A Project for Software Engineering Lab

Code Execution Engine

A project submitted to the Department of Computer Science and Engineering in partial fulfillment of CSEL-3206 Software Engineering Lab course for the Degree of B.Sc. in Computer Science and Engineering.

By

Md. Ashrafuzzaman

ID: B180305033

Naheed Rayhan

ID: B180305041

Supervised By

Dr. Md. Ashraf Uddin

Associate Professor

Dr. Md. Manowarul Islam

Associate Professor

Department of Computer Science and Engineering
Jagannath University

December,2022

Recommendation of the Board of Examiners

This project titled ***Code Execution Engine*** submitted by Md.Ashrafuzzaman (ID: B180305033) and Naheed Rayhan (ID: B180305041) has been found as satisfactory and accepted as partial fulfillment of CSEL-3206 Software Engineering Lab course for the Degree of B.Sc. in Computer Science and Engineering on December 2022.

Examiners

1. ..... Supervisor

2. ..... Examiner

3. ..... Examiner

4. ..... Chairman

Declaration of Authorship

This is to certify that the work presented in this project is carried out by the candidates - Md.Ashrafuzzaman (ID:B180305033) and Naheed Rayhan (ID:B180305041) under the supervision of Dr. Md. Ashraf Uddin in the Department of Computer Science and Engineering, Jagannath University, Dhaka - 1100, Bangladesh. It is also declared that neither of this project nor any part of this project has been submitted anywhere else for any degree of diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Countersigned	Signature	Signature
.....	.....	.....
(Dr. Md. Ashraf Uddin)	(Md. Ashrafuzzaman)	(Naheed Rayhan)
Supervisor	**Candidate**	**Candidate**

ACKNOWLEDGEMENT

First, we would like to thank the most influenced people our project time. They are our parents, then our sibling, for their continuous support and criticism.

We extend our sincere thanks to Jagannath University, CSE Department which provided us with the opportunity to fulfill our wish and achieve our goal.

We would like to express deep debt to honorable teachers **Dr. Md. Ashraf Uddin, Dr. Sajeeb Saha, Dr. Md. Manowarul Islam** and all faculty members for their vital suggestions, meticulous guidance and constant motivation which went a long way in the successful completion of this project.

We can't move on without thanking our beloved **Dr. Uzzal Kumar Acharjee**, Professor & Chairman of the Department for creating the required academic environment which made our task appreciable.

On a moral personal note, our deepest appreciation and gratitude to our beloved parents, who have been an inspiration and have provided with unrelenting encouragement and support.

Authors

Signature:	Signature: .....
Date:	Date: .....
Md. Ashrafuzzaman Id: B180305033 Batch: 10th Department of Computer Science & Engineering Jagannath University	**Naheed Rayhan** Id: B180305041 Batch: 10th Department of Computer Science & Engineering Jagannath University

Abstract

Cloud computing is the future of computing. What makes it stand apart is the fact that everything including data and applications are stored on the cloud itself and are accessible through an Internet connection and a web browser. Integrated Development Environment (IDE) which is used for coding and developing applications can be a software on the cloud.

This project ***Code Execution Engine*** provides us with a simple interface for compiling codes online with secure API, get results and save code. It can be used by any person to practice coding in different language like c, cpp and many more. Any Organization can also use our API in their system like virtual judge. This is a web-based system and our API is our main hero that clients can use as per their given access permissions. The main aim of this project is to create an API and website that can compile different coding language efficiently and save the user information like their codes. The user can update their information, can save and update code, can compile those codes online and also use our API. From this system, a user can easily code from anywhere in the world online. Throughout the project, the focus has been on presenting information in an easy and intelligible manner. This software system has better features and comfortable user requirements which are essential for better performance of the system.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction

The latest trend is to take desktop applications online and to provide them as a service. There are already many online editors like Google Docs & Online Microsoft word. Just like these text editors, even Integrated Development Environments can be taken online and provided as a service.

Today, programmers are facing an increasing need to program from anywhere, anytime. Also, a need is felt to be able to program from devices other than just the desktops and laptops. It would also be better if there was no requirement of downloading and installing software for the purpose of coding and running of programs e.g. the JDK / MinGw or a Desktop IDE.

All these requirements can be fulfilled with the help of an online IDE. An online IDE which can also be called a browser based IDE is an online coding environment that allows users from across the globe to access and use this software as a service. This software can be accessed from devices like smart phones, desktops, laptops, etc. that have the ability to access the Internet and have a web browser.

To run this online IDE our API comes in to play which is the brain of online IDE and takes all the load and execute codes online.

So in this project ***Code Execution Engine***, we present the idea of browser based IDE and our own API to execute the code and give us the results. This API can also be used in others web based IDE.

1.2 Works of Programmer

Programmers are the wizards of this era. They are the people who are contributing in every step of our life. Everything is run by code nowadays. Programmers write, modify, and test code and scripts that allow computer software and applications to function properly.

Programmers also solve real world problem with programming. They are known as Problem Solver.

Normally, a programmer's daily work will differ depending on the app he creates or problem that they tries to solve. Since he creates different software for different companies, his daily duties will vary as well. However, most programmers spend most of their time coding- either on their computers or with computers. This can be done at home or at work since most programmers work from home part-time so they can focus on their job without interruption.

Sometimes they need to test their code on the go that's where they need online code editor or online IDE. So that they can simply run & test a code in the web from their mobile also.

1.3 Motivations

Any work has a purpose. That's exactly what needs a motivation to do that job. There is also motivation behind creating this project. The world is moving towards the cloud computing. Our motivation is also trying to implement an IDE online to execute code.

The nominal purpose of the project is to develop a website and API to compile and run the code in the sandbox environment and save the user's code in the database with their information. The compiler server will compile and run the program. So that a user can easily execute the code from anywhere in the world. We are also making this API technology accessible to the other users so they can also implement that in their online IDE or judge.

1.4 Objective

The traditional realm of development tools is firmly entrenched in the realm of standalone applications. Because many of them rely heavily on the ability to compile, interpret, and analyze the code presented. Add to this the feature-rich IDE already available in the standalone domain, and it's no surprise that there is a marked lack of activity to develop a web-based equivalent. However, the current state of the Internet and its technology makes it easy to migrate applications (or at least some of their functionality) to online domains. IDE applications are no exception. It's easy to imagine a situation where access to a particular project's code and compiler would be made available through a common interface to developers regardless of (at least to some extent) the capabilities of their actual machines.

So our main objective to create an online IDE with simple interface that will give user the opportunity to execute the code from anywhere without thinking about the dependency and system capabilities.

Chapter 2

Background Study

2.1 Web Development

The face of the web has changed a great deal since the early days, with an ever-growing audience of Internet users pushing technology forward at an unprecedented pace and in unexpected ways, making the web experience much more interactive than was perhaps anticipated. With these advances has, quite naturally, come a number of changes in how web pages are developed, with a large amount of programming languages and tools being designed specifically for web development use. This section describes some of the aspects of modern web development, with focus on the technologies that are relevant in this days. We have used Java Script, JQuery, EJS, Node JS, HTML, CSS and Bootstrap for our project with relevant technologies.

2.1.1 Client vs. Server-side Programming

One of the most fundamental aspects of web development is that of the separation of application logic and display. In order for web applications to be as lightweight and responsive as possible while at the same time providing extensive capabilities in a state-less environment (HTTP, the Hypertext Transfer Protocol, does not allow websites to keep state — instead, updates are achieved by passing POST or GET messages from the client to the server), the functionality is split into two sections, a client and a server side. The client side is what is visible to the user and generally contains, in addition to the markup (see 2.1.1.1), the functions that interact with the user and alter the appearance of the website, e.g. the DOM manipulation functions, as well as utility functions, e.g. for sanity-checking user input. The main language for client-side scripting is JavaScript, an interpreted language with facilities influenced by both functional and imperative programming paradigms. Data processing and more advanced functionality, e.g. file I/O or database operations, are performed on the server side of the application, and are therefore independent of the capabilities of the users. This in turn means that it is up to the developer to decide what language to use, and that a vast number of programming languages can be used.

2.1.2 Database Programming

Many web applications require data to be stored on the server in order to provide the desired services, e.g. user-data for log-in functionality to work. The standard way to provide this information is through the use of a server-side database which stores the data in some data structure and with which the application can interact by means of data queries written in a query language, for example the Structured Query Language (SQL). Databases are usually provided by a database management system (DBMS) which specifies the organization of the data and what data structure and query language to use, as well as some transaction mechanism to ensure data integrity. Due to the nature of the data storage, the latter is one of the most important selling points of DBMSs. From a web development point of view, DBMSs are useful as they provide a straight-forward way of accessing particular pieces of information from a (potentially very) large data set. Furthermore,

depending on the DBMS used, they give the application a host of extra functionality, e.g. added security mechanisms and support for rollbacks, as well as more advanced ways of manipulating and separating data, e.g. through the use of triggers (event handlers), stored procedures and views.

2.1.3 API

API stands for application programming interface, which is a set of definitions and protocols for building and integrating application software.

APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols.

We have used Python in our API. Our API connects the website IDE with the Server so that the code can be execute.

Our API can handle C, C++, JAVA, Dart, JS and Python Codes. A programmer can compile this languages in our online Code Execution Engine.

2.2 Web Development Tools

We have used this tools and languages

2.2.1 Hypertext Markup

In order to display more than just bare bone text in a document, a method of providing more information about the components is needed. One way of attaching this information to document entities is by using a markup language, which allows pieces of text or other elements to be tagged with information guiding the rendering of the document. For the web, the main markup languages used are Hypertext Markup Language (HTML) and its successor extensible HTML (XHTML), both of which are used to allow non-textual items such as images, tables and lists to appear in webpages. (X)HTML and related extensible markup languages (XML) are organized in a tree-like fashion, making it easy to navigate the tree programmatically using the nodes' parent-child relationships. The tree-like structure is also ideal for DOM scripting, i.e. manipulating the various nodes of the tree in order to change their ordering or particular attributes. DOM manipulation is commonly used to allow a webpage to react to user actions, for example by displaying a menu when a user clicks a particular node in the tree.

2.2.2 Cascading Style Sheets

Web markup languages are useful for describing the structure and contents of a page, but are somewhat limited in their ability to accurately describe its layout. To counter this, extra styling information is usually attached to a markup document, giving more fine-grained control over the look-and-feel of both nodes in the page and the page as a whole. The information is described using Cascading Style Sheets, or CSS, which allows styles to be assigned in a hierarchical manner (meaning that the styles of a parent cascade to its children) to individual nodes, groups of nodes (known as classes), and nodes of a specific type. Styles can be used to alter almost all aspects of a node, including its position, color, background, mouse cursor on hover, and font-styles.

2.2.3 JavaScript

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

2.2.4 Express JS

Express is a node js web application framework that provides broad features for building web and mobile applications. It is used to build a single page, multipage, and hybrid web application. It's a layer built on the top of the Node js that helps manage servers and routes.

Express was created to make APIs and web applications with ease. It saves a lot of coding time almost by half and still makes web and mobile applications are efficient. Another reason for using express is that it is written in JavaScript as JavaScript is an easy language even if you don't have a previous knowledge of any language. Express lets so many new developers enter the field of web development.

2.2.5 JQuery

JQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, JQuery has changed the way that millions of people write JavaScript.

2.2.6 Bootstrap

Bootstrap is the most popular CSS Framework for developing responsive and mobile-first websites. Bootstrap 5 is the newest version of Bootstrap.

2.2.7 Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured, object-oriented and functional programming.

We have used this in our API.

2.2.8 MySQL

MySQL is an open-source relational database management system. Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the abbreviation for Structured Query Language.

2.2.9 Docker

Docker is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime.

2.2.10 Redis

Redis is an in-memory data structure store, used as a distributed, in-memory key–value database, cache and message broker, with optional durability. Redis supports different kinds of abstract data structures, such as strings, lists, maps, sets, sorted sets, HyperLogs, bitmaps, streams, and spatial indices.

2.2.11 Rabbitmq (Task Queue)

RabbitMQ is an open-source message-broker software that originally implemented the Advanced Message Queuing Protocol and has since been extended with a plug-in architecture to support Streaming Text Oriented Messaging Protocol, MQ Telemetry Transport, and other protocols.

2.2.12 Node JS

Node.js is an open source server environment. It is used for server-side programming, and primarily deployed for non-blocking, event-driven servers, such as traditional web sites and back-end API services, but was originally designed with real-time, push-based architectures in mind. Node is useful for developing applications that require a persistent connection from the browser to the server and is often used for real-time applications such as chat, news feeds and web push notifications.

2.3 Features

This system has the following features:

- User can open account
- User can use the online IDE to execute the code
- User can Save code in the database
- User will get the error massage, exception and time outs.
- User can also request for the API access

- User can use this API in their online IDE
- User can save their information.
- Run C, C++, JAVA, Dart, JS and Python Codes

2.4 Involved Activities and Procedures

This system has various tasks User. User need to open an account first with Signup and then user can use the online IDE to execute code.

User can also save their code in the database and also update them letter.

User can also use our API. To use our API user have send request with mail. Then the admin will provide user with an API link that they can Integrate with their website like online code editor, IDE and online judge.

User can also modify their information.

2.5 Compare System Reviews

There are different types of Online IDE, Online code editor and many API provider. From Sphere-engine, we have taken inspiration from. It also provide online ide and the API service. But our API is faster than the Sphere engine in simple coding task. Our Online Code Execution (Titan Engine) Engine is faster than many online editors. The comparison is given below in a graph

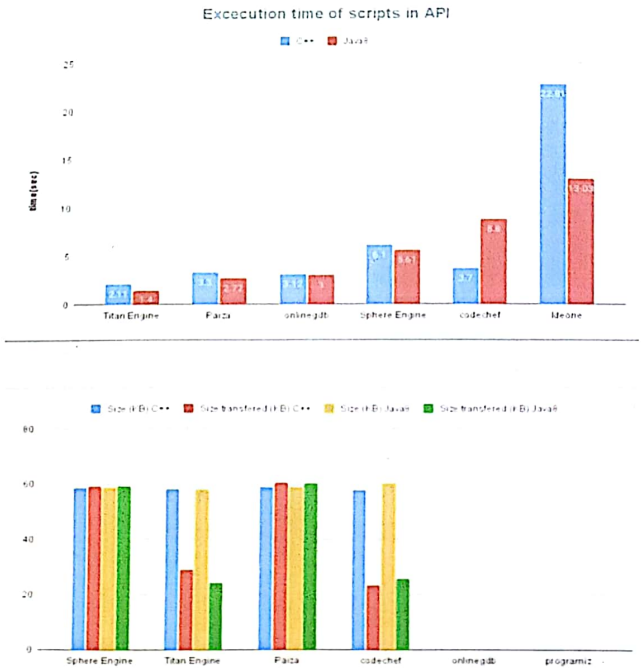

Figure 1: Comparing Systems with other web IDE

2.6 System Requirement Specification

2.6.1 Scope

This system is a reliable system to execute code. Through this system, user can run text code anywhere from the world without installing any additional requirements. Any device connected with internet witch can view website can run this system. So, a user can run code, test it, modify it and also can save it in the database so that he can modify it letter.

2.6.2 Goal

The main goal of implementing this project is to get hands-on experience working on the technologies used in this project which are JavaScript, EJS, Node JS, Python, Rabbitmq, Docker, redis server, MySQL etc. By obtaining knowledge of coding in JS, EJS, Node JS and designing efficient API servers will help me a lot in future projects involving these technologies.

2.6.3 Assumptions

The following assumptions are made when the users want to use this application-

- Use this system via desktop, laptop computer, or any Mobile device connected to the internet.
- The user will enter a valid user name and password.
- The API will handle the compiling load succesfully.
- Traffic will be manageable for the servers.

2.6.4 Hardware Requirements

- Processor : Pentium 2.66 GHz.
- Hard Disk : 1 TB.
- Ram : 4 GB.
- Mobile Devices Ram : 2 GB.

(Minimum requirements)

2.6.5 Software Specification

- Operating System : Windows 7,8,10,11, Linux
- Coding Language : EJS, NodeJS, JavaScript, Python, jQuery, Bootstrap
- Backend : MySQL
- Server : Xampp, redis, RebbitMQ, Docker
- Browsers : Any.

2.6.6 System Requirements Analysis

The most important goal of the application as mentioned above is to compile code from anywhere in the online with just a browser connected internet. We have carefully worked on understanding How API works and how we can compile codes in the server.

This system have users who are programmers, problem solver and those who just want to simply run codes without any installation.

Chapter 3

System Design

3.1 Project Approach

The main purpose of this project is to make life easy for programmers and problem solver so that they can do code anywhere. This is built to reduce the hassle of installing so many software to run a program. This website and API is managed by us.

The project's outputs are absorbed inside the project's consortium, as a project, based on the first phase of the study, the foundation for determining the report for continuous development of specifications and compiler related materials.

3.2 Architectural Design

Software requirements should be translated into an architecture that describes the top-level structure of the software and identifies its components. This is done through an architectural design (a.k.a. system design) that acts as the first "blue-green" upon which software can be built. This framework was developed to study software requirements documents and design models to provide implementation reports. These statements are used to define system components including inputs, outputs, functions, and interactions between them. The IEEE defines architectural design as "the process of defining a collection of hardware and software components and their interfaces to establish a framework for the development of computer systems."

Designing the architecture is a critical software engineering requirement related to time, reliability, cost, and performance. This task becomes difficult as the software engineering paradigm shifts from single-tier, monolithic "build from scratch" systems to compositional, imperative, standards-based, and product line-based systems. The challenge for designers is also to understand exactly how to get from architectural design to requirements. To avoid these problems, designers employ reuse, composition, platform-based, standards-based, and other techniques.

In our website we have three part. They are website, API and database.

However, architectural design involves the responsibilities of others such as developers, user representatives, systems engineers, hardware engineers, and operations staff. When reviewing the architectural design to minimize risk and error, these stakeholders must be consulted.

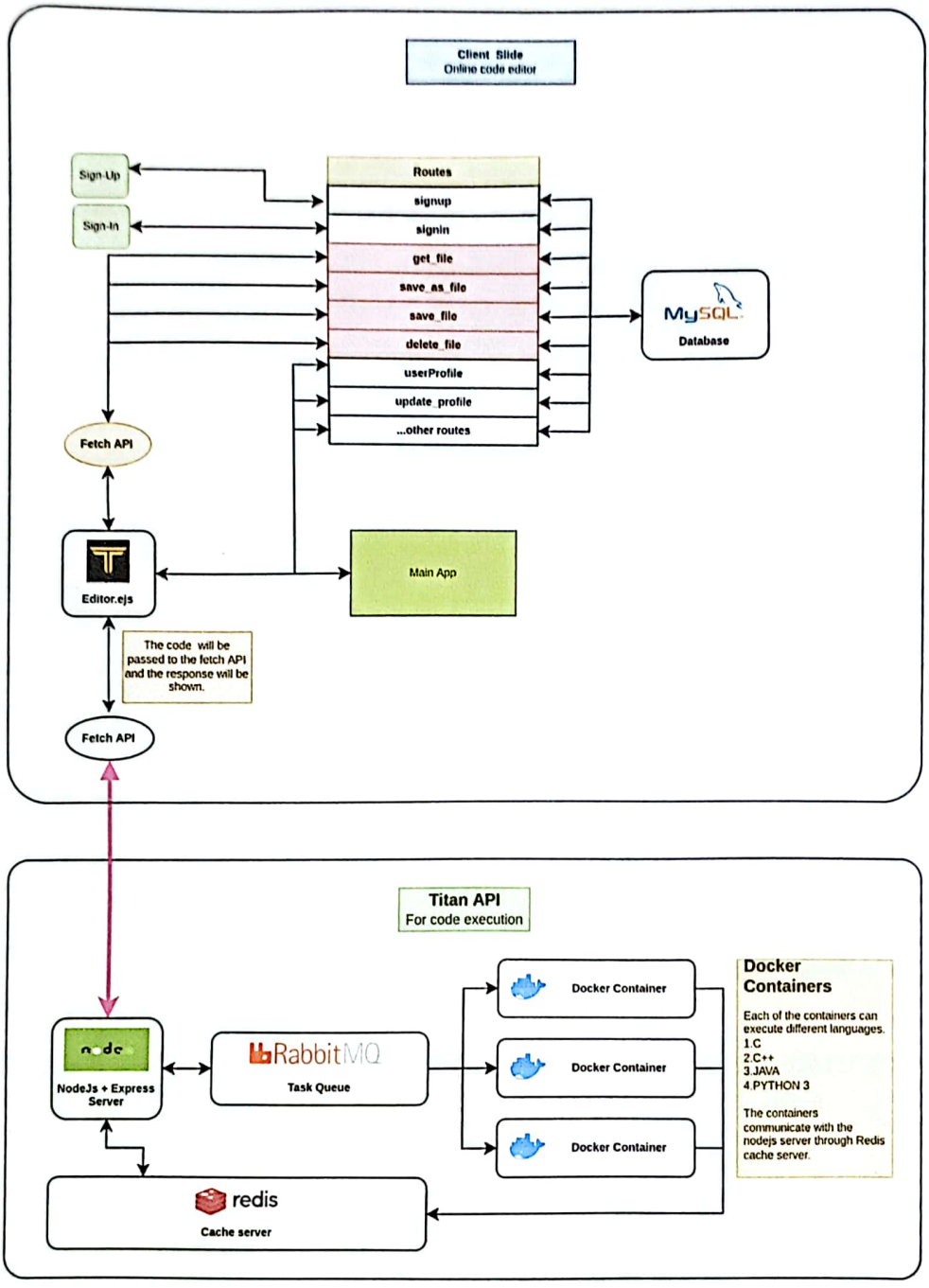

Figure 2: Architecture Design of Code Execution Engine

3.3 UML Diagram

UML stands for Unified Modeling Language. One way to visualize a software program is by using a collection of UML diagrams. Today, UML is recognized as the standard of modeling software development by Object Management Group (OMG). This idea has been derived from the use of Gradia Bouch, James Rumba, Evar Jacobson, and the Rational Software Corporation for Object-based Design, but it is widely involved in a variety of software engineering projects.

Behavior diagrams represent functionality of software system and emphasize on what must happen in the system being modeled. The different behavior diagrams are:

- **Activity Diagram:** Represents step by step workflow of business and operational components.
- **Use Case Diagram:** Describes functionality of a system in terms of actors, goals as use cases and dependencies among the use cases.
- **State Chart Diagram:** Represents states and state transition.
- **Class Diagram:** Class diagrams are the blueprints of your system or subsystem.
- **Deployment Diagrams:** In UML, deployment diagrams model the physical architecture of a system. Deployment diagrams show the relationships between the software and hardware components in the system and the physical distribution of the processing.
- **Interaction Overview Diagram:** Provides an overview and nodes representing communication diagrams.
- **Data Flow Diagram:** A data flow diagram (DFD) maps out the flow of information for any process or system.
- **Communication Diagram:** Represents interaction between objects in terms of sequenced messages.
- **Sequence Diagram:** A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction.

3.4 Use Case Diagram

A use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language as an actor) and a system to achieve a goal. The actor can be a human or other external system. In systems, engineering use cases are used at a higher level than within software engineering often representing missions or stakeholder goals.

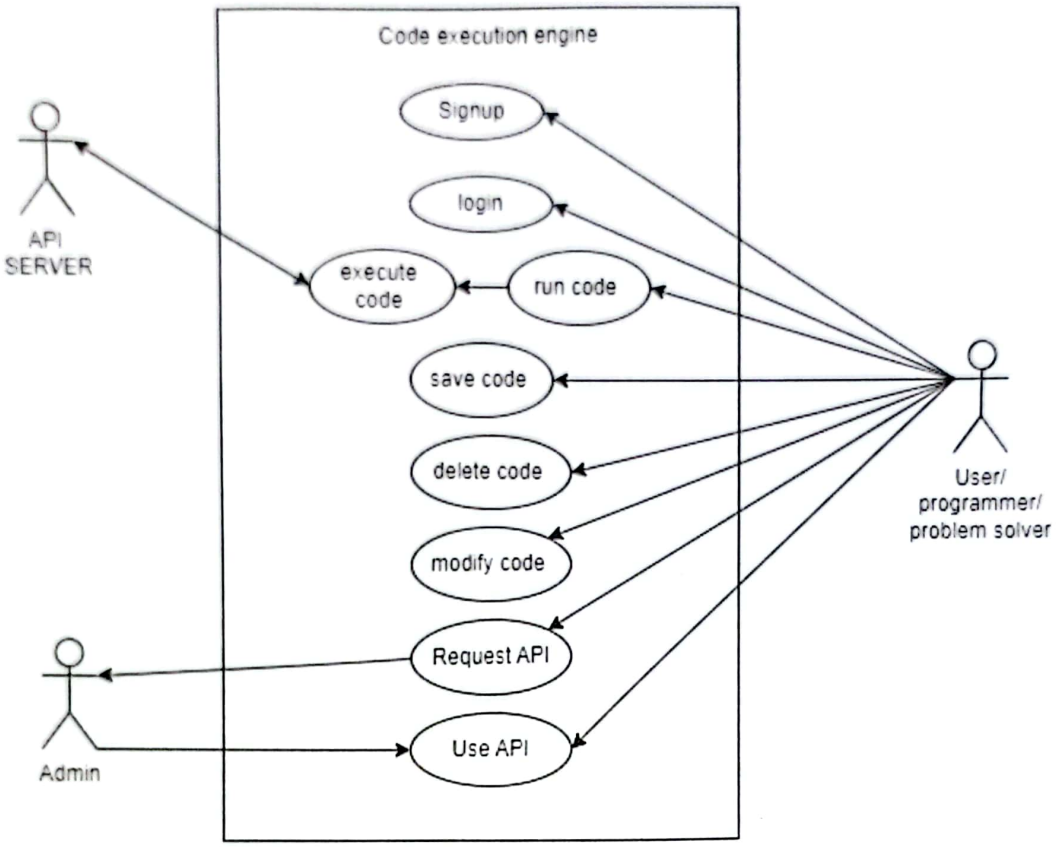

Figure 3 : Use Case Diagram of Code Execution Engine

The figure above shows the 'Use Case Diagram' (Figure 3) for *Code Execution Engine*. This system can be for two types of users, which are admin, user who can be programmer or problem solver or any day to day coder, and a system API server. User can sign up and login in this website. They can execute code there in the ide. User can also save their code and modify them any time. And in the website they can send request to the admin through the email for the API access. Then the admin can give access to the user after a payment plan or modify the API as the requirements of a user. The API run the code and send result to the IDE.

3.5 Data Flow Diagram (DFD)

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles, and arrows, plus short text labels, to show data inputs, outputs, storage points, and the routes between each destination.

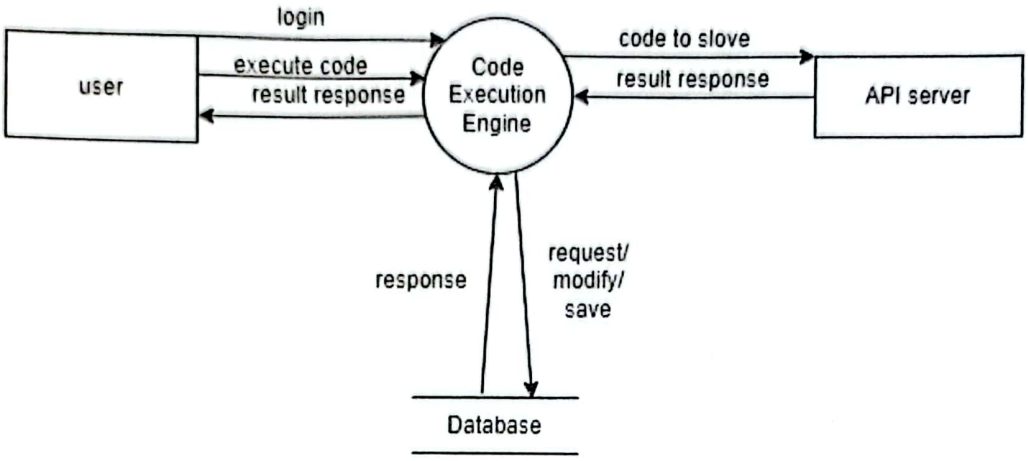

Figure 4 : DFD 0 Diagram for Code Execution Engine

The figure above shows the 'Data Flow Diagram (Level-0/Context-Level)' (Figure 4). This system can have user who can login send request for execution of code and modification of data. API server analyze those code compile them and send result to the user back. Database keep all the data stored in this system.

3.6 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. An activity is shown as a rounded box containing the name of the operation. This activity diagram describes the behavior of the system.

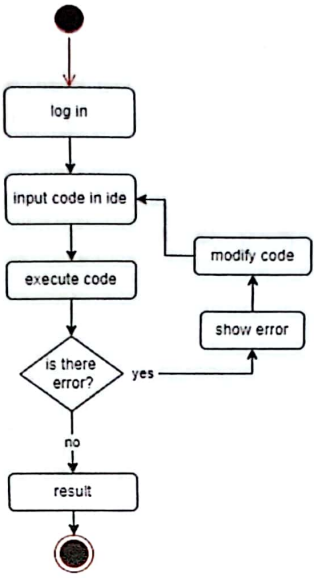

Figure 5 : Activity Diagram for execute code by user

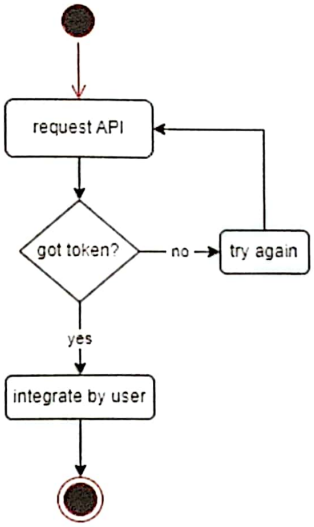

Figure 6 : Activity Diagram for using API by User

In Figure 5, User have to do all this things step by step to run code and get the result from API. Here user need to login, input code, then execute and if there is no error then get results.

In Figure 6, we have the activity diagram for taking API service. Requesting API and getting token for integrating this to the user website or any service.

3.7 State Chart Diagram

It consists of state, events and activities. State diagrams are a familiar technique to describe the behavior of a system. They describe all of the possible states that a particular object can get into and how the object's state changes as a result of events that reach the object.

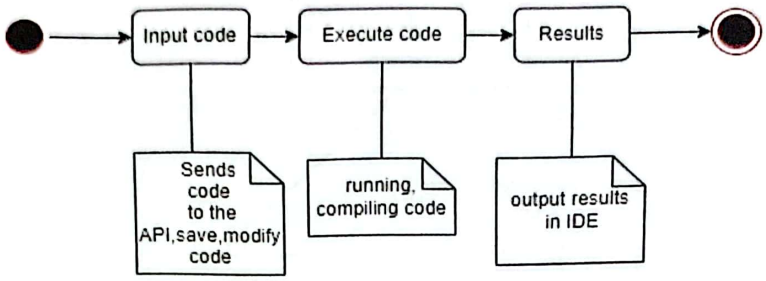

Figure 7 : State chart Diagram for executing code

In this diagram (Figure 7) it's a state diagram for execution of code for user.

3.8 Class Diagram

The class diagram, also referred to as object modeling is the main static analysis diagram. The main task of object modeling is to graphically show what each object will do in the problem domain. The problem domain describes the structure and the relationships among objects.

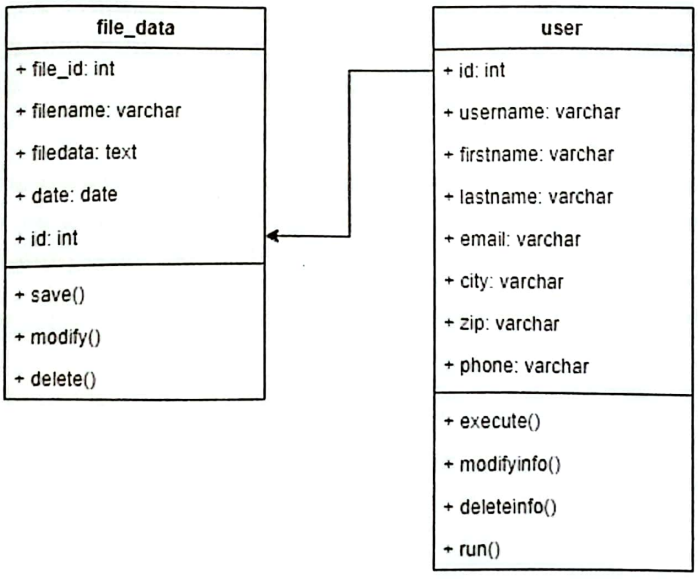

Figure 8: Class Diagram for User Database

3.9 Deployment Diagram

Deployment diagrams are used to visualize the topology of the physical components of a system where the software components are deployed.

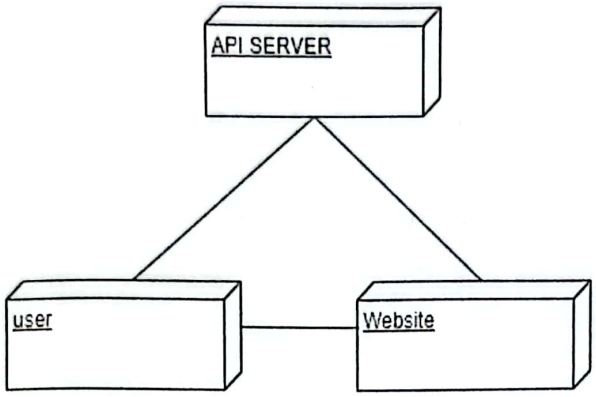

Figure 9 : Deployment Diagram for APi server

In this (figure 9) Deployment diagram we have the three basic components that are running.

3.10 Sequence Diagram

A sequence diagram is a Unified Modeling Language (UML) diagram that illustrates the sequence of messages between objects in an interaction. A sequence diagram consists of a group of objects that are represented by lifelines, and the messages that they exchange over time during the interaction.

In this Sequence Diagram (Figure 10), step to step guide to execute code and use the API. First user have to login to use the IDE and then user have to input the code then it will request API for the result. API will process the code and send response to the user.

User can save code to the database and also use token to use the API for website.

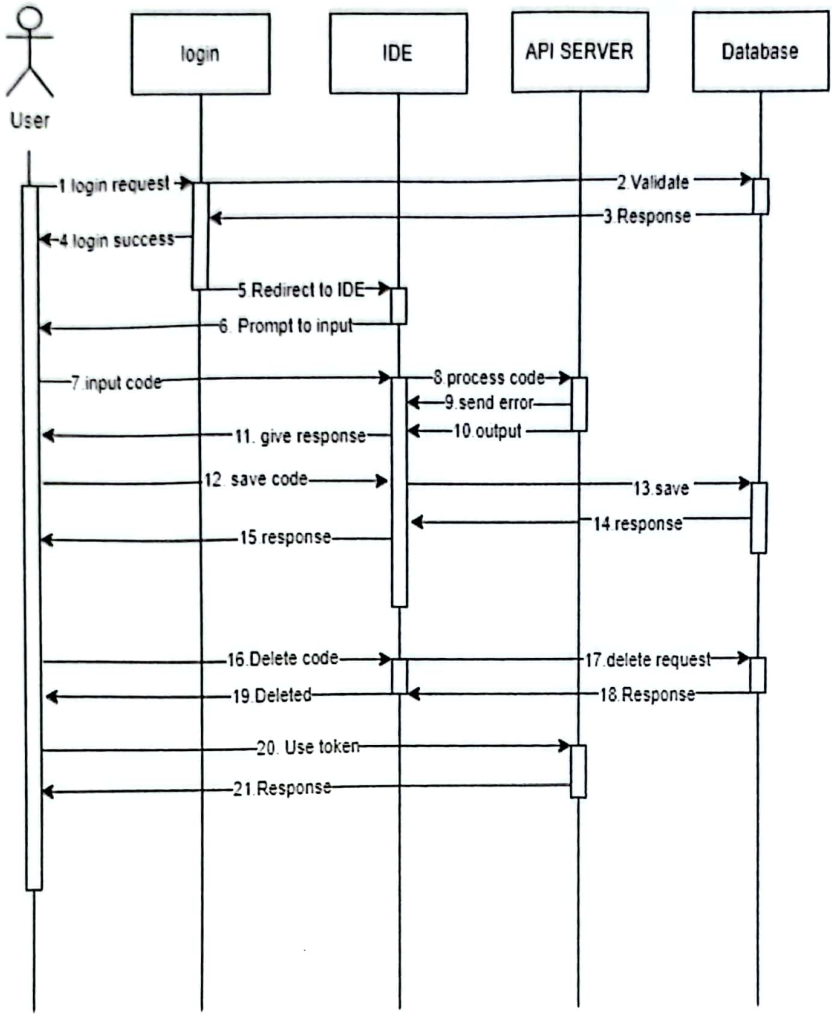

Figure 10 : Sequence Diagram for Code Execution Engine

3.11 Time Frame

3.11.1 Gantt Chart

A Gantt chart is a bar chart that provides a visual view of project tasks scheduled over time. A Gantt chart is used for project planning: it's a useful way of showing what work is scheduled to be done on specific days. It helps project managers and team members view the start dates, end dates and milestones of a project schedule in one simple stacked bar chart.

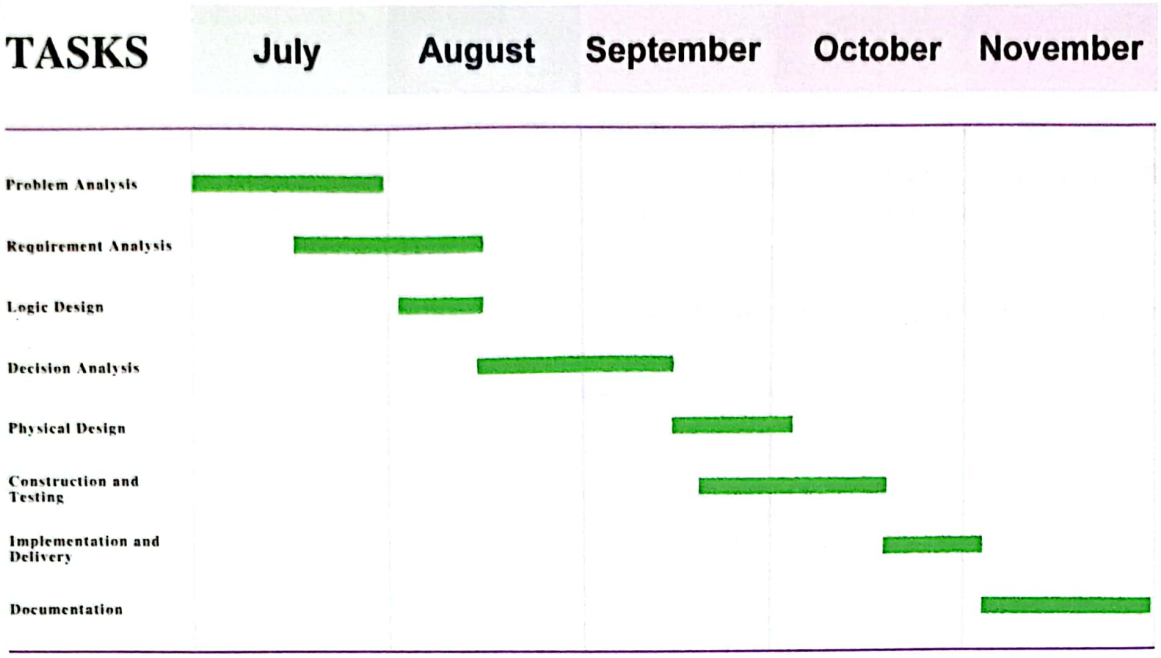

Figure 11 : Gantt chart for the Time Period of Project

This is the Gantt chart of this project figure (Figure 11). The Gantt chart show that how many times are needed to finish this Project. The diagram above shows the start and end of the project. The project work started in July 2022. Most designs are configured in August. And it is completed in October. In the chart left is a list of activities and a suitable time scale along the top. The chart identifies tasks that cannot begin until another chart is completed.

3.11.2 Evolution Timeline

Project evolution timeline, here we provide the iteration info of our Project.

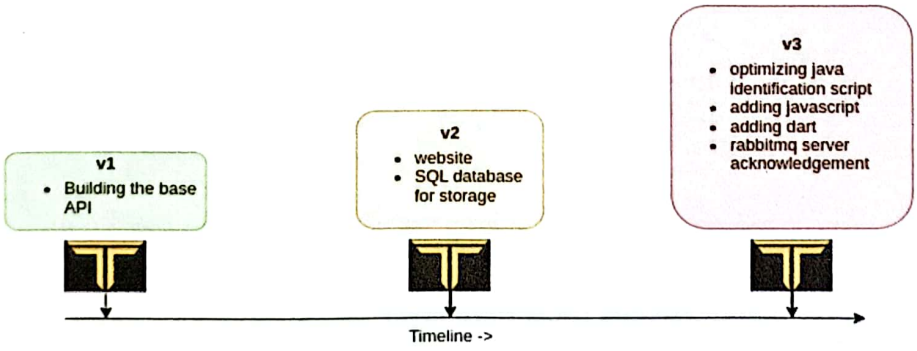

Figure 12 : Project Evolution Timeline

3.12 Entity Relationship Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships. An entity relationship diagram describes inter-related things of interest in a specific domain of knowledge.

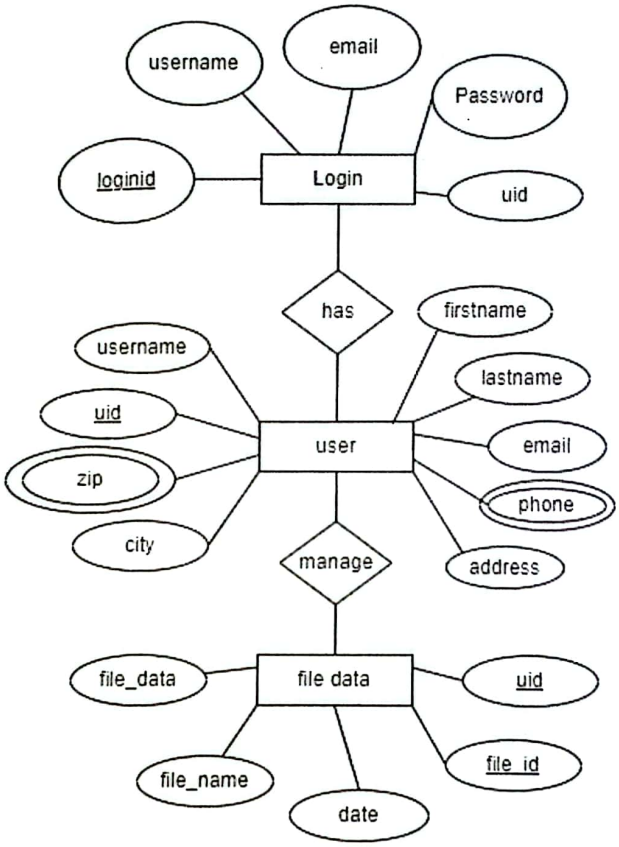

Figure 13 : ER Diagram for user database in Code Execution Engine

In this ER Diagram we have simply shown how user data is handled in the database. We basically have one type of user as those who wants to code on the online at any device without any installation hustle. User login info is handled first then the user info is saved. In the last we save the file like codes. So they can run it in future or modify it from anywhere.

Chapter 4

System Implementation

4.1 Sign Up

From here, a user can sign up into the system by using their information. Then they can use the website.

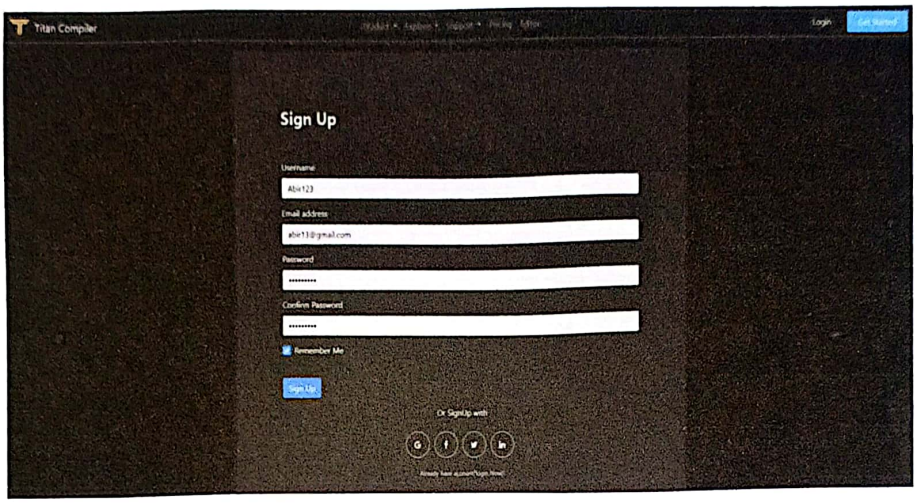

Figure 14 : Sign up page for website

4.2 Login

From here, a user can login into the system by using their user credentials.

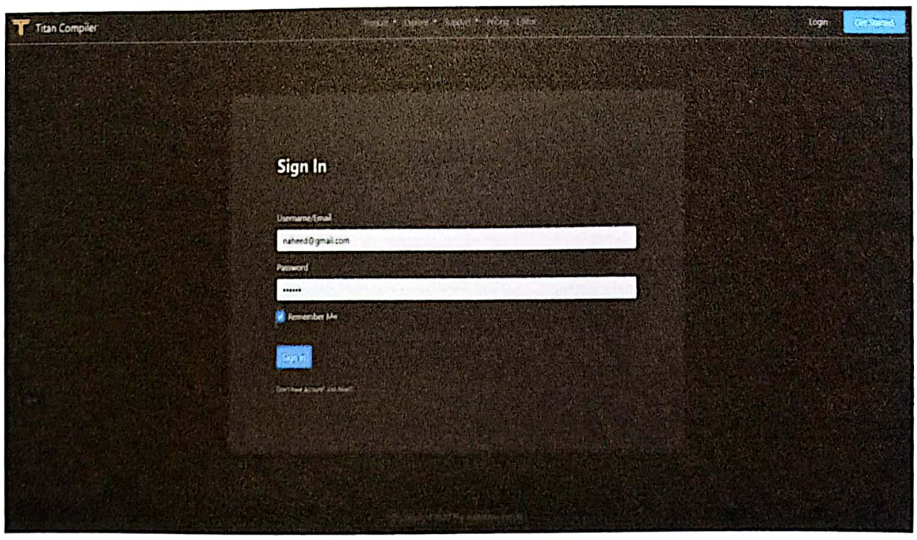

Figure 15 : Login or Sign In page

4.3 Validate

After successful login user will be redirected to the IDE. The main part of this system is IDE. So to login a legit user we need to validate the user that he exist in database or not.

4.4 Home Page

This the first page when a user will visit our website. In the top there is a navigation bar to navigate to different pages.

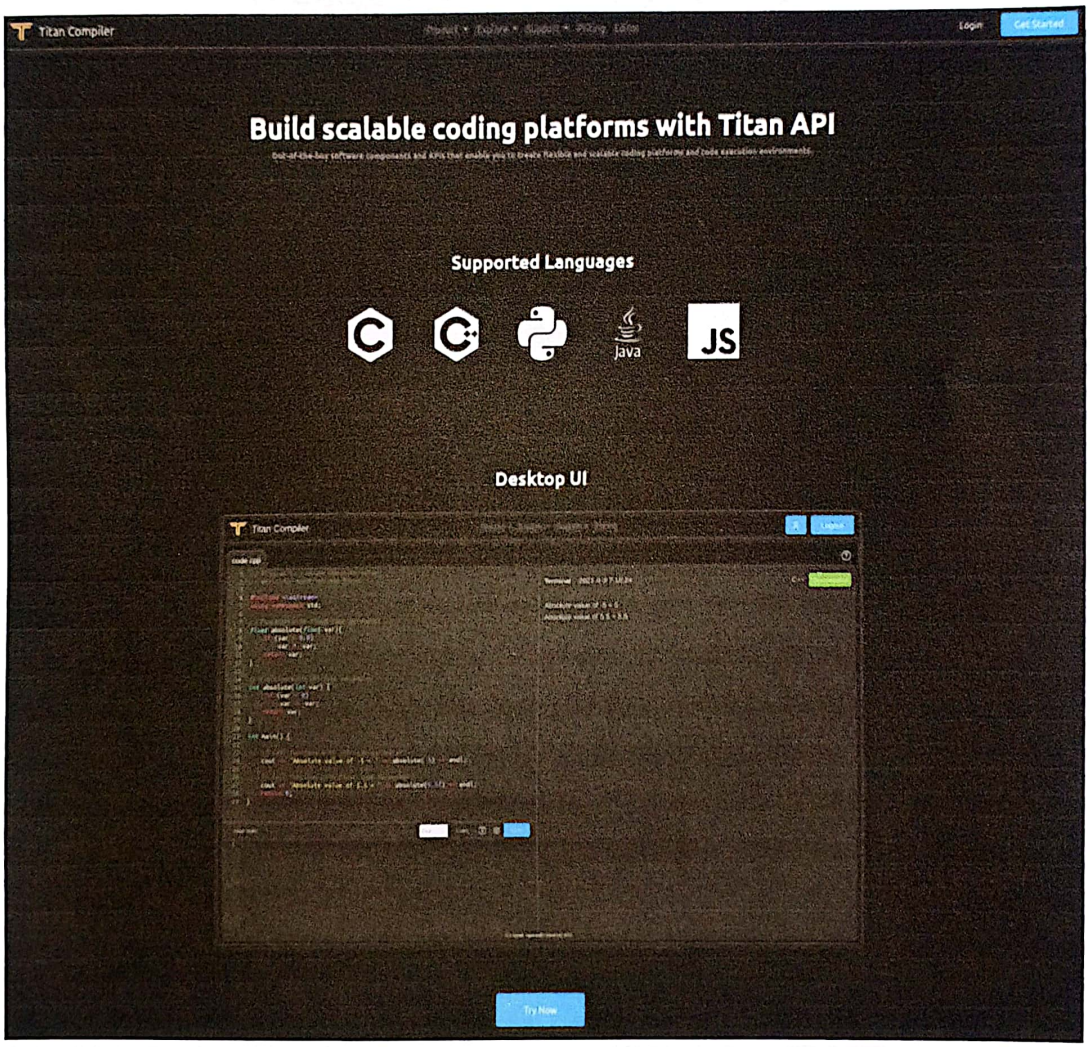

Figure 16 : Homepage

Figure 17 : Homepage (2)

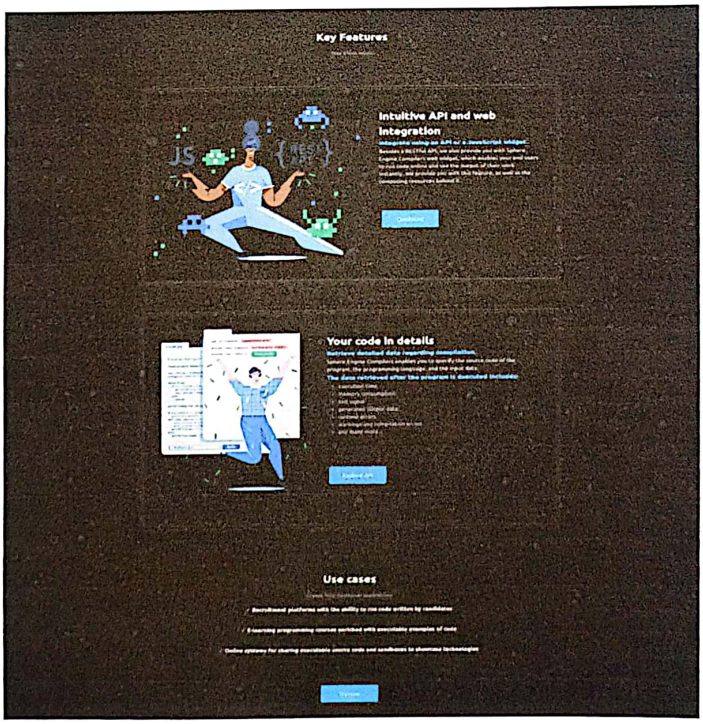

Figure 18 : Homepage (3)

4.5 IDE

This the page where a user will input his code and here all the code modification take place.

Figure 19 : IDE

4.5.1 IDE output

In the IDE, the output will be seen in the left side.

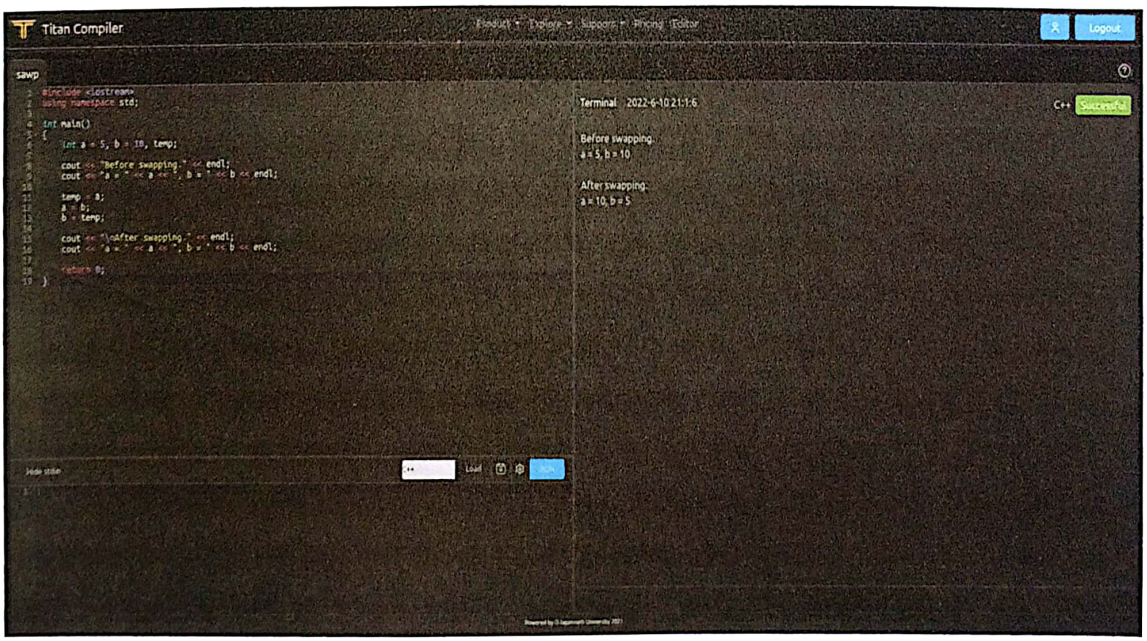

Figure 20 : IDE output section

4.5.2 Saving code

In this IDE model we can save the code. We can also rename the code also using save as button.

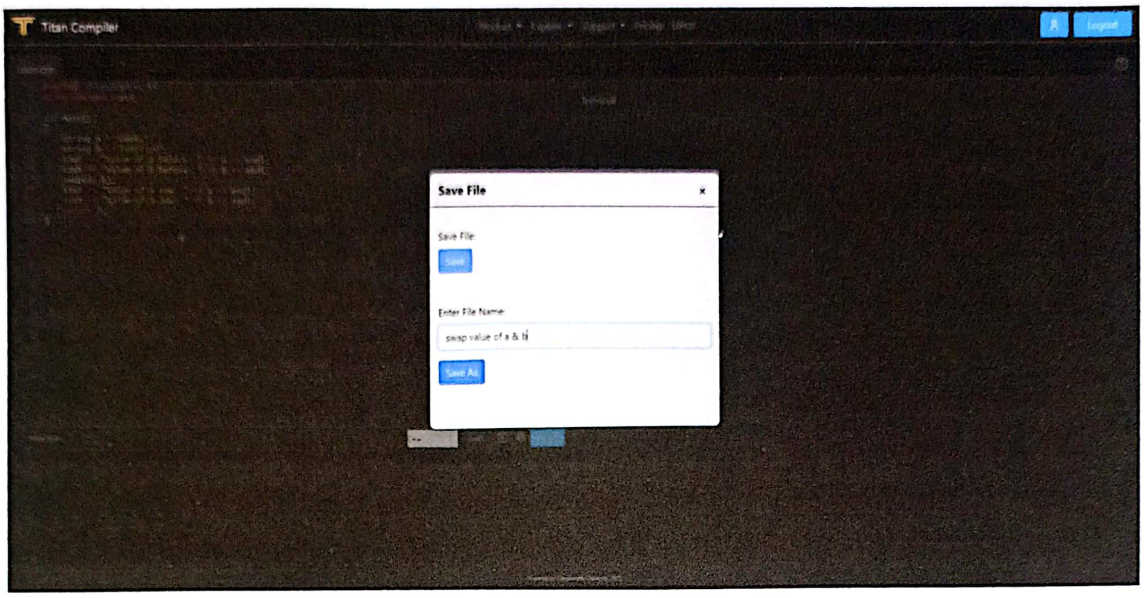

Figure 21 : Saving File

4.5.3 IDE Loading File

We can also load the code from the database, modify the code and delete the code.

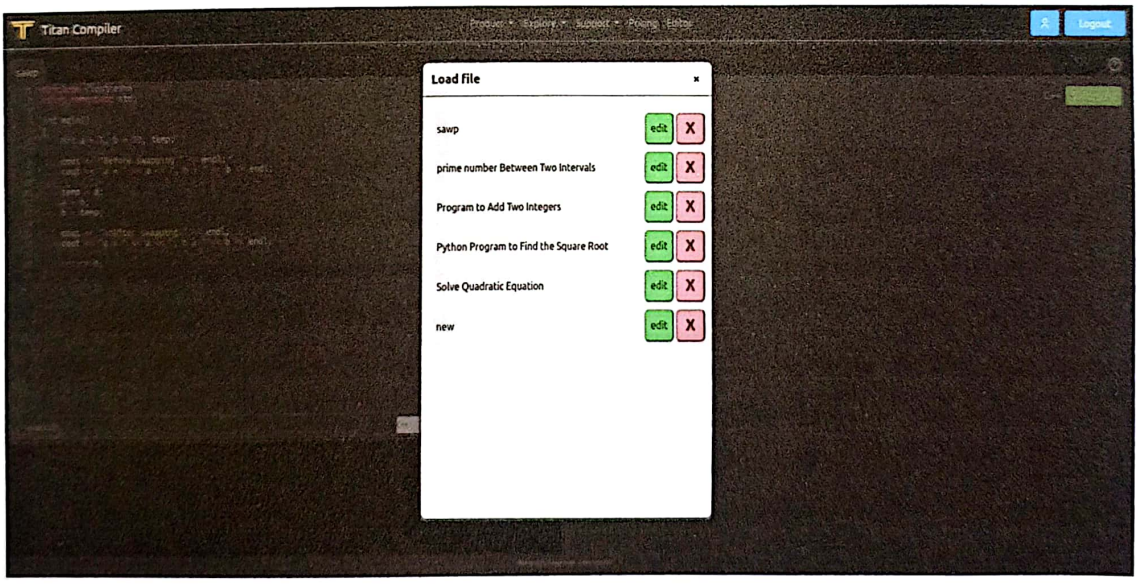

Figure 22 : Loading saved code

4.6 Service Request

There is a page for service request of API. User will send email for the API service. Admin will then set up a payment process and customize the API for the user then provide the token to the user so they can use the API in their website or system.

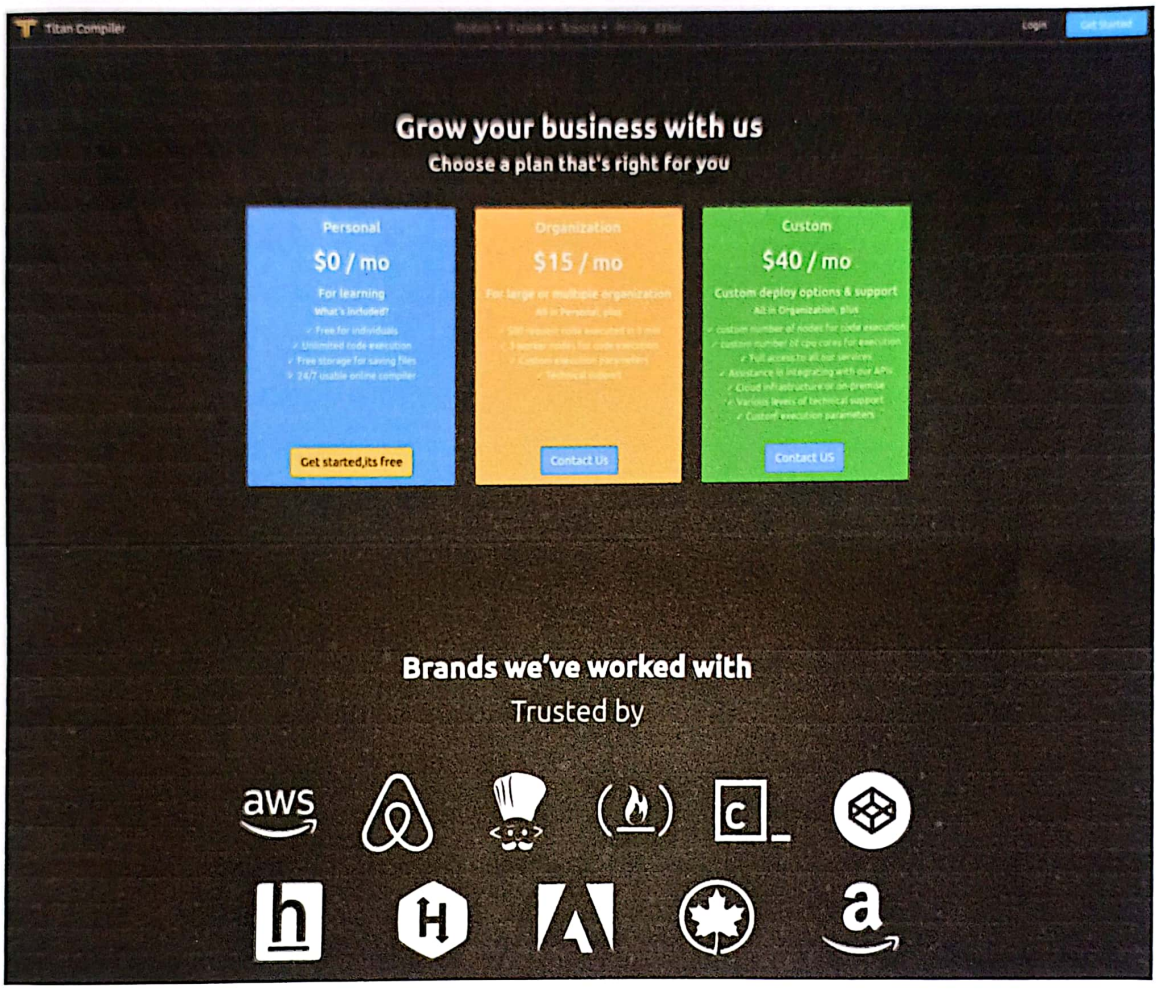

Figure 23 : Service Request for API

4.7 Mobile view of IDE

IDE have a great mobile view for the user so that the user can interact with the ide easily.

4.7.1 Input code (Mobile)

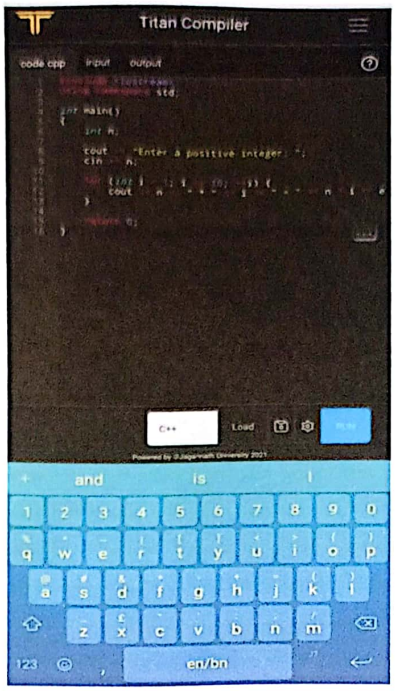

Figure 24 : Mobile view IDE

4.7.2 Output (Mobile view)

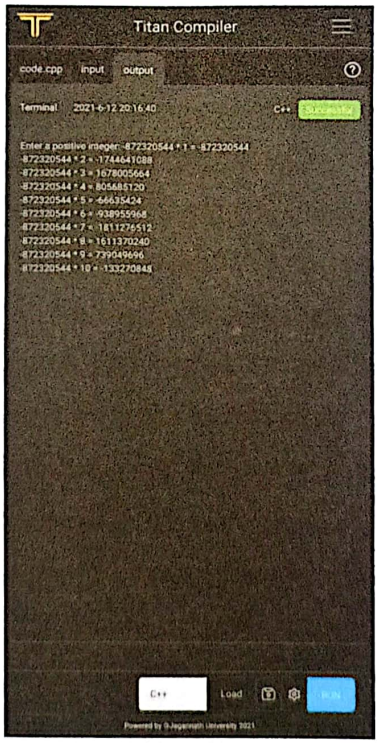

Figure 25 : Mobile view IDE output

4.8 User Profile

After signup user will have this profile page for their Information and modification.

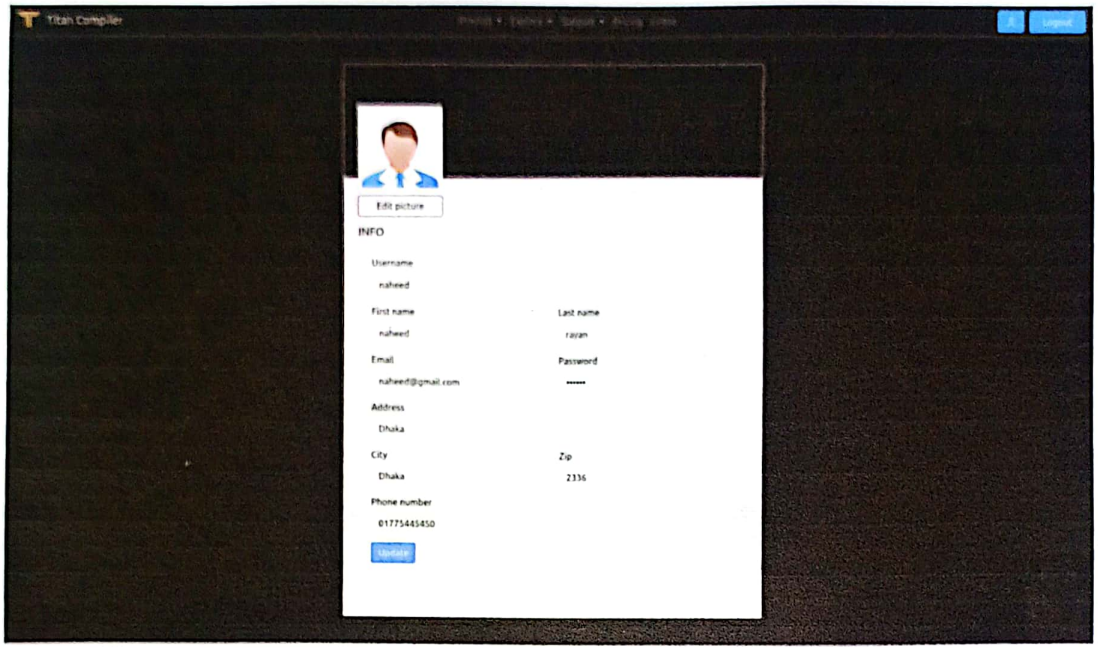

Figure 26 : User Profile

4.9 Database table

This is the database for saving user file and their information in the website.

Figure 27 : Database table from MySQL

4.10 API Server

The API is installed in Microsoft Azure server. The API can also run in a Desktop with docker.

4.10.1 Microsoft Azure Server

The API is set up in the Microsoft Azure Student account without credit card in virtual machine.

VM Specs:

- 1 core
- 1 GB RAM
- 32 GB Storage
- CPU: Intel Xeon E5-2673 v4 (1) @ 2.294GHz
- GPU: 00:08.0 Microsoft Corporation Hyper-V virtual VGA
- Server public ip : 52.172.231.206

we added 2GB swap memory for smooth operation.

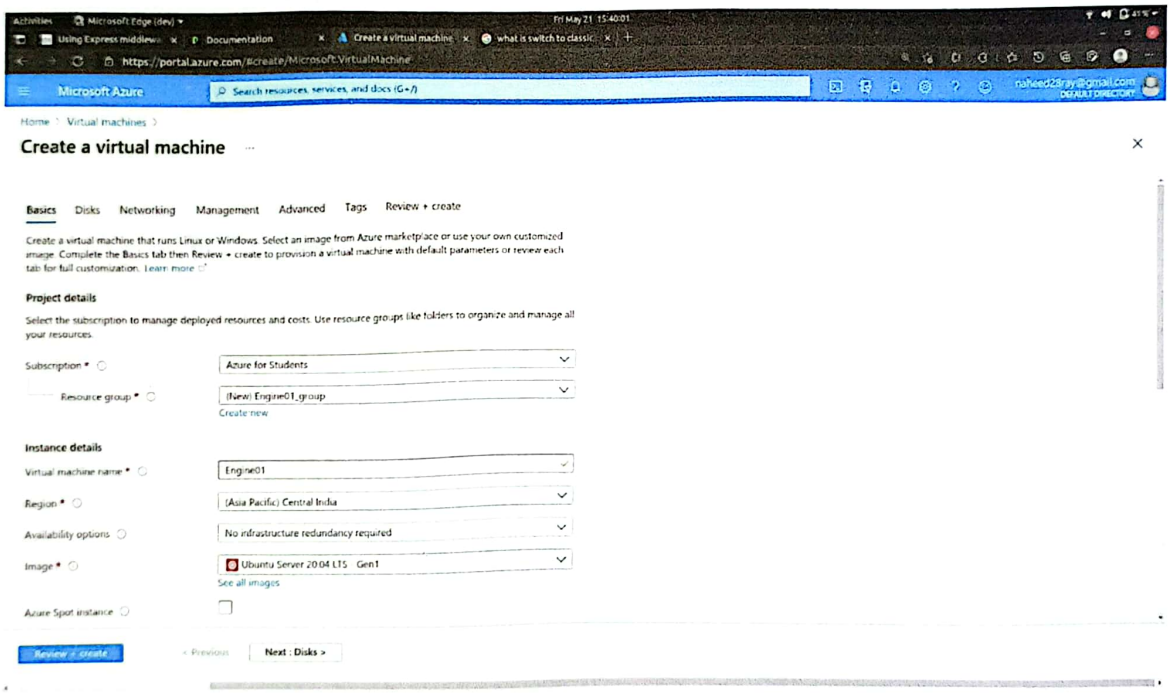

Figure 28 : Microsoft Azure Server for Virtual Machine to run API (1)

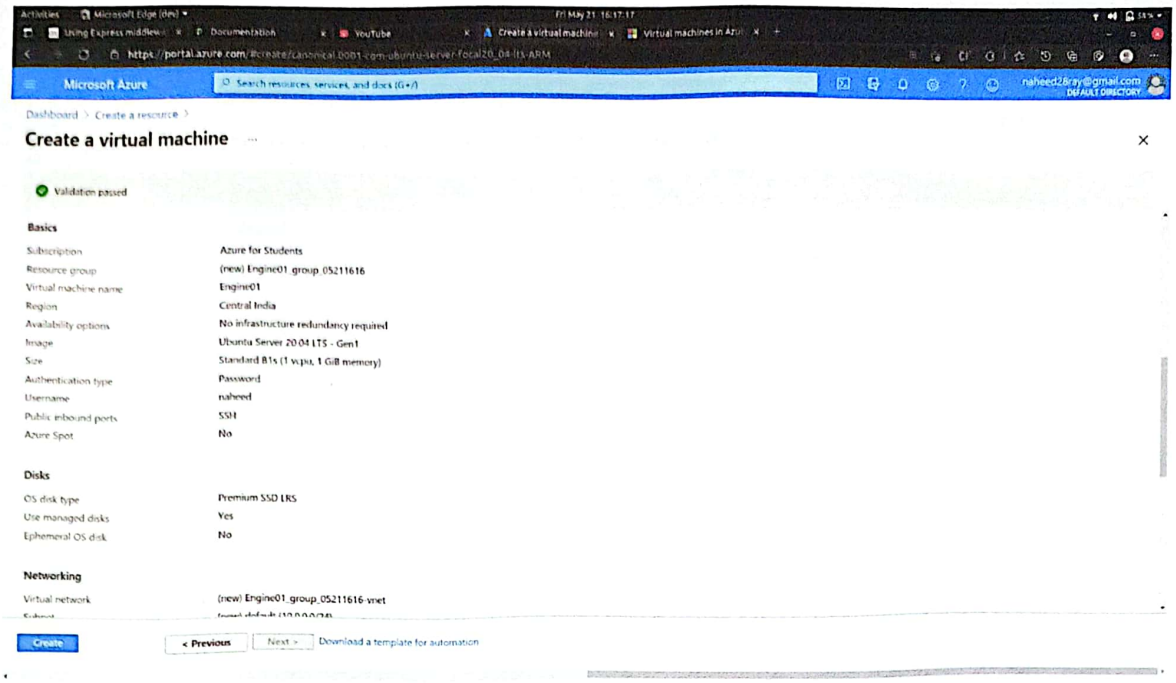

Figure 29 : Microsoft Azure Server for Virtual Machine to run API (2)

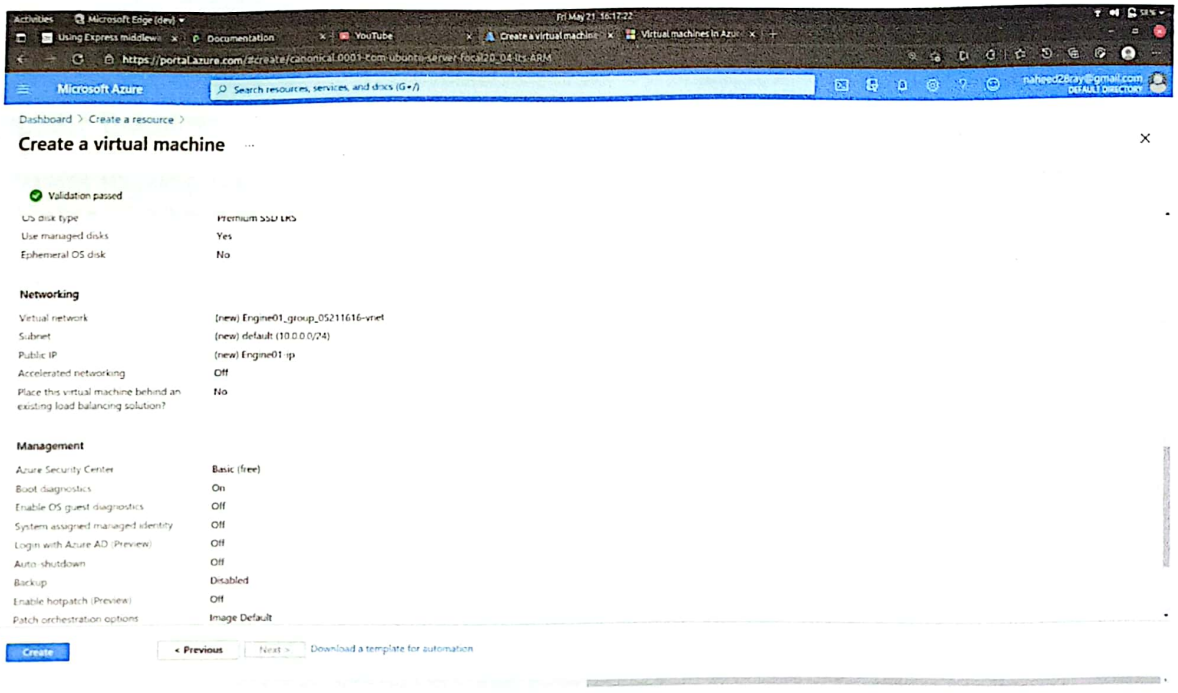

Figure 30 : Microsoft Azure Server for Virtual Machine to run API (3)

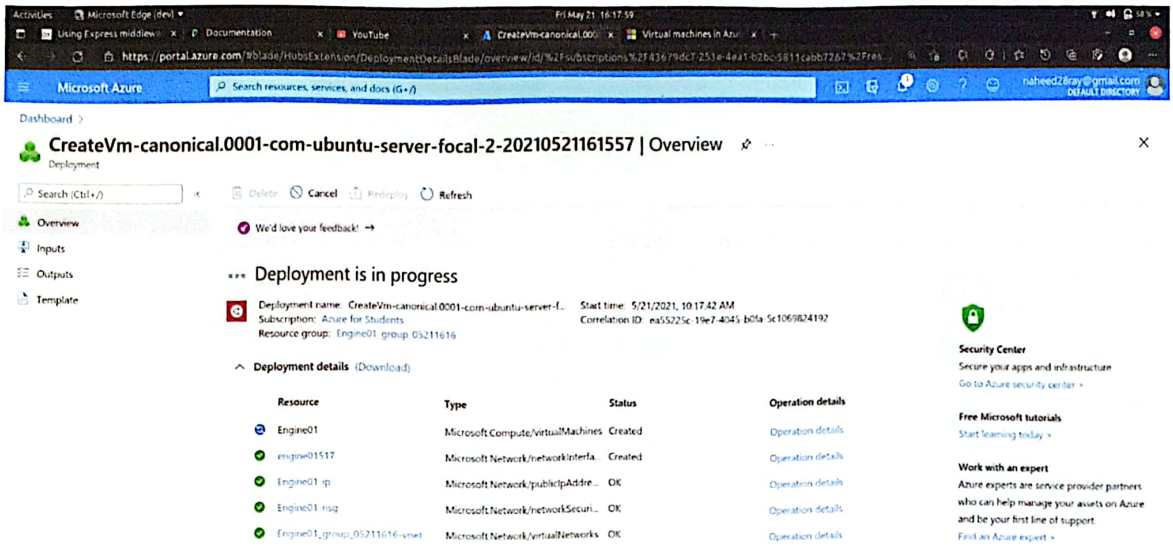

Figure 31 : Microsoft Azure Server for Virtual Machine to run API (4)

4.10.2 Local Server

To test the API and also To build the API we need to use the local server. We have built it with Docker. The local server is running locally now for fast testing.

We used Docker, Linux subsystem, RebbitMQ and Redis in Node Js server.

4.10.3 API Architecture

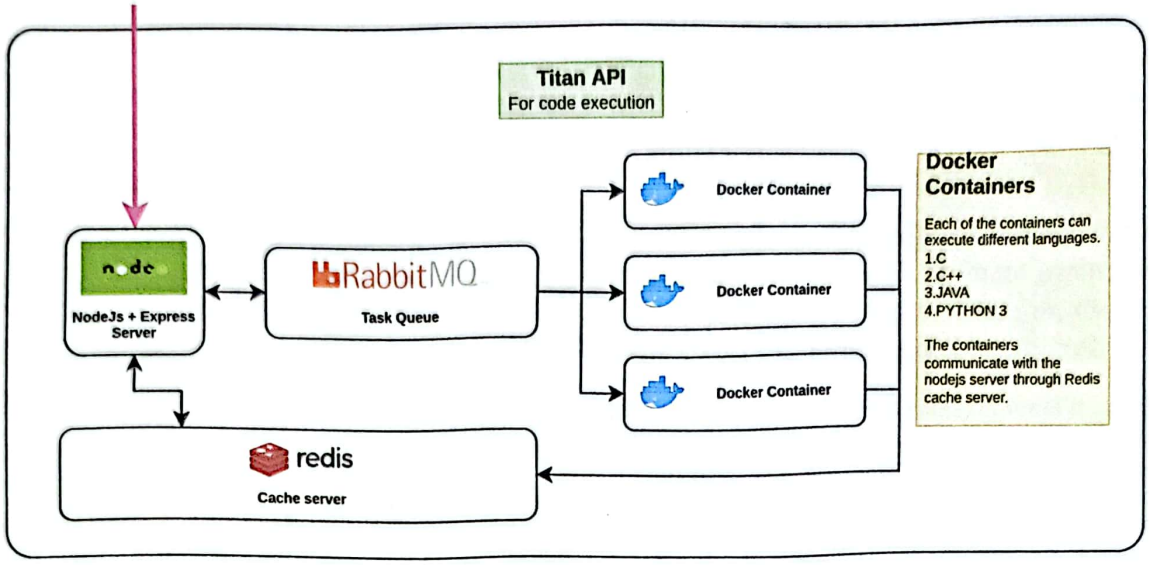

Figure 32 : API architecture

The architecture of our API (Figure 31). We used this technology Docker which runs the main system and redis as a cache server.

Chapter 5

Evaluation

5.1 System Evaluation

Evaluating software for accessibility is similar to website evaluation. The principles of functional accessibility apply, as do the steps of progressive evaluation - though with different evaluation tools at your disposal. This section will discuss techniques for evaluating software on various platforms. For now, Mac OS X, Windows XP/VISTA/7/8/10/11 and Mobile will be covered.

Accessibility evaluation is somewhat subjective, with a multitude of diverse requirements that can sometimes seem at odds. To simplify the evaluation process, consider a progressive approach, performing the easiest checks first and moving on to more in-depth checks from there. Progressive evaluation for software begins with checking keyboard navigation support, checking the interface in high-contrast mode, followed by using a screen reader to check interface labels. Unlike progressive evaluation for the web, it is often not possible or helpful to look at the source code of an application. Following these checks, create a prioritized list of issues for remediation. Evaluating in this order can eliminate the need for time-consuming evaluations when the simplest, most important accessibility support features are not present.

5.2 Acceptance Criteria

As directed by the target, this study requires the product capture and analysis rather than system design and implementation. Therefore, we cannot directly judge results from the first stage of system development if we are successful or not. Great many works is left to future system developers and programmers. Whether the theoretical methodology is practiced properly evaluate the way to investigate and run analyzes to assess whether such a product can be evaluated in a more intelligent way. Adapting the evaluation, those criteria are firstly defined:

- **All objectives and minimum requirements are covered:** Go through the project, whether the outcomes of the project have matched the initial objectives and minimum requirements.
- **Functionality**: System functionality need to be analysis. A comparison of the potential online IDE or online coding System and the current average system will illustrate the system strengths as well as opportunities for further development.
- **User acceptance:** Present the product to the potential end users as well as the system develops from us, get advices and suggestions from the resources in order to make system enhancement. We will also modify the API as user need.
- **User involvement:** We have discussed earlier about how users can use this system. As we said any user can use our IDE after the Signup and Use our API after a Token request.

5.3 User Interface Design

User interface design (UI) or user interface engineering is the design of user interfaces for machines and software, such as computers, home appliances, mobile devices, and other electronic devices, with the focus on maximizing usability and the user experience. The goal of user interface design is to make the user's interaction as simple and efficient as possible, in terms of accomplishing user goals (user-centered design).

Interface design is involved in a wide range of projects from computer systems, to cars, to commercial planes; all of these projects involve much of the same basic human interactions yet also require some unique skills and knowledge. As a result, designers tend to specialize in certain types of projects and have skills centered on their expertise, whether that be software design, user research, web design, or industrial design.

Good user interface design facilitates finishing the task at hand without drawing unnecessary attention to itself. Graphic design and typography are utilized to support its usability, influencing how the user performs certain interactions and improving the aesthetic appeal of the design; design aesthetics may enhance or detract from the ability of users to use the functions of the interface. The design process must balance technical functionality and visual elements (e.g., mental model) to create a system that is not only operational but also usable and adaptable to changing user needs.

Chapter 6

Testing

Software testing is the process of running a program with the objective of finding errors. This is done to improve software or find bugs in existing software. It finds software errors like whether any system of software is working properly or not. Testing is the process of checking that a software program is working as expected and meeting technical as well as business requirements. This can be done at any stage of software development. There are many types of software testing such as unit testing, performance testing, system testing, black box testing, white box testing etc.

6.1 Unit Testing

Unit testing is the process to check the independent and individual unit or module of software. Testing each module separately is called unit testing. This testing process tests whether the modules work independently. The primary goal is to take the smallest piece of testable software, isolate it from the rest of the code and check if it is behaving exactly as expected. Each unit is tested separately before integrating them into modules to test the interfaces between modules.

Each scenario of this application is tested individually and it is ensured that each module produces the desired output for the corresponding input. The application consists of various modules such as page navigation, submission of data by the user to the database, retrieval of data from the database, execution of code, running the API and modifying the code. It is ensured that each module of this system is tested separately to get a suitable result.

It has been ensured that the page navigation is working properly. Main functionality on the button has been tested individually and all the parts are working beautifully. It is ensured that a user Programmer can run his code easily and also can modify it online at any time.

All modal is working perfectly in the IDE. Server is also tested for huge traffic. It can handle infinity loops so the API will not crash.

So the Website has passed all the criteria.

6.2 Performance Testing

Software performance testing is used to determine the speed or effectiveness of a software program or device. Qualitative attributes of software such as reliability, scalability and interoperability may also be evaluated during this testing. The performance of a website comes into picture when some hundreds or thousands of users are accessing the website and performance testing ensures that all the users are getting efficient results in less time. By performance testing we can estimate the

maximum number of users accessing the web site simultaneously and by means of these testing results one can analyze the measures to further improve the performance of the application.

6.3 System Testing

In this test we put the software in different environments and check if the software is compatible with the new environment. Overall system testing is performed by testers. Software is usually the only component of a large computer-based system. System testing is actually a series of tests with the sole purpose of making sure the entire system is working in the new environment. Finally, checking that the software is working properly with other software/hardware systems. There are two types of system testing - black box testing and white box testing.

6.3.1 Black Box Testing

Black box testing is also known as functional testing. Black box testing is testing where the tester has no coding knowledge. In this test, we only check the functionality of the given input software and whether the output produced is working correctly. This type of test depends on the software requirements and features. A black-box can be a software system that you want to test. For example, an operating system like Windows, a website like Google, Oracle or even a database like your own custom application. Under black box testing, you focus on input and output to test the app without knowing the implementation of the software code.

6.3.2 White Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of software testing that examines the internal structure or functionality of an application. In white box testing, the tester has knowledge of coding and internal structure of existing software whether the internal structure is correct or not. It means the tester of this system is the software developer who made the system. The tester chooses the input to exercise the path through the code and checks that the appropriate output is coming. White-box testing can be applied at the unit, integration, and system levels of the software testing process. While traditional testers considered white-box testing to be done at the unit level, it is now more often done and used for system testing. Although this method of test design can uncover many defects or problems, it is likely to miss unrealistic parts of the specification or missing requirements.[5]

Chapter 7

Future Work and Conclusion

7.1 Future Work

Today's modern world is changing day by day. Every person wants a better product. From that perspective makers produce their products with the latest updates. In the web system, there is also getting updated, that's why the product also needs the latest updates. Because of this reason what we made looks too smaller after some time. According to this situation, we are also trying to add some new features to our system.

The Online Code Execution System is a system where we can add more language in future so programmer can use it more. There is enough time and effort needed to create good things. There is nothing to say about the end of the job, there may be a lot to do. This system can also be transformed in a Online Judge or site like Code Forces in future. There is no work done about those features. We will be working on this in the future. Apart from this, various functionality for improving the IDE and user experience will be added to this system.

7.2 Conclusion

The proposed Code Execution Engine System is a very effective and efficient server based online IDE. This software is well tested; it works properly to meet the user requirements as described in the project. Currently, the system is giving all the features and facilities that are needed for a programmer or day to day problem solver to run or execute code easier. Every effort has been made to make this system user-friendly. This system is equipped with enough features according to the user's needs which can fulfill the user's desired system needs.

References

[1] Sphere Research *Coding skills assessment and code execution apis, Sphere Engine.* Available at: https://sphere-engine.com/ (Accessed: November 11, 2022).

[2] Alwyn Botha · *et al.* (2019) *Docker Container Resource Management: CPU, RAM and IO, part 1 - dzone cloud, dzone.com*. Available at: https://dzone.com/articles/docker-container-resource-management-cpu-ram-and-i (Accessed: November 11, 2022).

[3] Genlin (no date) *Microsoft Azure Troubleshooting Documentation, Microsoft Learn.* Available at: https://learn.microsoft.com/en-us/troubleshoot/azure/ (Accessed: November 11, 2022).

[4] Kuenzli, S. (2019) *Docker memory resource limits and a heap of Java, #NoDrama DevOps*. Available at: https://nodramadevops.com/2019/10/docker-memory-resource-limits/ (Accessed: November 12, 2022).

[5] Stacy Nelson (June 2003), NASA/CR–2003-212806 Certification Processes for Safety-Critical and Mission-Critical Aerospace Software (PDF), Ames Research Center, p. 25, [Glossary] White Box Testing: Design-driven testing where engineers examine internal workings of code.

[6] Lai, H. (2017) *Docker and the PID 1 zombie reaping problem, Phusion Blog*. Phusion Blog. Available at: https://blog.phusion.nl/2015/01/20/docker-and-the-pid-1-zombie-reaping-problem/ (Accessed: November 12, 2022).