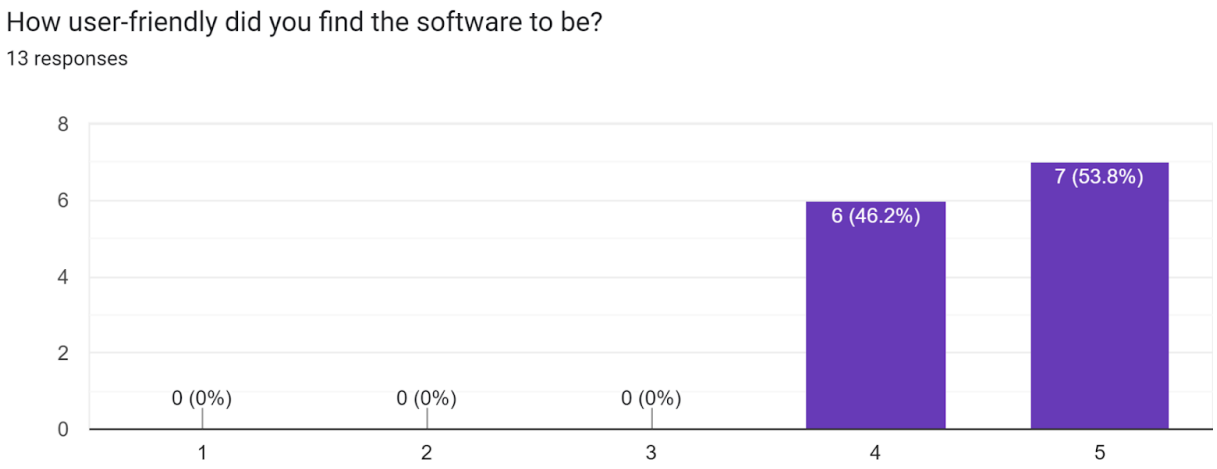


Figure 45: Feedback questionnaire Q1

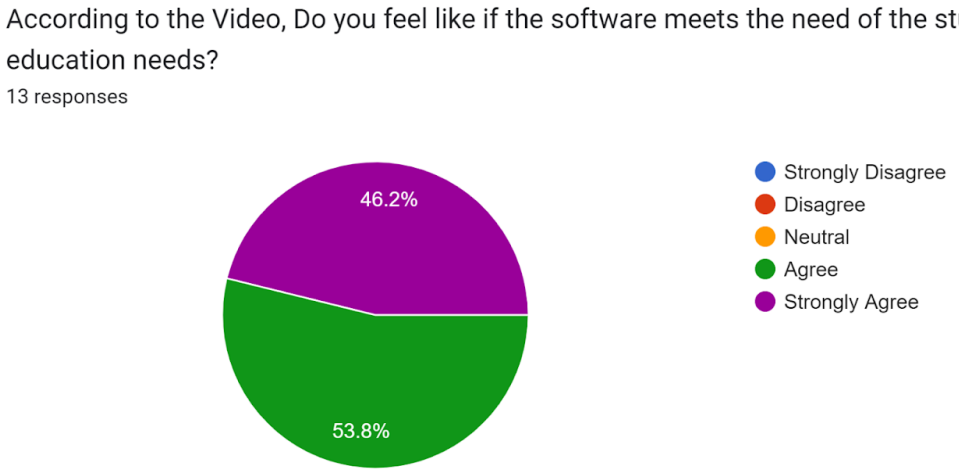


Figure 46: Feedback questionnaire Q2

Were the features of the software helpful in meeting the individual needs of each student?

13 responses

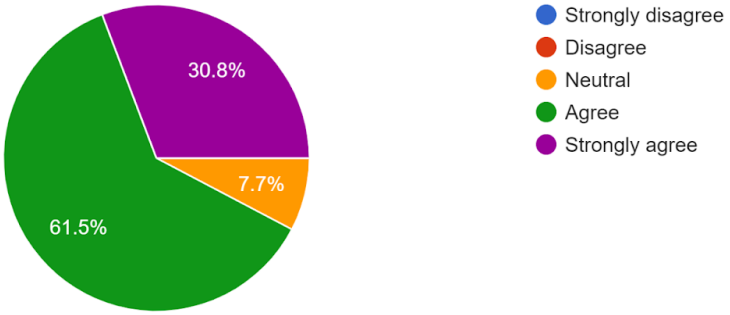
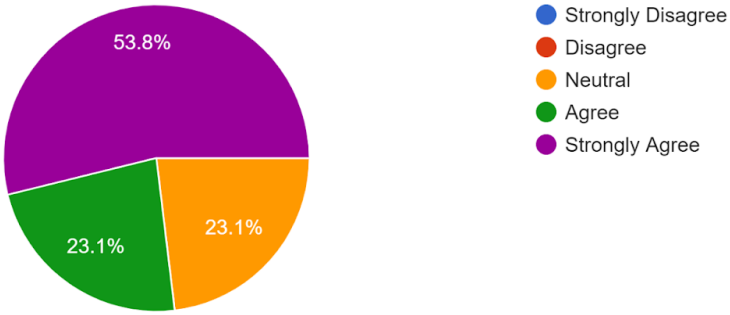


Figure 47: Feedback questionnaire Q3

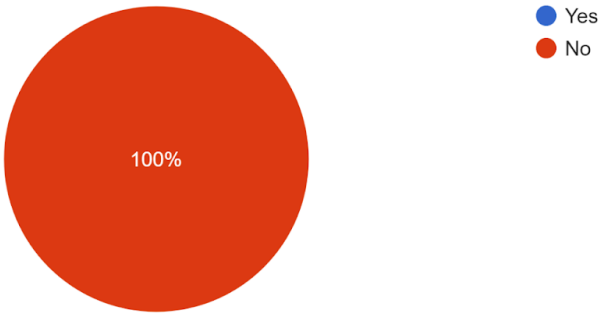
According to the Video, Do you feel like if the software provides accurate and useful data to track progress and identify areas for improvement?

13 responses



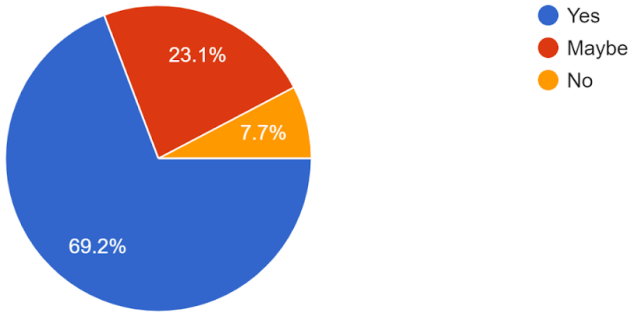
Were there any visible technical issues or glitches with the software?

13 responses



Would you recommend this software to other educators or schools with students who have special education needs?

13 responses



In a scale of 1-5 how likely are you to recommend this to an educator in the Special Education Needs field?

13 responses